Proceedings of the 6th International Conference on Educational Data Mining (EDM 2013)

July 6-9, Memphis, TN, USA

S. K. D'Mello, R. A. Calvo, & A. Olney (Eds.)

Published by International Educational Data Mining Society
ISBN: 978-0-9839525-2-7

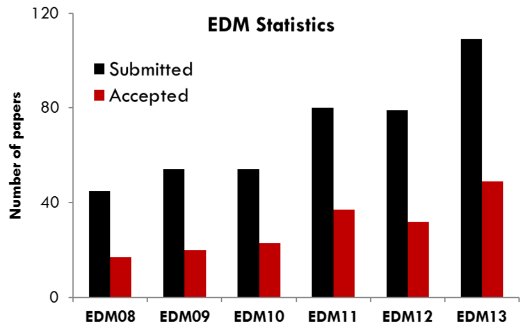
Foreword

Welcome to the sixth installment of the International Conference on Educational Data Mining (EDM 2013), which will be held in sunny Memphis, Tennessee from the 6th to 9th of July 2013. Since its inception in 2008, the EDM conference series has featured some of the most innovative and fascinating basic and applied research centered on data mining, education, and learning technologies. This tradition of exemplary interdisciplinary research has been kept alive in 2013 as evident through the imaginative, exciting, and diverse set of papers spanning the fields of Machine Learning, Artificial Intelligence, Learning Technologies, Education, Linguistics, and Psychology. The EDM 2013 conference program features a rich collection of original research embodied through oral presentations, posters, invited talks, a young researchers track, tutorials, interactive demos, and a panel session.

We received 109 submissions for the main track. Each submission was assigned to three members of the Program Committee based on their areas of expertise. Their reviews were then examined by the Program Chairs who coordinated discussions among the reviewers in order to arrive at a decision. Twenty-seven out of the 109 submissions were accepted as full papers (a 25% acceptance rate) and 22 as short papers (a 45% acceptance rate for full and short papers). An additional 27 were accepted as poster presentations.

In addition to the main track, the conference received 15 submissions to the young researchers track (YRT), 7 to the late-breaking results track, and 9 to the interactive events track. Six of the YRT submissions were accepted into the YRT with an additional two being accepted as posters. Five late-breaking results papers were accepted, as were the nine demo papers.

Each day of the conference will be kick-started by invited talks by three outstanding researchers: Valerie Shute (Florida State University), John Anderson (Carnegie Mellon University), and Ryan Shaun Joazeiro de Baker (Teachers College Columbia University). The main conference will end with a panel session on the future of EDM with panelists including Tiffany Barnes, Ed Dieterle, Neil Heffernan, Taylor Martin, and Sebastian Ventura, and moderated by Sidney DMello and Ryan Baker. The conference will be followed by a series of mini-tutorials led by Agathe Merceron, David Cooper, and Tristan Nixon.



EDM 2013 has broken a number of records. As noted in the figure on the left, we received a record number of 109 submissions this year, a 36% increase from the last two years and a 140% increase since the first EDM conference. This allowed us to accept a larger number of papers for oral presentation (53% increase from EDM 2012) while still maintaining the historic acceptance rate (45% in 2013 compared to average 41% rate from 2008 to 2012). Although EDM has historically been a single-track conference, the increase in the number of submissions and accepted papers led us to a blended approach of both single and dual tracks. Another novelty to EDM is the introduction of short mini-tutorials that will be held on July 9th.

The EDM 2013 conference would not have been possible without the vision and dedicated effort of a number of people. We are indebted to the Program Committee and the additional reviewers for their exceptional work in reviewing the submissions and helping us select the best papers for the conference. We would like to acknowledge Tiffany Barnes and Davide Fossati for

organizing the YRT. Kristy Boyer and Usef Faghihi conceived the brilliant idea of having EDM mini-tutorials and we would like to thank them for putting those together. We would also like to thank Fazel Keshtkar and Sebastian Ventura for organizing the interactive events. A special thanks to Phil Pavlik for managing the website and for joining us on the awards committee. Finally, thanks to the authors for sending us their best work and to all the attendees who bring EDM to life.

Andrew Olney, Phil Pavlik, and Art Graesser would like to thank Ryan Baker for his encouragement to host the 2013 conference, John Stamper for his efforts in securing sponsorships, and Natalie Person for masterminding our incredible banquet. We are indebted to a number of students who worked tirelessly in the months leading up to the conference, including Jackie Maas, Breya Walker, Haiying Li, Nia Dowell, Blair Lehman, Brent Morgan, Carol Forsyth, Patrick Hays, and Whitney Cade. Additional thanks go to Conference Planning and Operations at the University of Memphis, especially Courtney Shelton and Holly Stanford. We would also like to thank our sponsors, The University of Memphis (Office of the Provost), Carney Labs, the Institute for Intelligent Systems, and Pearson, who generously provided funds to help offset registration costs for students. Finally, we would like to gratefully acknowledge the National Science Foundation who provided funds to offset costs for students to attend the YRT and the conference under grant IIS 1340163.

In summary, 2013 appears to be an excellent year for Educational Data Mining. The keynotes, oral and poster presentations, live demos, young researchers track, panel session, mini tutorials, and attendees from all over the world will undoubtedly make the EDM 2013 conference an intellectually stimulating, enjoyable, and memorable event.

Sidney DMello, *University of Notre Dame, USA*
Rafael Calvo, *University of Sydney, Australia*
Andrew Olney, *University of Memphis, USA*

Program Committee

Omar Alzoubi	Carnegie Mellon University in Qatar
Mirjam Augstein Köck	Upper Austria University of Applied Sciences
Tiffany Barnes	North Carolina State University
Kristy Elizabeth Boyer	North Carolina State University
Rafael A. Calvo	University of Sydney
Min Chi	Stanford University
Christophe Choquet	University of Maine
Cristina Conati	University of British Columbia
David G. Cooper	University of Pennsylvania
Richard Cox	University of Edinburgh
Sidney D'Mello	University of Notre Dame
Michel Desmarais	Ecole Polytechnique de Montreal
Hendrik Drachsler	Open University of the Netherlands
Toby Dragon	Saarland Univeristy
Mingyu Feng	SRI International
Katherine Forbes-Riley	University of Pittsburgh
Davide Fossati	Carnegie Mellon University in Qatar
Dragan Gasevic	Athabasca University
Eva Gibaja	University of Cordoba
Janice Gobert	Worcester Polytechnic Institute
Daniela Godoy	ISISTAN Research Institute
Ilya Goldin	Carnegie Mellon University
Joseph Grafsgaard	North Carolina State University
Neil Heffernan	Worcester Polytechnic Institute
Arnon Hershkovitz	Worcester Polytechnic Institute
Roland Hubscher	Bentley University
Sébastien Iksal	University of Maine
Fazel Keshtkar	University of Memphis
Jihie Kim	University of Southern California
Evgeny Knutov	Technische Universiteit Eindhoven
Kenneth Koedinger	Carnegie Mellon University
Irena Koprinska	University of Sydney
Diane Litman	University of Pittsburgh
Ming Liu	University of Sydney
Vanda Luengo	Université Joseph Fourier
Lina Markauskaite	University of Sydney
Noboru Matsuda	Carnegie Mellon University
Manolis Mavrikis	London Knowledge Lab
Riccardo Mazza	University of Lugano
Gordon McCalla	University of Saskatchewan
Agathe Merceron	Beuth University of Applied Sciences
Julià Minguillón	Universitat Oberta de Catalunya
Tanja Mitrovic	University of Canterbury
Jack Mostow	Carnegie Mellon University
Kasia Muldner	Arizona State University
Roger Nkambou	Université du Québec À Montréal

Andrew Olney	University of Memphis
Alexandros Paramythis	Johannes Kepler University
Abelardo Pardo	University of Sydney
Zachary Pardos	Massachusetts Institute of Technology
Philip I. Pavlik Jr.	University of Memphis
Mykola Pechenizkiy	Eindhoven University of Technology
Peter Reimann	University of Sydney
Cristobal Romero	University of Cordoba
Carolyn Rose	Carnegie Mellon University
Ryan S.J.D. Baker	Columbia University Teachers College
John Stamper	Carnegie Mellon University
Jun-Ming Su	National Chiao Tung University
Steven Tanimoto	University of Washington
Sebastián Ventura	University of Cordoba
Katrien Verbert	K.U. Leuven
Stephan Weibelzahl	National College of Ireland
Fridolin Wild	The Open University
Martin Wolpers	Fraunhofer Institute of Applied Information Technology
Kalina Yacef	University of Sydney
Michael Yudelson	Carnegie Mellon University
Amelia Zafra Gómez	University of Cordoba

Additional Reviewers

Baker, Ryan

Fournier-Viger, Philippe

Heffernan, Neil
Hussain, Md. Sazzad
Hussain, Sazzad

Joksimovic, Srecko

Kiseleva, Julia
Kovanovic, Vitomir

Latyshev, Alexey
Li, Nan
Liu, Li
Long, Yanjin

Maclellan, Christopher
Martinez Maldonado, Roberto
Mostow, Jack

Olney, Andrew

Rau, Martina

Sethi, Ricky
Shareghi Najar, Amir
Sodoke, Komi
Stampfer, Eliane

Van Velsen, Martin
Vanhoudnos, Nathan

Wang, Yutao

Zhao, Yu

Table of Contents

Keynotes

Stealth Assessment in Games: Why, What, and How

Valerie J. Shute
Dept. of Educational Psychology and Learning Systems
Florida State University
1114 W. Call Street
Tallahassee, FL 32306-4453
+1 850-644-8785
vshute@fsu.edu

ABSTRACT

You can discover more about a person in an hour of play than in a year of conversation (Plato). For the past 6-7 years, I have been examining ways to leverage good video games to assess and support important student competencies, especially those that are not optimally measured by traditional assessment formats. The term "stealth assessment" refers to the process of embedding assessments deeply and invisibly into the gaming environment. Though this approach produces ample real-time data on a player's interactions within the game environment and preserves player engagement, a primary challenge for using stealth assessment in games is taking this stream of data and making valid inferences about players' competencies that can be examined at various points in time (to see growth), and also at various grain sizes (for diagnostic purposes). In this talk, I will present recent work related to creating and embedding three stealth assessments--for creativity, conscientiousness, and qualitative physics understanding--into Newton's Playground, a game we developed that emphasizes non-linear gameplay and puzzle-solving in a 2D physics simulation environment. I will begin by framing the topic in terms of why this type of research is sorely needed in education, then generally describe the stealth assessment approach, and finally provide some concrete examples of how to do it and how well it works regarding validity issues, learning, and enjoyment from a recent research study.

SHORT BIO

Valerie Shute is the Mack & Effie Campbell Tyner Endowed Professor in Education in the Department of Educational Psychology and Learning Systems at Florida State University. Before coming to FSU in 2007, she was a principal research scientist at Educational Testing Service where she was involved with basic and applied research projects related to assessment, cognitive diagnosis, and learning from advanced instructional systems. Her general research interests hover around the design, development, and evaluation of advanced systems to support learning--particularly related to 21st century competencies. An example of current research involves using immersive games with stealth assessment to support learning—of cognitive and non-cognitive knowledge, skills, and dispositions. Her research has resulted in numerous grants, journal articles, chapters in edited books, a patent, and several recent books such as *Innovative assessment for the 21st century: Supporting educational needs* (Shute & Becker, 2010) and *Measuring and supporting learning in games: Stealth assessment* (Shute & Ventura, 2013).

Discovering the Structure of Mathematical Problem Solving

John R. Anderson
Carnegie Mellon University
Pittsburgh, Pennsylvania, 15213
1-412-268-2788
ja@cmu.edu

ABSTRACT

It is possible to combine multivariate pattern analysis (MVPA) and hidden Markov models (HMM) to discover the major phases that students go through in solving complex problems. I will illustrate this methodology by applying it to the learning of graphical isomorphs of algebra problems. We discovered a sequence of 5 major phases that students went through: An Orient Phase where they identified the problem to be solved, an Encode Phase where they encoded the needed information, a Compute Phase where they performed the necessary arithmetic calculations, a Transform Phase where they performed any mathematical transformations, and a Respond Phase where they generated the answer. The duration of the Compute and Transform Phases were they only ones that distinguished different problem types. Increased duration in these two phases is also associated with making errors. Looking at learning, 2 features distinguished the problems on which participants came to understand a new problem type. First, the duration of late phases of the problem solution increased. Second, there was increased activation in the rostrolateral prefrontal cortex (RLPFC) and angular gyrus (AG), regions associated with metacognition. We think this indicates the importance of reflection to successful learning.

SHORT BIO

John Anderson received his B.A. from the University of British Columbia in 1968 and his Ph.D. from Stanford University 1972. He has been at Carnegie Mellon University since 1978 where he is the Richard King Mellon Professor of Psychology and Computer Science. He has been served as president of the Cognitive Science Society, and has been elected to the American Academy of Arts and Sciences, the National Academy of Sciences, and the American Philosophical Society. He is the current editor of Psychological Review. He has received numerous scientific awards including the American Psychological Association's Distinguished Scientific Career Award, the David E. Rumelhart Prize for Contributions to the Formal Analysis of Human Cognition, the inaugural Dr A.H. Heineken Prize for Cognitive Science, and the Benjamin Franklin Medal in Computer and Cognitive Science.

He is known for developing ACT-R, which is the most widely used cognitive architecture in cognitive science. Anderson was also an early leader in research on intelligent tutoring systems. Computer systems based on his cognitive tutors teach currently mathematics to over 500,000 children in American schools. He has published a number of books including Human Associative Memory (1973 with Gordon Bower), Language, Memory, and Thought (1976), The Architecture of Cognition (1983), The Adaptive Character of Thought (1990), Rules of the Mind (1993), and The Atomic Components of Thought (1998 with Christian Lebiere), and How Can the Human Mind Occur in the Physical Universe? (2007). His current research interest is focused on combining cognitive modeling and brain imaging to understand the processes of mathematical learning.

EDM in a Complex and Changing World

Ryan S.J.d. Baker
Teachers College, Columbia University
525 W. 120th Street
New York, NY 10027 USA
+1 (212) 678-8329
Baker2@exchange.tc.columbia.edu

ABSTRACT

We've started to answer the questions of what we can model through EDM, and we're getting better and better at modeling each year. We publish papers that present solid numbers under reasonably stringent cross-validation, and we find that our models don't just agree with training labels, but can predict future performance and engagement as well. We're making progress as a field in figuring out how to use these models to drive and support intervention, although there's a whole lot more to learn.

But when and where can we trust our models? One of the greatest powers of EDM models is that we can use them outside the contexts in which they were originally developed, but how can we trust that we're doing so wisely and safely? Theory from machine learning and statistics can be used to study generalizability, and we know empirically that models developed with explicit attention to generalizability and construct validity are more likely to generalize and to be valid. But our conceptions and characterizations of population and context remain insufficient to fully answer the question of whether a model will be valid where will apply it. What's worse, the world is constantly changing; the model that works today may not work tomorrow, if the context changes in important ways, and we don't know yet which changes matter.

In this talk, I will illustrate these issues by discussing our work to develop models that generalize across urban, rural, and suburban settings in the United States, and to study model generalizability internationally. I will discuss work from other groups that starts to think more carefully about characterizing context and population in a concrete and precise fashion; where this work is successful, and where it remains incomplete. By considering these issues more thoroughly, we can become increasingly confident in the applicability, validity, and usefulness of our models for broad and general use, a necessity for using EDM in a complex and changing world.

SHORT BIO

Ryan Shaun Joazeiro de Baker is the Julius and Rosa Sachs Distinguished Lecturer at Teachers College, Columbia University. He earned his Ph.D. in Human-Computer Interaction from Carnegie Mellon University. Baker was previously Assistant Professor of Psychology and the Learning Sciences at Worcester Polytechnic Institute, and he served as the first Technical Director of the Pittsburgh Science of Learning Center DataShop, the largest public repository for data on the interaction between learners and educational software.

He is currently serving as the founding President of the International Educational Data Mining Society, and as Associate Editor of the Journal of Educational Data Mining. His research combines educational data mining and quantitative field observation methods in order to better understand how students respond to educational software, and how these responses impact their learning. He studies these issues within intelligent tutors, simulations, multi-user virtual environments, and educational games.

Oral Presentations (Full Papers)

Limits to Accuracy: How Well Can We Do at Student Modeling?

Joseph E. Beck
Computer Science Department
Worcester Polytechnic Institute
josephbeck@wpi.edu

Xiaolu Xiong
Computer Science Department
Worcester Polytechnic Institute
xxiong@wpi.edu

ABSTRACT

There has been a large body of work in the field of EDM involving predicting whether the student's next attempt will be correct. Many promising ideas have resulted in negligible gains in accuracy, with differences in the thousandths place on RMSE or R^2. This paper explores how well we can expect student modeling approaches to perform at this task. We attempt to place an upper limit on model accuracy by performing a series of cheating experiments. We investigate how well a student model can perform that has: perfect information about a student's incoming knowledge, the ability to detect the exact moment when a student learns a skill (binary knowledge), and the ability to precisely estimate a student's level of knowledge (continuous knowledge). We find that binary knowledge model has an AUC of 0.804 on our sample data, relative to a baseline PFA model with a 0.745. If we weaken our cheating model slightly, such that it no longer knows student incoming knowledge but simply assumes students are incorrect on their first attempt, AUC drops to 0.747. Consequently, we argue that many student modeling techniques are relatively close to ceiling performance, and there are probably not large gains in accuracy to be had. In addition, knowledge tracing and performance factors analysis, two popular techniques, correlate with each other at 0.96 indicating few differences between them. We conclude by arguing that there are more useful student modeling tasks such as detecting robust learning or wheel-spinning, and estimating parameters such as optimal spacing that are deserving of attention.

Keywords

Cheating experiments, student modeling, limits to accuracy, knowledge tracing, performance factors analysis

1. INTRODUCTION

The field of educational data mining has seen many papers published on the topic of student modeling, frequently predicting next item correctness (e.g. [1-6]). Next item correctness refers to the student modeling task where the student's past performance on this skill is known, and the goal is to predict whether the student will respond correctly or incorrectly to the current item. This task was the topic of the KDD Cup in 2010. It is typically assumed that data from other students are also available to aid in fitting modeling parameters. This research area certainly appeared to be ripe grounds for rapid improvement, with reported R^2 values for Performance Factors Analysis (PFA; [7]) and Bayesian knowledge tracing [8] of 0.07 and 0.17, respectively [9]. PFA and Bayesian knowledge tracing were two better known, baseline techniques, and their apparent poor performance left tremendous room for improvement by developing more refined modeling techniques.

Researchers tried a variety of approaches to improve accuracy. One natural idea was to consider awarding students partial credit for their attempts. Many researchers use a simple, binary scoring metric of full points for a student who responds correctly on the first attempt with no hints, and zero points for a student who makes any mistakes or requests any hints. Thus, there is no distinction between a student who makes a mistake and corrects himself 3 seconds later, and a student who asks the system to tell him the answer and types it in — both are simply marked as "incorrect." Work on partial credit decreased the amount of credit awarded in proportion to the number of hints requested [3]. By accounting for student partial credit, it improved model accuracy from an R^2 of 0.1903 to 0.1922[1].

Another potential weakness in student models is that the domain models are developed by human experts, who are often guided by intuition. Perhaps an approach that uses data to automatically refine student models will result in a better fit to the data? Across eight datasets where model accuracy was available for the original and the data-generated models, the model fit (un-weighted average, computed by the authors) improved slightly from 0.4143 to 0.4020. However, perhaps the primary outcome of the work was better estimates of the rates at which students learn skills, which is certainly a useful artifact.

Some approaches were possibly larger successes. One underlying assumption is that there is one set of model parameters. For example, all students have the same initial knowledge of a particular skill; all students learn the skill at the same rate, etc. Relaxing that assumption and modeling students as two separate distributions improved R^2 from 0.162 to 0.205, and AUC from 0.74 to 0.77[10]. However, to the authors' knowledge, no one has tried to replicate this work on another dataset, so the results should be treated with skepticism.

Many techniques assume that all students have the same initial knowledge of a particular skill. Such an assumption is clearly incorrect, as student knowledge typically varies considerably. So why not incorporate such flexibility into our models? Some interesting work on extending knowledge tracing allowed student initial knowledge to vary based on initial performance [11]. The main finding was that model fit was notably improved, from an R^2 of 0.0374 to 0.1236. However, on replication, this approach of customizing initial student knowledge was found to perform worse than the baseline knowledge tracing technique with an R^2 of 0.089 vs. 0.1257[2][12]. This later study was also interesting in that it tested different techniques for estimating model parameters,

[1] Note that RMSE, R^2 and AUC values are not comparable across studies due to differing datasets.

[2] The R^2 statistics for both studies were computed by the authors of this paper for consistency.

of the three attempts studied, model fit varied from an R^2 of 0.1203 to 0.1257. There had been prior work experimenting with different methods of parameter estimation with conflicting results about which approach worked better [9, 13]. We find ourselves agreeing with the authors of [12] that "It is not yet clear what features of a specific data set (and the tutor it comes from) are associated with better or worse performance for specific types of student models." By creating a machine-learned ensemble of student models and features, they managed to improve A' from 0.705 to 0.769 [12]. This is a definitely large improvement in model accuracy, but raises questions of interpretability, which we will discuss later in the paper.

A fair question is why improvements in student model accuracy have been so limited? In general, improvements in model accuracy have been minimal, particularly given the relatively low baseline performances. Improving a model with an R^2 of 0.9 is challenging, but improving one that starts at an R^2 of 0.17 should be simpler. The ideas listed above were sensible, but improvements have generally been modest, and often do not replicate across data sets. The results generate two questions:

1. What is it about this prediction problem that makes it difficult?
2. Is there perhaps a much lower upper limit on model accuracy than might otherwise be suspected?

The motivation for this paper was to explore potential reasons behind the inability to create highly accurate models.

2. CHEATING EXPERIMENTS WITH THEORETIC MODELS

Our first investigation into the plausible performance ceiling of student modeling is done using *cheating experiments*. The idea of a cheating experiment is to test a methodology, simulating some non-existent technology as part of it as a means of discovering how well a technique would perform if certain limitations are removed. The key element of a cheating experiment is relaxing certain limitations in scientific knowledge or methodology, but not to create an artifact that is too powerful. For example, for our task of predicting student correct next response, one cheating experiment would be an algorithm that simply peeks into the future, and predicts whatever the student will do. While this approach would certainly be very powerful, it does not give us much guidance about limiting factors on performance as the only conclusion one could draw would be "a student modeling technique that could see the future with perfect accuracy would do a very good job." Therefore, we focus on more limited, but still currently infeasible, extensions of a student model's capabilities.

This paper investigates three aspects of student modeling. First, we explore how a student model would do with a perfect detector of learning [14]. Second, we investigate how important understanding student incoming knowledge is. Third, we examine how a continuous estimate of knowledge would perform.

Thus, the goal of our analysis is to estimate how well we could perform at student modeling if we had a perfect model of several aspects of learner cognition. However, we first give our baseline assumptions, then describe our data, and finally provide baseline model performance when trained on those data.

2.1 UNDERLYING ASSUMPTIONS

We were interested in understanding what factors limit our ability to model the student. One input, typically implicit in student modeling research, is the domain model the maps items to skills (sometimes called a "Q-matrix"). This model enables us to map student performance on an item to a particular skill in the domain. If this aspect of the system is poorly done, model accuracy can suffer. The authors are unaware of any large gains in accuracy by refining a "reasonably constructed" transfer model; we add the restriction as we are certain that refining a randomly generated transfer model would improve accuracy. Recent work [6] found a slight improvement in accuracy from refining models, but the effect was not large. Therefore, we do not consider improvements in the transfer model within this paper.

Furthermore, we assume that we do not know the underlying model generating students' responses. While it is certainly possible to make such an assumption and to use, for example, knowledge tracing, to generate student responses, such an approach assumes far too much, and is of questionable applicability to real-world tutoring scenarios. Therefore, rather than generate hypothetical student responses and estimate our ability to recover our initial models, we simply use the student performance data as provided and compute our predictive accuracy.

2.2 DATA DESCRIPTION AND BASELINE MODEL

For our analyses, we use two datasets. The first is from the ASSISTments (www.assistments.org) web-based tutor. These data are from 343 eighth-grade students (approximately 13 years old) in four classes in urban school districts in the Northeastern United States. There were 86,528 first attempts at responding to a mathematics problem, and students were correct 64.5% of the time. The domain is represented as 104 mathematics skills. Our second dataset is the 2010 KDD Cup dataset, from the Cognitive Algebra Tutor. We used one of the training datasets, and filtered out rows with missing values resulting in 607,026 rows of data with students correct 75.5% of the time. These data are from 574 students working on 158 skills in mathematics. Although both systems involve math skills, they are actually rather different from each other. ASSISTments serves primarily as computer-assisted practice for students' nightly homework and review lessons, while the Cognitive Tutor is part of an integrated curriculum and has more support for learners during the problem-solving process.

For our baseline approach, we have selected the Performance Factors Analysis (PFA; [7]) model. A PFA model takes the form of a logistic regression model, where the independents are the number of correct and incorrect responses, and the difficulty of the item the student is attempting. As PFA estimates the impact on performance by weighting types of learning opportunities differently (correct vs. incorrect responses), it can be seen as a variant of learning decomposition [15]. For our data, we have found that PFA typically does a better job at predicting [9] the data than Bayesian knowledge tracing [8]. Thus it is more appropriate as a baseline metric. For both our baseline approach and our cheating models, we represent data separately for each student on each skill. Thus, when we discuss successive student attempts, we mean successive attempts on the same skill, and ignore intervening problems on other skills.

For performance metrics, we use the Area Under the Curve (AUC) and R^2. AUC is an approximation of A'; it is a commonly used metric when comparing student modeling techniques. The upper limit on AUC is 1.0, and the practical lower limit is 0.5[3]. AUC evaluates techniques based on how well they *order* their predictions. For four problems, if a model predicts that a student has a 95%, 90%, 87%, and 86% chance of responding correctly and the student gets the first two items correct and the next two items incorrect, the AUC will be a perfect 1.0 (assuming a threshold of 50% is used, which would be a poor choice in this scenario). Even though the model predicts the student is likely to get the last two items correct, since those items are *relatively less* likely to be correct than the first two, AUC gives a perfect score. R^2 is based on the squared error between the predicted and actual value, but is normalized relative to the variance in the dataset. A perfect R^2 value is 1.0, while 0 is a lower bound for (non-pseudo) R^2. R^2 is similar to Root Mean Squared Error (RMSE), but is more interpretable due to the normalization step. For example, it is unclear whether an RMSE of 0.3 is good or bad, perhaps a better error could be obtained simply by predicting the mean value? However, an R^2 of 0.8 indicates the model is account for most of the variability in the data. For computational simplicity, we do not use the pseudo- R^2 method such as Nagelkerke in this paper. Neither AUC nor R2 is a perfect evaluation metric, but, combined; they account for different aspects of model performance (relative ordering, and absolute accuracy, respectively) and provide us a basis for evaluating our models.

Table 1 shows performance of the baseline PFA model on both the ASSISTments and KDD Cup data. We can see that the model does not fit the KDD Cup data set as well as the ASSISTments data. Also, the AUC scores are reasonably high, indicating PFA is able to order its predictions relatively well. However, the lower R^2 values indicate the magnitude of the errors is still substantial.

Table 1. Performance of baseline PFA model

Data source	AUC	R^2
ASSISTments	0.745	0.170
KDD Cup	0.713	0.100

2.3 CHEATING MODEL 1: A PERFECT DETECTOR OF LEARNING AND INITIAL KNOWLEDGE

Our first cheating experiment involves what would be a useful piece of technology: a *perfect* detector of the moment of student learning [14]. When a student is practicing a skill, there is hopefully an "aha!" moment where the student has a large jump in understanding the skill. This cheating model simulates the ability to detect such learning. In addition, the model is also aware of whether the students begin using the tutor with knowledge of the skill. Even though it has no data about the student on this skill, it can apply its learning detector on the first attempt when the student sits down. This cheating model behaves by examining the student's next response, and if the student will respond correctly, this model will mark the student as just learned the skill, and predict a correct response.

However, this model is not permitted to cheat for *unlearning* or forgetting. For example, if the model believes the student knows the skill, it will predict a correct response. If the next response is incorrect, it must first make an error on that response by predicting correct, and then may reevaluate whether the student really knows the skill. To make this decision, the cheating model is permitted to peek ahead at the future student responses and decide whether it wants to change its mind about the student's knowledge and mark him as not knowing the skill.

Although this procedure may sound baroque, there are two reasons for it. First, a cheating model that perfectly detected learning as well as forgetting would never make a mistake, and not produce a useful result. Second, most student modeling approaches focus on learning, and ignore the impact of forgetting [8] (with a few exceptions, such as [2]). Since forgetting is often caused by interference [16], and it is difficult to know all the relevant stimuli to which the student is exposed, this aspect would be difficult to model. To be clear, although Cheating Model 1 (CM1) does not have a perfect detector of forgetting, it is not monotonic in its predictions. That is, when presented with evidence the student does not know the skill (i.e., an incorrect response), it is permitted to backtrack in its estimate of student knowledge.

For an example of how CM1 performs, see Table 2. For the first item, CM1 is permitted to know whether the student knows the skill, and so predicts an incorrect student response. For the second item, CM1's perfect detector of learning enables it to realize the student has learned the skill (by peeking ahead at the next response), and so it predicts a correct student response. Two items later, the student makes a mistake, and as CM1 believes the student knows the skill, the model makes an incorrect prediction (bold, underlined entry). The model then peeks ahead to determine whether it is better to ignore this transient slip, or whether it should change its mind about whether the student knows the skill. Since there is a second incorrect response, CM1 can improve its accuracy by believing the student does not know the skill. Thus, CM1 adjusts its predictions to be in best accordance with future student data, but is not permitted to predict the forgetting before it receives direct evidence.

Table 2. Predictions for CM1 and CM2 on sample student performance data. Bold underlined entries indicate incorrect predictions

Correct?	CM1	CM2i	CM2c	CM2m
0	0	0	**<u>1</u>**	**<u>1</u>**
1	1	1	1	1
1	1	1	1	1
0	**<u>1</u>**	**<u>1</u>**	**<u>1</u>**	**<u>1</u>**
0	0	0	0	0
1	1	1	1	1
1	1	1	1	1

For a more formal definition, Figure 1 provides the pseudocode for the Cheating Model #1 (CM1).

On the ASSISTments data, this initial cheating model has an AUC of 0.804 and an R^2 of 0.50. On the KDD Cup dataset, it has an AUC of 0.762 and R^2 of 0.453. Our cheating model clearly outperforms PFA (as it should) on both datasets; AUC is increased by 0.5 or 0.6, and R^2 by 0.35.

[3] Although technically AUC can be below 0.5, in that case the model's accuracy would be improved, and the resulting AUC would be above 0.5, simply by inverting all of its predictions.

```
Cheating Model 1

Input: Student performance data d

for student in d do
    for skill in d[student] do
        previous_prediction = none;
        for problem in d[student][skill] do:
            if (previous_prediction is none) then
                prediction = response;
            else if (previous_prediction == 0) then
                prediction = response;
            else if (previous_prediction == 1) then
                if (previous_prediction was right) then
                    prediction = 1;
                else
                    prediction = response;
                end if
            end if
            previous_prediction = prediction;
        end for
    end for
end for
```

Figure 1. Pseudocode for Cheating Model 1 (CM1)

2.4 CHEATING MODEL 2: WHAT IS THE IMPACT OF KNOWING STUDENT INCOMING KNOWLEDGE?

The first cheating experiment provided substantial gains in model performance. However, these gains are a result of two pieces of non-existent technology: a perfect detector of learning, and a perfect detector of incoming knowledge. What if we ablate the model slightly, and remove its ability to know student incoming knowledge? Thus, our model will retain its ability to detect "aha!" moments by the learner, but cannot necessarily correctly predict the student's first attempt.

To handle imperfect first prediction, we consider three baseline models:

1. A model that assumes the student knows all skills so will respond Correctly (CM2c) on first attempts.
2. A model that assumes the student knows nothing, so will respond Incorrectly (CM2i) on first attempts.
3. A model that assumes the student answers correctly on easy items, instantiated as those items that are answered correctly a Majority (CM2m) of the time across all students.

Table 2 provides an example of how CM2c, CM2i, and CM2m perform relative to CM1. As can be seen, the only difference is on how each of these models predicts the first element. Since CM2c predicts the first response will be correct, it makes an error in prediction. For this example, we have arbitrarily assigned the first item a difficulty of 0.4, so since the majority of students get this item correct, CM2m predicts the current student will. For a more formal definition, the pseudocode for CM2 can be seen in Figure 2

The impact of prior knowledge for the ASSISTments data may be seen in Table 4; assuming the student will respond incorrectly on the first attempt is the best (simple) approach for prediction. For the ASSISTments dataset, the drop-off in accuracy is noticeable: AUC drops from 0.804 to 0.747, just slightly better than the baseline PFA model's 0.745. Similarly R^2 drops from 0.5 to 0.239—a very substantial drop, and moderately better than PFA's 0.17. For ASSISTments, understanding student initial knowledge is important.

In the KDD Cup dataset, as can be seen in Table 5, the results were broadly similar to those for ASSISTments, with one major points of divergence: the impact of understanding student first problem performance is much less dramatic. For the KDD Cup data, the AUC only drops from 0.762 to 0.754. One explanation is in this dataset there are relatively more problems solved per student per skill. Within ASSISTments, on average each student solves 4.7 problems on each skill he works on. In the Cognitive Algebra Tutor, students practice 19 problems per skill. Thus, since more problems are solved by each student in each skill in the Cognitive Algebra Tutor, initial knowledge estimation has a smaller impact on accuracy. ASSISTments is probably on the lower end of amount of practice per skills; while the Cognitive Tutors, due to the integrated curriculum, is probably on the higher end. Here we see another example of the point made in [12] that there is often inconsistency in approaches across datasets. A hypothesis consistent with our data is that a good model of incoming student knowledge is more useful in scenarios when there are fewer data per skill. We suspect modeling prior knowledge is also more effective when students are more heterogeneous; if all students (don't) know a skill, there is little point in modeling their incoming knowledge separately.

```
Cheating Model 2:

Input: Student performance data d

for student in d do
    for skill in d[student] do
        previous_prediction = none
        for problem in d[student][skill] do:
            if (previous_prediction is none) then
                if CM2c:
                    prediction = 1;
                else if CM2i
                    prediction = 0;
                else if CM2m
                    if (problem_difficulty >= 0.5) then
                        prediction = 0;
                    else
                        prediction = 1;
                    end if
                end if
            else if (previous_prediction == 0) then
                prediction = response;
            else if (previous_prediction == 1) then
                if (previous_prediction was right) then
                    prediction = 1;
                else
                    prediction = response;
                end if
            end if
            previous_prediction = prediction;
        end for
    end for
end for
```

Figure 2. Pseudocode for Cheating Model 2 (CM2)

2.5 CHEATING MODEL 3: ESTIMATING CONTINUOUS KNOWLEDGE AND PERFORMANCE

The final cheating model takes a clue from CM2m, which bases its predictions on item difficulty. Rather than simply assuming that a learner knows a skill or does not, CM3 maintains a *degree of knowledge* for each learner. Our semantics are that knowledge and difficulty are both in the range [0,1]. Larger values represent higher degrees of learner knowledge and more difficult items. If item difficulty is less than or equal to knowledge, this model maintains the learner will respond correctly. Otherwise the learner will respond incorrectly to the item. In this manner, it can represent a student who can respond correctly to some items within a skill, but get other items wrong. The intuition is that the model raises knowledge just high enough to account for student correct responses. On observing an incorrect response, it has the option to decrease student knowledge. The reasoning is similar to that for CM1: a model that could increase and decrease knowledge estimates at will (before seeing the student's response) would achieve perfect accuracy. When a student answers an incorrect response, it can decrease its knowledge estimate arbitrarily low, and will lower it enough to account for later incorrect responses.

Table 3 shows how CM3 performs given the same student performance data as before, but also incorporate item difficulty information. CM3, like CM1, is permitted to peek ahead on the first student performance. Since the student responds incorrectly, the student's knowledge is set to be just under what is required. Since the student responds correctly to the next item, the knowledge is increased to 0.7 to be just sufficient. The student responds incorrectly to the next item, but its difficulty is higher than the student's knowledge, so CM3 predicts the student will get the item wrong and no update to student knowledge is required. Two items later, the student responds incorrectly to an item of difficulty 0.65. This response causes the model to make a mistake as this item is lower than the student's knowledge of 0.7. CM3 responds by decreasing the knowledge, not to .649, but to 0.599. The reason is that CM3 looks ahead, and determines what level of knowledge will best predict the current streak of incorrect responses, and sets knowledge to the maximal level. Since the student gets the next item, with a difficulty of 0.6, incorrect, knowledge is set just below that point. A formal definition of CM3 is provided in Figure 3.

Table 3. Example of predictions and updating knowledge estimates for CM3. Bold underlined entries indicate incorrect predictions

Correct?	Item difficulty	Prediction	Knowledge estimate
0	0.4	0	0.399
1	0.7	1	0.7
0	0.8	0	0.7
1	0.6	1	0.7
0	0.65	**<u>1</u>**	0.599
0	0.6	0	0.599
1	0.3	1	0.599

The performance of CM3 on the ASSISTments dataset is seen in the first row of Table 4. CM3, due to its ability to incorporate continuous levels of knowledge, is the strongest performer on the ASSISTments dataset by a large margin. Apparently representing knowledge as a binary value, even with a model with a perfect detector of learning, results in a considerable weakness. Representing gradations of student knowledge appears to be much more effective.

The performance of CM3 on the KDD Cup data is seen in the first row of Table 5. Again, continuous knowledge resulted in strong performance. For the KDD Cup data, we were a bit stymied as to the meaning of "item difficulty". For these results, we used a concatenation of problem name and step name. However, many such pairs were only attempted by 1 student, leading to considerable over-fitting. Using just the problem name suffers from the problem of underspecificity, and gives an AUC and R^2 of 0.798 and 0.442, respectively.

Table 4. Full performance results on ASSISTments data

	Initial knowledge	**Continuous knowledge**	**AUC**	**R^2**
CM3	Known	Yes	0.884	0.634
CM1	Known	No	0.804	0.5
CM2i	Assume incorrect	No	0.747	0.239
PFA			0.745	0.17
CM2m	Based on difficulty	No (except first item)	0.724	0.273
CM2c	Assumed correct	No	0.678	0.266

```
Cheating Model 3:

Input: Student performance data d

for student in d do
    for skill in d[student] do
        knowledge = first_response == 1? first_response.difficulty : first_response.difficulty -.001;
        for problem in d[student][skill] do:
            if (knowledge > difficulty) then
                prediction = 1;
            else
                prediction = 0;
            end if
            if (response == response.previous and response.previous is not none) then
                continue;
            else if (response == 1)
                next_response = response.next;
                while (next_response == 1)
                    knowledge = max (knowledge, next_response.difficulty);
                     next_response = next_response.next;
                end while
            else if (response == 0)
                next_response = response.next;
                if (next_response == 0)
                    while (response == 0)
                        knowledge = min (knowledge, next_response.difficulty - .001)
                        next_response = next_response.next;
                    end while
                else
                    while (next_response == 1)
                        knowledge = max (knowledge, next_response.difficulty)
                        next_response = next_response.next;
                    end while
                end if
            end if
        end for
    end for
end for
```

Figure 3. Pseudocode for Cheating Model 3 (CM3)

3. EMPIRICAL CHEATING EXPERIMENTS

In addition to the theoretic cheating experiments, we also examine data from recent work [12] on ensembling multiple techniques together. This dataset[4] is of interest as it provides the predictions of multiple student modeling techniques as a means of estimating

[4] Kindly provided by the paper authors.

an upper bound on performance. For this work we focus on four approaches to Bayesian knowledge tracing (BKT), and a PFA model. The first two BKT variants use different search techniques to estimate the model parameters; the third approach restricts the training data to find more relevant cases; the fourth variant extends the BKT model by allowing each student to have a different level of initial knowledge:

1. Brute force (BF): estimates model parameters by exhaustively searching the set of initial knowledge, learning, guess, and slip parameters.
2. Expectation maximization (EM): finds the model parameters that maximize the data likelihood.
3. Less data (LD): a variant of knowledge tracing that uses fewer training data.
4. Prior per student (PPS): rather than assuming all students have the same prior knowledge for a skill, it makes initial knowledge conditional on student performance.

Table 5. Full performance results on KDD Cup data

	Initial knowledge	**Continuous knowledge**	**AUC**	**R^2**
CM3	Known	Yes	0.887	0.673
CM1	Known	No	0.762	0.453
CM2i	Assume incorrect	No	0.754	0.353
CM2m	Based on difficulty	No (except first item)	0.713	0.357
PFA			0.713	0.1
CM2c	Assumed correct	No	0.711	0.356

In the original work on ensembling from which these data derive, the authors used machine learning approaches to find the best way of combining the models' predictions to create a more accurate model. Instead of that approach, we will consider how a model that managed to always select the best base model would do. Specifically, we instantiate our model as follows:

```
If student response is correct
    then prediction = max(BF, EM, LD, PPS, PFA)
    else prediction = min(BF, EM, LD, PPS, PFA)
```

Table 6 provides some sample predictions for the five algorithms and shows how our cheating experiment behaves. For each student response, the model selects whichever prediction is closest. Unlike the earlier cheating models, this one does not have explicit assumptions about learning or initial knowledge, but simply picks whichever prediction is closest. Thus, for the third student response, a (perhaps) unexpected incorrect response, this approach simply selects the lowest value. In other words, our empirical cheating experiment postulates the existence of a perfect ensembling approach that always selects the best of its options.

We computed our model's predictions across all 178,000 rows in the provided dataset. The performance of our model and baseline techniques PFA and KT-LD (best performing of the KT techniques [12]) is shown in Table 7. The empirical cheating technique strongly outperformed the baseline techniques, and appears to be somewhat better than the best ensembling technique found, which had an A' (approximately equivalent to AUC) of 0.769. Therefore, there may be room to develop more refined ensembling techniques, and discover additional features in order to improve predictive accuracy. We discuss the utility of this line of research in the Future Work section.

Table 6. Empirical cheating experiment

Correct?	BF	EM	LD	PPS	PFA	prediction
0	0.31	0.31	0.31	0.25	0.34	0.25
1	0.60	0.60	0.59	0.60	0.60	0.60
0	0.35	0.37	0.37	0.29	0.38	0.29
1	0.46	0.47	0.47	0.42	0.47	0.47
1	0.37	0.37	0.37	0.36	0.39	0.39

Table 7. Performance in empirical cheating experiments

	AUC	R^2
Empirical cheating	0.831	0.324
PFA	0.706	0.130
KT-LD	0.701	0.126

One suggestive item in Table 6 is that it appears that each of the student modeling techniques is making fairly similar predictions to its competitors. This phenomenon was also noted when comparing techniques on another data set [17]. To test this idea, we looked across all 178,000 data points, and found that the five student modeling techniques (BF, EM, LS, PPS, and PFA) intercorrelated with each other at 0.92 on average. The prior per student (PPS) model was the most idiosyncratic (and the worst performing), with an average correlation of 0.85 with the other models. If PPS is removed, the remaining four techniques intercorrelated at 0.96, an astonishingly high value. This high number is not an artifact of comparing variants of knowledge tracing with each other: PFA's predictions correlates with KT-LD's at 0.95. It should be noted that each of these techniques has typically been the subject of multiple papers investigating its strengths and weaknesses, and exploring different variations (e.g. item- vs. skill-based PFA [5]). However, it appears the major story is that all of the techniques are in large-scale agreement with each other.

4. MAIN RESULTS AND LIMITATIONS

This paper has estimated likely upper bounds on student modeling performance. Our approach was to consider the basic cognitive factors influencing student performance, and then construct a cheating experiment that perfectly models those factors. On ASSISTments data, we found that the ability to perfectly model student learning, but imperfect information about prior knowledge, led to a model that performed only slightly better than a baseline PFA model. This result is rather surprising.

The other striking result is that, in spite of being an active research area in the EDM, AIED, and ITS communities, competing student modeling approaches make remarkably similar predictions. Given the relative closeness of empirical results to our cheating models, and the high intercorrelations, a plausible conclusion is that the majority of the work in the field of predicting next item correctness has been done, and there are not large gains in performance remaining to be found.

The cheating models described in this paper are extremely powerful, and examine the basic cognitive inputs to student performance. We are unlikely to get perfect detectors of learning any time in the near future. For student modeling approaches that rely on determining when a student has learned, trying to infer incoming knowledge, and account for item difficulty, these cheating models provide a reasonable upper limit on accuracy. However, what of approaches those are not based on cognitive principles? For example, student mistakes could be due to lack of knowledge, or could be due to a careless error. Such careless errors appear to be non-random, as such mistakes have been found to be associated [18] with gaming the system [19], and there is work on *contextual* detectors of slip and guess [13]. The potential improvement from such work is not accounted for by the analyses presented in this paper.

In addition, approaches such as collaborative filtering [20] provide an avenue for non-cognitive approaches to improving student modeling. With collaborative filtering approaches, rather than modeling student knowledge explicitly, instead the goal is to find similar past students and use their performance to make a prediction for the current student (e.g., [21]).

5. FUTURE DIRECTIONS

It is unclear how much additional gain there is from refining student models to achieve ever higher predictive accuracy. Many promising approaches have resulted in little real-world improvement in accuracy. One drawback is the seductive combination of statistical hypothesis testing with increasingly-large datasets. It is possible to find statistically reliable results corresponding to very small effects. Even with a relatively small dataset of 48,000 item responses, a result with a p-value of 0.002 resulted in an improvement of less than 0.001 in R^2 [22]. While larger datasets enable us to estimate such miniscule quantities quite precisely (thus, the low p-value), it raises the question of whether this result *useful* in any way?

We should reflect on why so much effort is being devoted to the problem of predicting student next response. Two candidate answers are that's where the data are, and this task was the goal of the 2010 KDD Cup. Certainly, correctness performance on each item for each student is a very vast source of data. Ten years ago that argument would have been a strong rationale, but now there are large quantities of educational data of all sorts. As a thought experiment, imagine a research result were published in EDM 2014 with a new student modeling approach that achieved an A' of 0.9 (comparable to an AUC of 0.9, but A' has simpler semantics). Effectively that would mean that given a correct and an incorrect student response, this student model could determine which was which 90% of the time. Such an accomplishment would be a major step forward in our capabilities. But, what would we actually *do* with the model? This question is non-rhetorical, as the authors do not have a good answer. To be clear, there are plenty of useful problems our student models could address, such as the probability of a student receiving an "A" in the course, or whether he is ready to move and learn subsequent material.

Ironically, as a field we have settled on a common test problem that has little impact on tutorial decision making or on informing the science of learning. We got to this point for good reasons. Student modeling in ITS is primarily about the estimation of student knowledge. In addition to plentiful data at the item response level, one natural method of validating [23] an instrument is to compute its predictive validity. That is, how well does the measure correlate with things the construct should correlate with. If our model of student knowledge is a good one, it should have a high correlation with student performance on items. Thus, from an instrumentation standpoint our scientific approach is reasonable.

However, while showing that a measure has a high correlation is a necessary condition in validating a measure, it is never a sufficient condition [23]. In other words, constructing a student model with a higher predictive accuracy is not sufficient to create a better estimate of the student's knowledge. As a concrete example, consider ensembling methods, which consider the outputs of different student modeling approaches, and finds a means to combine their predictions with additional features to better predict student performance. Such approaches are in fact successful at noticeably raising the bar (e.g. [12]). However, is there any interpretable component relating to student knowledge? Can we use this model to predict whether an intervention will lead to more learning? If not, then what do we do with the model?

To be clear, this paper does not assert that the field of student modeling is completed. Rather, it makes a more modest claim: the research thread of predicting next item correctness is approaching limits to accuracy, and has probably progressed beyond a useful point. However, there remain several interesting, known problems in student modeling that can inform us about student learning, and have a clear correspondence to improving tutorial decision making.

First, consider the robust learning framework of the Pittsburgh Science of Learning Center. The components of robust learning are preparation for future learning (of related skills), transfer to novel contexts, and retention. Constructing a detector of the first two components of robust learning (e.g., [24]) is a worthwhile modeling goal. Other work has focused on predicting the third component, retention (e.g., [25]).

As a second example, work on the optimal interval to wait before presenting an item on the same skill would be useful. Items presented to close together temporally waste time on repetitive practice; too far apart risks having the student forget and having to relearn [26]. However, such intervals vary by student and skill. This problem can be seen as the complement of retention: how long can we wait before risking the student will forget the item?

As a third example, detectors of student behaviors that are out of bounds of our simplified model of the learner are a useful avenue to explore. Our model is that students are attempting to solve problems, and as a result are learning a little bit each time. But what if the student is bored [27] or frustrated and discouraged [28]? A recent example of such a detector is wheel-spinning [29], named after how a student spins his wheels and goes through the motions of learning, but learning repeatedly does not occur. Detecting and suggesting remediation for, such problems is an interesting third avenue to explore.

ACKNOWLEDGMENTS

We want to acknowledge the funding on NSF grant DRL-1109483 as well as funding of ASSISTments. See here (http://www.webcitation.org/67MTL3EIs) for the funding sources for ASSISTments

REFERENCES

1. Thai-Nghe, N., T. Horváth, and L. Schmidt-Thieme. 2011. Factorization models for forecasting student performance. In proceedings of Educational Data Mining. p. 6-8
2. Qiu, Y., Y. Qi, H. Lu, Z.A. Pardos, and N.T. Heffernan. 2011. Does Time Matter? Modeling the Effect of Time with Bayesian Knowledge Tracing. In proceedings of Fourth International Conference on Educational Data Mining. p.
3. Wang, Y., N. Heffernan, and J.E. Beck. 2010. Representing student performance with partial credit. In proceedings of Educational Data Mining. p. 335-336
4. Wang, Y. and N.T. Heffernan. 2011. Towards Modeling Forgetting and Relearning in ITS: Preliminary Analysis of ARRS Data. In proceedings of Fourth International Conference on Educational Data Mining. p.
5. Gong, Y. and J.E. Beck. 2011. Items, Skills, and Transfer Models: Which really matters for student modeling? In proceedings of Educational Data Mining. p. 81-90
6. Koedinger, K.R., E.A. McLaughlin, and J.C. Stamper. 2012. Automated Student Model Improvement. In proceedings of Educational Data Mining. p. 17-24
7. Pavlik, P.I., H. Cen, and K.R. Koedinger. 2009. Performance factors analysis—A new alternative to knowledge tracing. In proceedings of Proceedings of the 14th International Conference on Artificial Intelligence in Education. p.
8. Corbett, A.T. and J.R. Anderson, 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. User Modeling and User-Adapted Interaction. **4**: p. 253-278.
9. Gong, Y., J.E. Beck, and N.T. Heffernan, 2011. How to Construct More Accurate Student Models: Comparing and Optimizing Knowledge Tracing and Performance Factor Analysis International Journal of Artificial Intelligence and Education.
10. Gong, Y., J.E. Beck, and C. Ruiz. 2012. Modeling Multiple Distributions of Student Performances to Improve Predictive Accuracy. In proceedings of User Modeling and Adaptive Personalization. p. 102-113
11. Pardos, Z. and N. Heffernan. 2010. Modeling individualization in a Bayesian networks implementation of knowledge tracing. In proceedings of User Modeling, Adaptation, and Personalization. p. 255-266
12. Pardos, Z., R.S.J.D. Baker, S.M. Gowda, and N.T. Heffernan, 2012. The Sum is Greater than the Parts: Ensembling Models of Student Knowledge in Educational Software. ACM SIGKDD Explorations. **13**(2).
13. Baker, R.S.J.d., A.T. Corbet, and V. Aleven. 2008. More Accurate Student Modeling Through Contextual Estimation of Slip and Guess Probabilities in Bayesian Knowledge Tracing. In proceedings of Intelligent Tutoring Systems. p. 406-415
14. Baker, R.S.J.d., A.B. Goldstein, and N.T. Heffernan, 2011. Detecting Learning Moment-by-Moment. Journal of Artificial Intelligence in Education. **21**(1-2): p. 5-25.
15. Beck, J.E. and J. Mostow. 2008. How who should practice: Using learning decomposition to evaluate the efficacy of different types of practice for different types of students. In proceedings of Ninth International Conference on Intelligent Tutoring Systems. p. 355-362
16. Anderson, J.R., 1993. Rules of the Mind. Hillsdale, NJ: Lawrence Erlbaum Associates.
17. Pardos, Z., S.M. Gowda, R.S.J.d. Baker, and N.T. Heffernan. 2011. Ensembling Predictions of Student Post-Test Scores for an Intelligent Tutoring System. In proceedings of Educational Data Mining. p. 189-198
18. Gong, Y., J.E. Beck, N.T. Heffernan, and E. Forbes-Summers. 2010. The Impact of Gaming (?) on Learning. In proceedings of International Conference on Intelligent Tutoring Systems. p.
19. Baker, R.S.J.d., et al. 2006. Adapting to When Students Game an Intelligent Tutoring System. In proceedings of Proceedings of the 8th International Conference on Intelligent Tutoring Systems. p. 392-401
20. Breese, J., D. Heckerman, and C. Kadie. 1998. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In proceedings of Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence. p.
21. Toscher, A. and M. Jahrer, 2010. Collaborative filtering applied to educational data mining. Journal of Machine Learning Research.
22. Li, S., X. Xiong, and J.E. Beck. 2013 (under review). Modeling student retention in an environment with delayed testing. In proceedings of Educational Data Mining. p.
23. Crocker, L. and J. Algina, 1986. Introduction to Classical and Modern Test Theory. Fort Worth: Harcourt Brace Jovanovich College Publishers. 482.
24. Baker, R.S.J.d., S.M. Gowda, A.T. Corbett, and J. Ocumpaugh. 2012. Towards Automatically Detecting Whether Student Learning is Shallow. In proceedings of Intelligent Tutoring Systems. p. 444-453
25. Wang, Y. and J.E. Beck. 2012. Using Student Modeling to Estimate Student Knowledge Retention. In proceedings of Educational Data Mining. p. 201-203
26. Pavlik, P.I. and J.R. Anderson, 2005. Practice and Forgetting Effects on Vocabulary Memory: An Activation-Based Model of the Spacing Effect. Cognitive Science. **29**(4): p. 559-586.
27. Baker, R.S.J.d., S.K. D'Mello, M.M.T. Rodrigo, and A.C. Graesser, 2010. 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): p. 223-241.
28. Arroyo, I., B.P. Woolf, D. Cooper, W. Burleson, and K. Muldner. 2011. The Impact of Animated Pedagogical Agents on Girls' and Boys' Emotions, Attitudes, Behaviors and Learning. In proceedings of Advanced Learning Technologies. p. 506-510
29. Beck, J.E. and Y. Gong. 2013. Wheel-spinning: student who fail to master a skill. In proceedings of Artificial Intelligence in Education. p. (in press)

Student Profiling from Tutoring System Log Data: When do Multiple Graphical Representations Matter?

Ryan Carlson
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA, USA
ryancarlson@cmu.edu

Konstantin Genin
Department of Philosophy
Carnegie Mellon University
Pittsburgh, PA, USA
kgenin@andrew.cmu.edu

Martina Rau
Human-Computer Interaction Institute
Carnegie Mellon University
Pittsburgh, PA, USA
marau@cs.cmu.edu

Richard Scheines
Department of Philosophy
Carnegie Mellon University
Pittsburgh, PA, USA
scheines@cmu.edu

ABSTRACT

We analyze log-data generated by an experiment with Fractions Tutor, an intelligent tutoring system. The experiment compares the educational effectiveness of instruction with single and multiple graphical representations. We extract the error-making and hint-seeking behaviors of each student to characterize their learning strategy. Using an expectation-maximization approach, we cluster the students by learning strategy. We find that a) experimental condition and learning outcome are clearly associated b) experimental condition and learning strategy are not, and c) almost all of the association between experimental condition and learning outcome is found among students implementing just one of the learning strategies we identify. This class of students is characterized by relatively high rates of error as well as a marked reluctance to seek help. They also show the greatest educational gains from instruction with multiple rather than single representations. The behaviors that characterize this group illuminate the mechanism underlying the effectiveness of multiple representations and suggest strategies for tailoring instruction to individual students. Our methodology can be implemented in an on-line tutoring system to dynamically tailor individualized instruction.

1. INTRODUCTION

Multiple graphical representations (MGRs) are ubiquitous in math and science instruction: they are frequently used to emphasize complementary conceptual interpretations of complex learning materials. Fraction instruction is one domain in which graphical representations, such as number lines, pie-charts, and rectangles are used to help students overcome the difficulty of the material. Although the educational psychology literature suggests that requiring students to translate between representations supports the creation of deep knowledge structures [6], the experimental results are somewhat ambiguous [1] and the mechanisms underlying these advantages are not well understood [2].

Because student interaction with intelligent tutoring systems (ITSs) generates very fine-grained behavioral and outcome data, these systems are well-suited for conducting experiments on the effect of MGRs on learning outcomes [14]. Machine learning methods can be profitably applied to identify the kinds of students whose learning outcomes are improved by MGRs and the factors mediating their success [19]. Such insights enable developers of ITSs to design individualized instructional support that can make learning with MGRs even more effective. This may involve encouraging students to reflect on the material with self-explanation prompts [17] or detecting ineffective strategies and implementing interventions on-the-fly. Work in the latter area ranges from detecting abuse of the ITS hint system and other "gaming" behaviors [8, 7] to providing spontaneous help to students lacking the metacognitive skills to know when they could use a hint [3, 4, 5].

Prior research conducted on elementary-school students working with a Fractions Tutor suggests that prompting students to self-explain while working with MGRs improves their educational effectiveness [17]. Subsequent studies examining error-rate, hint-use and time-spent in tutor's log failed to identify variables that mediate the effectiveness of MGRs [16]. The mechanisms by which multiple graphical representations improve learning outcomes remain poorly understood.

We conjecture that previous efforts to identify mediating factors were frustrated by heterogeneity in the problem-solving habits and behaviors of the student population under investigation. Using a mixture modeling technique, we cluster students by the patterns of interaction with the tutor in the log-data that characterize their learning strategy. Clustering based on student characteristics has proved successful in grouping students into meaningful subpopulations across both collaborative [15] and individual [9, 13, 20] educational environments.

Four strategic profiles emerge from our analysis, each with a natural interpretation. Two of the profiles are characterized by a low propensity to seek help from the tutor. In one of these the students are simply confident: they make few errors, solicit little help and don't seem to need any. In the other the students are reluctant to solicit help even though they seem like they need it: they make a relatively large number of mistakes but make little use of the support mechanisms the tutor provides. We characterize this second class as "stubborn" without intending any pejorative connotations. A third class is highly interactive: they make many mistakes, seek assistance readily and frequently exhaust the hints available in a given problem. Students in the fourth class occupy a middle ground between the interactive and the stubborn: they make an average number of mistakes and will eventually seek help when they are having trouble.

We proceed to explore how the experimental conditions affect post-test outcomes. Confirming previous results [16], we find that students in the multiple-representation condition had greater learning gains than those in the single-representation condition. MGRs seem to have a robust and positive effect on long-term knowledge consolidation. We then explore the effect of multiple representations in the sub-populations defined by each strategic profile. We first establish independence between learning strategy and experimental condition. This suggests that we are detecting pre-existing strategic profiles, rather than artifacts of the experimental setup. Most interestingly, we discover that learning gains from MGRs depend heavily on learning strategy. Students exhibiting a "stubborn" profile profited substantially from instruction with multiple rather than single representations. For the remaining students, experimental condition and learning gain were independent. We conjecture that "stubborn" students lack the metacognitive skills to judge when their learning strategies are failing. These students are the most sensitive to pedagogical decisions because they are the least equipped to structure and manage their own learning.

Section 2 of what follows describes the initial experiment and elaborates on the differences between the representational conditions. We describe our feature extraction process and modeling decisions in Section 3. Section 4 summarizes the results of the model estimation and statistical analysis of the effects of multiple representations at the population and sub-population levels. We suggest profitable future directions in Section 5.

2. EXPERIMENT

In the Spring of 2010, Rau conducted an experiment wherein 290 4$^{\text{th}}$ and 5$^{\text{th}}$ grade students worked with a Fractions Tutor for about 5 hours of their mathematics instruction. Students were randomly assigned to one of five experimental conditions, which varied by the frequency with which students would be presented with a new fraction representation (see Figure 1). Students in the SINGLE representation condition worked exclusively with either a number line, a circle or a rectangle. Students in the FULLY INTERLEAVED condition saw a different representation than was used in the preceding problem. Students in the intermediate conditions went longer before seeing a different representation.

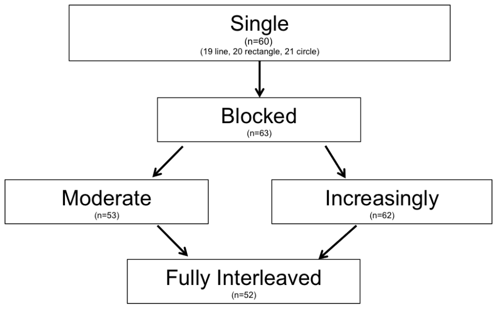


Figure 1: A partial ordering of experimental conditions by the frequency with which a new representation is presented.

When interacting with different graphical representations of fractions, students were able to drag-and-drop slices of a pie chart, for example, into separate areas. They were also able to experiment with changing the number of subdivisions in each graphical representation. Students received a pre-test on the day before they began working with the tutor and an immediate post-test on the day after they finished. Students also took a delayed post-test a week after the first. Previous investigation found that students in the MGR conditions significantly outperformed students in the single representation condition on the delayed post-test [16, 18].

3. METHOD

We proceed in three stages: (1) we extract features characterizing error and hint-seeking behavior from the data logs, (2) we transform the longitudinal log data into a cross-sectional form, with one observation per student, and (3) we estimate a mixture model to identify sub-populations of students, using AIC and BIC to select the number of classes.

Once we have clustered our students by their learning strategy, we investigate the interaction between the strategies and the experimental conditions. We construct a contingency table binning the experimental conditions into the clusters estimated by the mixture model. We then run a Chi-squared test for independence between experimental condition and learning strategy. Chi-squared tests are also run to investigate dependence between pre-test outcome and strategy, strategy and post-test outcome and the conditional dependence of outcome and experimental condition, given a strategic profile.

3.1 Extracting Features

The Fractions Tutor captures a detailed log of each student's interactions with the tutor. It stores a time series of correct and incorrect answers, hint requests, interface selections and durations between interactions. Previous analysis [16] extracted the average number of errors made per step, the average number of hints requested per step, and the

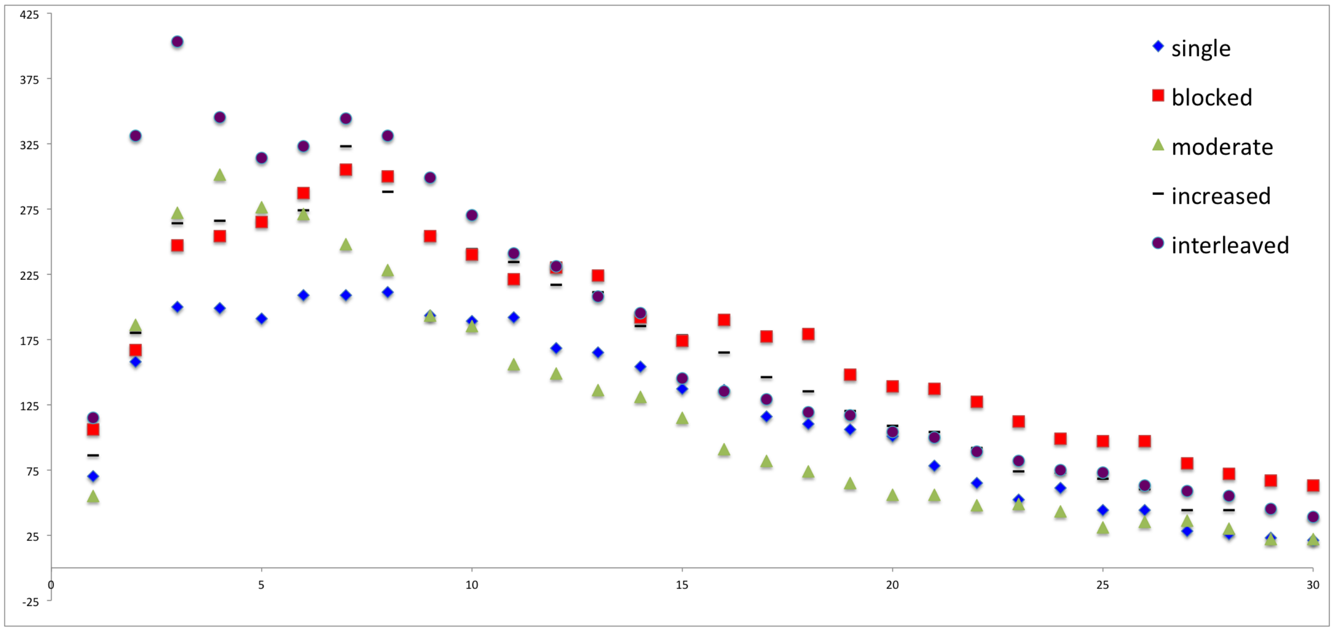


Figure 2: The x-axis represents the n_{th} interaction with the tutor across all problems. The y-axis is the total number of hints requested at the n_{th} step.

average time spent per step from the log data. These variables were used to characterize gross behavioral strategies and dispositions. Similarly, we include the average number of hints requested (HintsRequested) and number of errors (NumErrors) made per *problem* by each student. We also extract the average number of bottom-out hints (NumBOH) per student per problem – this is the average number of times a student exhausts the available hints in a given problem. We also note that it is not always the average of these features that best characterizes a student. For example, examination of the distribution of hints requested per step across experimental condition, shows a telling picture.

Note that students who received only one representation start out requesting the fewest hints, but students in the moderate condition eventually need fewer (see Figure 2). Also, students in the interleaved condition tend to request many hints in the early steps of a problem, potentially reflecting the cognitive load associated with translating between representations [1]. Such considerations suggested that exploiting the timing of student interactions within a problem might expose structural features obscured by stepwise averages (as used in [16]). We fit geometric distributions to the number of steps taken before the first hint request (FirstHintGeometric) and to the number of errors before the first hint (StubbornGeometric). The estimated parameter is used to characterize the student's hint-seeking propensity in general and hint-seeking propensity when faced with adversity. For example, students in the first quintile of StubbornGeometric seek help soon after making a mistake, whereas students in the fifth quintile don't change their hint-seeking behavior even after making a large number of errors. Students in the first quintile of FirstHintGeometric are likely to request hints early in a problem, whereas students in the fifth quintile are unlikely to request hints at any point.

3.2 Expectation-Maximization Clustering

Expectation-Maximization (EM) clustering is a modeling technique that determines subtypes based on multinomial distributions. We use the model to categorize students into subpopulations using discretized versions of the features described above. Table 1 shows summary statistics and cut-off points for the extracted features. The model maps a set of observed categorical variables onto a set of inferred classes.

We note that the categorical nature of the model has the potential to add some noise, since we must select numeric cutoffs to transform our variables into nominals. However, categorical models can offer greater interpretability by allowing us to organize our data into a small set of variables, which forms the basis for categorizing students into a small set of meaningful, homogenous groups. Furthermore, it is not unreasonable to suspect that our variables are in some sense "truly" categorical [10, pp8–9]. EM clustering requires a relatively small set of variables to train the model. As the number of training variables increases, the number of model parameters blows up and the model becomes overspecified.

Unlike some common clustering algorithms (e.g., k-means), EM produces "fuzzy" clusters (i.e., probability distributions over features for each class). We use these probability distributions in our qualitative discussion about the subpopulations (Section 4.1), however we ultimately need to identify each student's most likely class. For each student s and class c we calculate

$$\arg\max_{c} P(S = s \mid C = c) \tag{1}$$

where the probabilities are determined by the EM algorithm.

Table 1: Summary Statistics for Variables Used in Clustering

	mean	sd	median	min	max	20%	40%	60%	80%	100%
HintsRequested	0.78	1.27	0.34	0	11.22	0.06	0.19	0.5	1.31	11.22
NumErrors	2.21	1.27	1.92	0.34	8.39	1.15	1.7	2.18	3.19	8.39
FirstHintGeometric	0.35	0.27	0.27	0.04	1	0.13	0.2	0.33	0.57	1
Stubborn Geometric	0.36	0.21	0.31	0.07	1	0.19	0.27	0.38	0.47	1
NumBOH	0.04	0.08	0	0	0.62	0	0	0.01	0.05	0.63

We still need to fix N, the number of classes. We use two complexity-penalized log-likelihood scores to select an appropriate N: Akaike information criterion (AIC) and Bayesian information criterion (BIC). Plotting these statistics as we increment the number of classes, we look for a "knee" where both statistics either bottom-out or level off to identify the optimal value of N. To run analysis, we used `poLCA`, a freely available R package.[1]

4. RESULTS

In the sections that follow we analyze the results of our clustering algorithm. We describe the strategic profiles that were generated and characterize the students fitting each profile. We then consider the relationships between our variables of interest: *(a)* adjusted delayed post-test score, *(b)* experimental condition, and *(c)* learning strategy. Specifically, we run a series of Chi-squared tests for independence to determine how each variable relates with the others, commenting on the importance of each comparison. Finally, we explore the stability of these classes, which bears on whether future systems could detect students' strategic profiles in real time.

4.1 Exploring the Learning Strategies

Figure 3 shows the parameter selection process described in Section 3.2. Note that we chose to model four classes because BIC bottoms out and AIC levels off at that point.

After selecting the appropriate N parameter, we extract membership probabilities for the individual students. Given a strategic profile, we can estimate the probability distribution over each feature, and use Equation 1 to identify the most likely profile for each student.

The feature distributions over each profile are represented graphically in Figure 4. Each feature is listed along the horizontal x-axis, the value each variable takes is along the front-to-back y-axis, and the probability that the feature takes that value is given along the vertical z-axis. For example, consider the HintsRequested feature (average hints requested per problem) in the "interactive" class. In that class, with high probability, students requested many hints (i.e., the highest categorical value for hints) per problem on average. As another example, students in the "moderate" class are more likely to make a moderate number of errors,

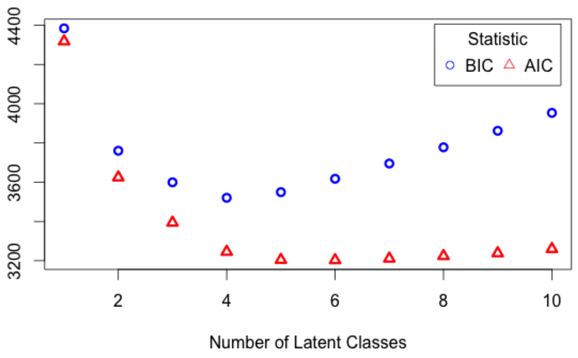


Figure 3: AIC and BIC over increasing number of clusters. BIC bottoms out and AIC levels off at four clusters, so we conclude that four clusters best fits the data.

though other error levels also occur with nontrivial probabilities. Lower values of FirstHintGeometric and StubbornGeometric indicate a steep geometric slope, corresponding to a higher hint-seeking propensity and stubbornness, respectively.

How do we interpret cluster membership? Students in Class 1 are "Moderate", they ask for a moderate number of hints, make a moderate number of errors, and are moderately responsive to the interface. Students in Class 2 are "Interactive", they make a lot of errors, but respond by requesting many hints. These students are proactive in asking for help and are not shy about using the resources the Fractions Tutor makes available. Students in Class 3 are "Confident", they don't ask for hints, but they don't seem to need them (since they make few errors). Finally, we call students in Class 4 "Stubborn" because they are fairly mixed in error-profile but they don't respond to mistakes with hint-requests. These students are not using all the resources that the Fractions Tutor makes available.

4.2 Condition and Outcome

We use normalized learning gain at the delayed post-test as our measure of student improvement.

$$\text{Learning Gain} = \frac{\text{DelayedPostTest} - \text{PreTest}}{1 - \text{PreTest}}$$

[1] `http://userwww.service.emory.edu/~dlinzer/poLCA/`

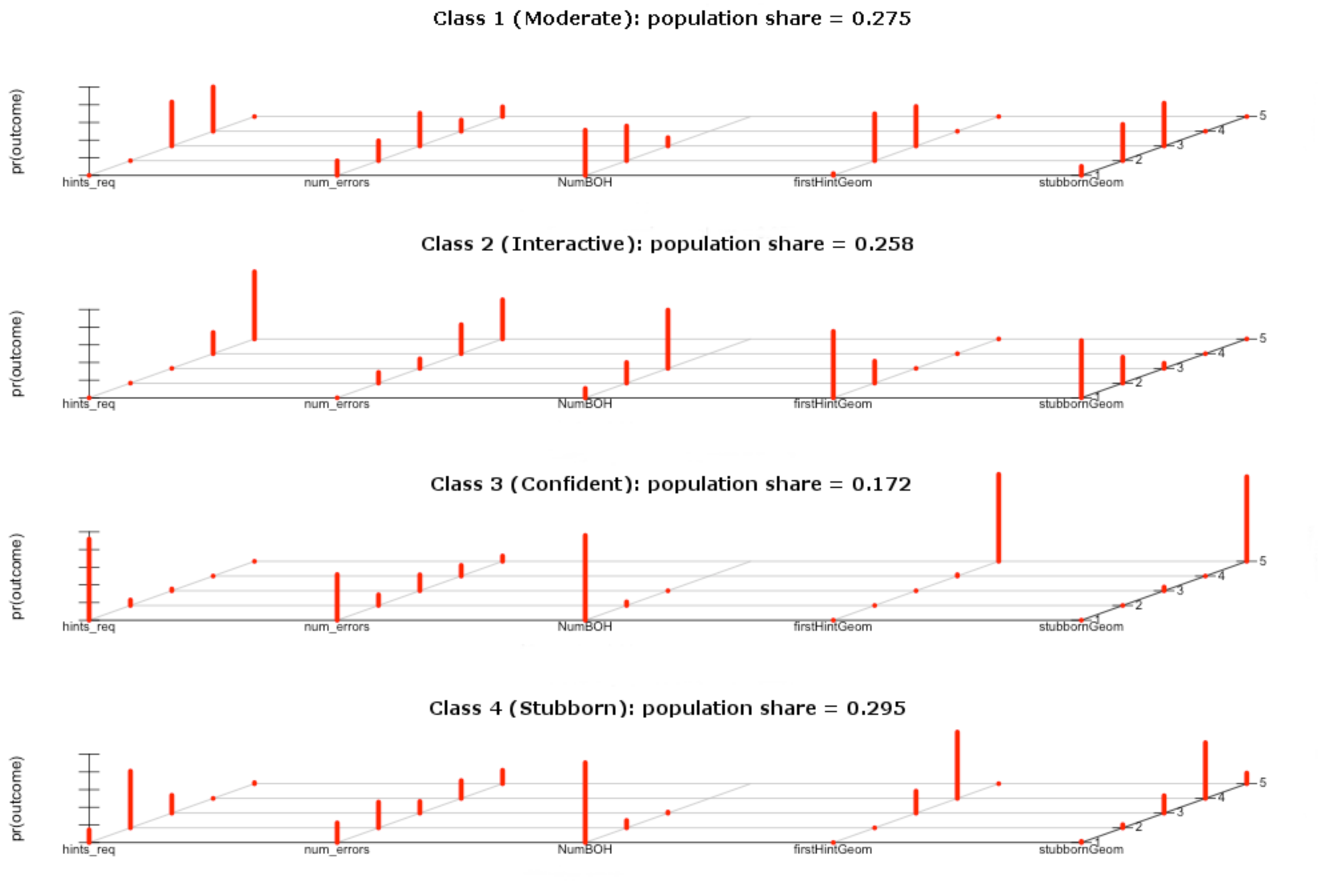


Figure 4: Visualization of feature distributions for each learning profile. The left-to-right x-axis identifies each feature, the front-to-back y-axis identifies which value that feature takes, and the top-to-bottom z-axis describes the probability that the feature takes the value. Thus, given a feature and a class, the z-axis also describes the probability distribution over that feature in that class.

We then construct terciles of the Adjusted Delayed Post-Test Score and run a Chi-squared test for independence of outcome from experimental condition. Confirming previous results, we reject independence at a p-value of .024 (see Table 2). As expected, students in the multiple representation conditions were more likely to be in the second or third tercile of adjusted delayed post-test score, whereas students in the single representation condition were more likely to be in the first.

4.3 Learning Strategy and Test Scores

We would expect that a student's learning strategy would predict (and perhaps cause) their ultimate educational outcome. To test this intuition, we calculate a Chi-squared statistic for independence of learning strategy from normalized delayed post-test gain. We reject independence at a p-value of .0075 (see Table 3). The behaviors encoded by strategic profile seem highly relevant to knowledge consolidation in the long run. Students in the moderate class are found mostly in the second and third tercile. These students are implementing a subtle but effective strategy. Their moderation in hint-seeking indicates a level of self-reflectiveness

	33%	66%	99%
blocked	14	29	20
increased	22	20	20
interleaved	13	21	18
moderate	18	13	22
single	30	13	17

$\mathcal{X}^2 = 17.65$, df $= 8$, p-value $=$ **0.024**

Table 2: Experimental Condition by Tercile of Adjusted Delayed Post-Test Score.

that we would expect from students with highly developed metacognitive skills. Students in the interactive class are characterized by a high number of errors, so we are not surprised to find them represented mostly in the first and second terciles. These students are the most likely to exhaust all the hints available in a given problem. If one were looking for students engaging in "gaming" behavior this would be the class to search, perhaps using techniques from [7]. As one would expect, the confident students are likely to end up in the third tercile. The stubborn students are clustered at the extremes: they are more likely to end up in the first or third tercile than the second.

	Learning Gain			Pre-Test		
	33%	66%	99%	33%	66%	99%
moderate	20	35	29	30	32	22
interactive	33	26	14	37	27	9
confident	13	15	22	5	14	31
stubborn	31	20	32	26	22	35

Learning Gain: $\mathcal{X}^2 = 17.52$, df = 6, p-value = **0.0075**
Pre-Test: $\mathcal{X}^2 = 42.3764$, df = 6, p-value = **<0.001**

Table 3: Learning Strategy by Tercile of Normalized Delayed Post-Test and Pre-Test Score

Although we implicitly account for the pre-test scores in our learning gain metric, we also investigate the relationship between learning strategy and pre-test scores (Table 3). As expected, we reject independence between strategic profile and pre-test score, suggesting that these profiles are genuinely meaningful descriptions of student behavior.

Although pre-test score and strategic profile are dependent, the average pre-test score for the "stubborn" students does not differ significantly from the rest of the population.[2] Pairwise t-tests between the four profiles show significant differences in mean pre-test score for all pairs except stubborn and moderate. This analysis suggests that the dependence we detect between experimental condition and outcome for the "stubborn" students does not hinge essentially on pre-test score. If pre-test is an accurate proxy for preparedness, the stubborn students do not occupy a preparedness "sweet-spot" that makes multiple representations uniquely effective. Rather, it seems to be their unique strategic profile that accounts for the effectiveness of MGRs.

4.4 Condition and Learning Strategy

We may also worry that experimental condition is inducing learning strategy. If this were the case, we would suspect that we were picking up on artifacts of the experimental design rather than pre-existing student profiles. However, using the Chi-squared test, condition and cluster membership appear independent (see Table 4).[3] To anticipate Simpson's paradox-type worries, we collapse all four multiple representation conditions (blocked, moderate, increased, interleaved) into a single "multiple representation" condition, but still find independence.[4] These results suggest that our method is detecting genuine student profiles, independent of experimental condition.

	mod.	inter.	conf.	stub.
blocked	13	15	10	25
increased	21	16	10	15
interleaved	17	18	7	10
moderate	18	10	12	13
single	15	14	11	20

$\mathcal{X}^2 = 12.85$, df = 12, p-value = 0.38

Table 4: Experimental Condition by Learning Strategy

4.5 Condition, Outcome and Strategy

Finally, we explore the relationship between learning outcome and experimental condition for each of the strategic profiles we have identified. Interestingly, we find that experimental condition has a substantial effect on learning outcome among the "stubborn" students, but virtually no effect on learning among the "moderate", "interactive", and "confident" (see Table 5). Most students perform in the second and third tercile when given multiple graphical representations, but are overwhelmingly in the first tercile when given a single representation.

Students in the other three classes are not significantly affected by their representation condition. The learning strategies that these students implement seem to make them resilient to representational choice, at least in this experimental regime. Recall that students exhibiting the "stubborn" profile rarely requested hints, even when they encountered difficulty. We speculate that they lack the metacognitive skills to judge when their learning strategies are failing, and thus are not seeking help at appropriate times [4]. They are the most sensitive to pedagogical decisions because they are the least equipped to structure and manage their own learning.

An ITS ought to ensure that these students are targeted with multiple representations, and perhaps other forms of metacognitive support. While not all "stubborn" students improve when given MGRs, the vast majority of them do. An ITS might help scaffold effective learning behaviors by spontaneously offering hints to these students when they appear to need them the most. A teacher informed that a student exhibits this learning profile may try to encourage the student to ask for help and target their metacognitive skills more generally. Moreover, studying this subpopulation seems to be a promising avenue for illuminating the mechanism by which MGRs improve learning outcomes. Future experiments could test the effect of offering spontaneous hint-support to students that fit the "stubborn" profile.

[2]Student's T-test: t = 0.9978, df = 139.602, p-value = 0.3201
[3]We fail to reject independence at a p-value of .38.
[4]$\mathcal{X}^2 = 1.1517$, df = 3, p-value = 0.7646

moderate	33%	66%	99%
blocked	2	8	3
increased	4	9	8
interleaved	4	9	4
moderate	4	5	9
single	6	4	5

$\mathcal{X}^2 = 8.08$, df $= 8$, p-value $= 0.43$

interactive	33%	66%	99%
blocked	7	6	2
increased	9	5	2
interleaved	5	8	5
moderate	7	2	1
single	5	5	4

$\mathcal{X}^2 = 6.95$, df $= 8$, p-value $= 0.54$

confident	33%	66%	99%
blocked	0	5	5
increased	3	3	4
interleaved	2	2	3
moderate	3	4	5
single	5	1	5

$\mathcal{X}^2 = 7.41$, df $= 8$, p-value $= 0.49$

stubborn	33%	66%	99%
blocked	5	10	10
increased	6	3	6
interleaved	2	2	6
moderate	4	2	7
single	14	3	3

$\mathcal{X}^2 = 17.4837$, df $= 8$, p-value $=$ **0.025**

Table 5: Condition and Tercile of Adjusted Delayed Post-Test Score, by Learning Strategy

We note that there are competing interpretations of our results that also suggest interesting future experiments. Studies have found that well-designed feedback from errors may be very effective for improving learning outcomes [12]. It may be that "stubborn" students, by not shying away from mistakes, are taking advantage of a more effective support system than students who avoid mistakes by soliciting hints. Since instruction with multiple representations is generally more difficult, stubborn students in a multiple representation condition would get more of this kind of feedback on average. This interpretation would predict that students in the "interactive" profile would benefit if some hints were withheld [11]. However, this hypothesis could only be tested by subsequent experiments.

4.6 Profile Stability

If an intelligent tutoring system could implement our classification methodology on-the-fly, it could tailor its pedagogical interventions to the needs of the individual student. To substantiate the promise of the methodology we investigate how efficiently the algorithm stabilizes to the final classification. To measure this, we first cluster on the entire corpus and assign each student to their most likely profile. We then artificially subset the data by restricting the number of problems seen by the clustering algorithm, compute the proportion of students who are in their "final" profile, and then iteratively increase the size of the subset. This simulates how well our algorithm identifies student profiles as they make their way through the material.

Figure 5 shows the percentage of total data used to estimate the model plotted against the proportion of students assigned to their final strategic profile. At each iteration, we look at an additional 10 problems from each student and re-estimate the cluster assignments. The regression estimates that 63% of the data is sufficient to classify three quarters of

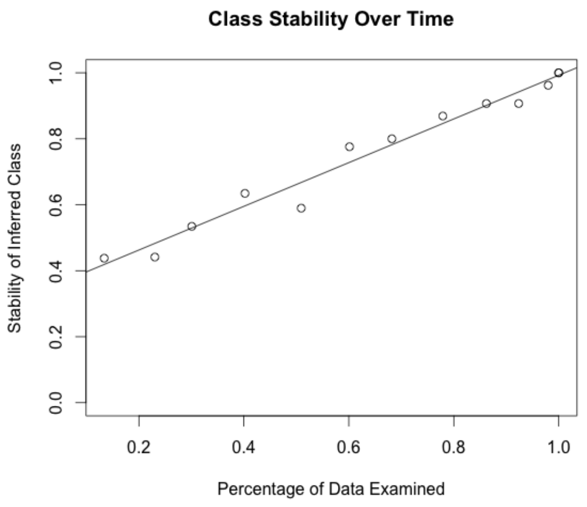


Figure 5: We measure the number of students who were classified into their ultimate strategic profile as the amount of data available to EM is increased. We see that at about 60% of the data we can classify about 75% of the students into their ultimate profile.

the students into their ultimate cluster. Thus, after seeing about 60 problems – about two days of classroom instruction – a dynamic intelligent tutoring system might intervene on students who fit the "stubborn" profile by ensuring that they are presented with multiple graphical representations, offering them spontaneous hints or targeting them with some other form of metacognitive support.

5. CONCLUSION & FUTURE WORK

We estimated an expectation maximization clustering model to classify students into four strategic profiles based on their error-rates and hint-seeking behaviors. We detected an effect of experimental condition on post-test outcome only in the class of students characterized by high-error rate and low hint-seeking propensity. That is, students who did not seem to take full advantage of the resources that the Fractions Tutor offered were the ones most strongly affected by experimental condition. These students may not have the metacognitive skills required to know when to seek hints.

Our methods could be used by ITS designers to detect students with this profile in real time. Tutoring systems could then intervene to target these students with MGRs, scaffold their hint-seeking behaviors or target them with other forms of metacognitive support. Future research into the mediating mechanisms of multiple representations could leverage our results to identify the relevant student sub-populations to investigate. Our post-hoc analysis is not designed to identify the cognitive processes underlying the student's problem solving behavior, so interviews or a cognitive task analysis with students who fit the "stubborn" profile could reveal more details about their experience than we can detect from the log data. Additional investigation into different features may help further characterize student behavior and could help us more accurately group students into relevant subpopulations. Although our analysis seems to have revealed interesting differences in student learning strategies, more informative features constructed from log data may do better. Constructing more informative features, for example, might allow us to separate the "stubborn" students into those who did and did not benefit from multiple graphical representations.

6. REFERENCES

[1] S. E. Ainsworth. The functions of multiple representations. *Computers and Education*, 33(2-3):131–152, 1999.

[2] S. E. Ainsworth. Deft: A conceptual framework for learning with multiple representations. *Learning and Instruction*, 16(3):183–198, 2006.

[3] V. Aleven and K. R. Koedinger. Limitations of student control: Do students know when they need help? In G. Gauthier, C. Frasson, and K. VanLehn, editors, *Proceedings of the 5th International Conference on Intelligent Tutoring Systems, ITS 2000*, pages 292–303, 2000.

[4] V. Aleven, B. McLaren, I. Roll, and K. R. Koedinger. Toward meta-cognitive tutoring: A model of help seeking with a cognitive tutor. *International Journal of Artificial Intelligence in Education*, 2006.

[5] V. Aleven, E. Stahl, S. Schworn, F. Fischer, and R. M. Wallace. Help seeking and help design in interactive learning environments. *Review of Educational Research*, 73(2):277–320, 2003.

[6] S. A. Ambrose, M. W. Bridges, M. DiPietro, M. C. Lovett, and M. K. Norman. *How Learning Works.* Jossey-Bass, 2010.

[7] R. Baker. Differences between intelligent tutor lessons, and the choice to go off-task. In *Proceedings of the 2nd International Conference on Educational Data Mining*, 2009.

[8] R. Baker, A. Corbett, I. Roll, and K. R. Koedinger. Developing a generalizable detector of when students game the system. *User Modeling and User-Adapted Interaction*, 2009.

[9] C. R. Beal, L. Qu, and H. Lee. Mathematics motivation and achievement as predictors of high school students' guessing and help-seeking with instructional software. *Journal of Computer Assisted Learning*, 24:507–514, 2008.

[10] L. M. Collins and S. T. Lanza. *Latent Class and Latent Transition Analysis: With Applications in the Social, Behavioral, and Health Sciences.* Wiley Publishing, 2009.

[11] K. R. Koedinger and V. Aleven. Exploring the assistance dilemma in experiments with cognitive tutors. *Education Psychology Review*, 19:239–264, July 2007.

[12] J. E. McKendree. Effective feedback content for tutoring complex skills. *Human Computer Interaction*, 5:381–414, 1990.

[13] A. Merceron and K. Yacef. Clustering students to help evaluate learning. *Technology Enhanced Learning*, pages 31–42, 2005.

[14] A. Newell and P. Rosenbloom. Mechanisms of skill acquisition and the law of practice. *Cognitive Skills and Their Acquisition*, 1981.

[15] D. Perera, J. Kay, I. Koprinska, K. Yacef, and O. R. Zaiane. Clustering and sequential pattern mining of online collaborative learning data. *Knowledge and Data Engineering, IEEE Transactions*, 21(6):759–772, 2009.

[16] M. Rau and R. Scheines. Searching for variables and models to investigate mediators of learning from multiple representations. In *International Conference on Educational Data Mining*, 2012.

[17] M. A. Rau, V. Aleven, and N. Rummel. Intelligent tutoring systems with multiple representations and self-explanation prompts support learning of fractions. *International Conference of Artificial Intelligence in Education*, pages 441–448, 2009.

[18] M. A. Rau, V. Aleven, Z. Tunc-Pekkan, L. Pacilio, and N. Rummel. How to schedule multiple graphical representations? a classroom experiment with an intelligent tutoring system for fractions. In *Proceedings of International Conference of the Learning Sciences*, 2012.

[19] J. Self. The application of machine learning to student modelling. 14(3–4):327–338, 1986.

[20] C. Unsupervised and S. C. to Build User Models for Exploratory Learning Environments. Saleema amershi and cristina conati. *Journal of Educational Data Mining*, 1, 2009.

Unsupervised Classification of Student Dialogue Acts With Query-Likelihood Clustering

Aysu Ezen-Can
Department of Computer Science
North Carolina State University
Raleigh, North Carolina 27695
aezen@ncsu.edu

Kristy Elizabeth Boyer
Department of Computer Science
North Carolina State University
Raleigh, North Carolina 27695
keboyer@ncsu.edu

ABSTRACT

Dialogue acts model the intent underlying dialogue moves. In natural language tutorial dialogue, student dialogue moves hold important information about knowledge and goals, and are therefore an integral part of providing adaptive tutoring. Automatically classifying these dialogue acts is a challenging task, traditionally addressed with supervised classification techniques requiring substantial manual time and effort. There is growing interest in unsupervised dialogue act classification to address this limitation. This paper presents a novel unsupervised framework, query-likelihood clustering, for classifying student dialogue acts. This framework combines automated natural language processing with clustering and a novel adaptation of an information retrieval technique. Evaluation against manually labeled dialogue acts on a tutorial dialogue corpus in the domain of introductory computer science demonstrates that the proposed technique outperforms existing approaches. The results indicate that this technique holds promise for automatically understanding corpora of tutorial dialogue and for building adaptive dialogue systems.

Keywords

Tutorial dialogue, dialogue act modeling, unsupervised machine learning

1. INTRODUCTION

Tutorial dialogue systems are highly effective at supporting student learning [1, 8, 9, 11, 13, 14, 20]. However, these systems are time-consuming to build because of the substantial engineering effort required within their various components. For example, understanding and responding to the rich variety of student natural language input has been the focus of great attention, addressed by a variety of techniques including latent semantic analysis [15], enriching natural language input with spoken language capabilities [21], linear regression for assessing correlation of dialogue acts with learning [8] and integration of multiple dialogue policies [12]. However, a highly promising approach is to automatically mine models of user utterances from corpora of dialogue using machine learning techniques [16, 24].

A task of particular importance in modeling student utterances is determining the *dialogue act* of each utterance [25, 28]. The premise of dialogue act modeling is that it captures the communicative goal or action underlying each utterance, an idea that emerged within linguistic theory and has been leveraged with great success by dialogue systems researchers [2][27]. Dialogue act modeling, in practice, is based on creating taxonomies to use in dialogue act classification. Within tutorial dialogue systems, first the dialogue act for a student utterance is inferred, and this label serves as the basis for selecting the next tutorial strategy.

There are two approaches for learning dialogue act models from a corpus: supervised and unsupervised. Supervised models require a manually labeled corpus on which to train, while unsupervised models employ machine learning techniques that rely solely on the structure of the data and not on manual labels. A rich literature on supervised modeling of dialogue acts has shown success in this task by leveraging a variety of lexical, prosodic, and structural features [29, 30]. However, supervised models face two significant limitations. First, manual annotation is a time-consuming and expensive process, a problem that is compounded by the fact that many annotation schemes are domain-specific and must be re-engineered for new corpora. Second, although there are standard methods to assess agreement of different human annotators when applying a tagging scheme, developing the tagging scheme in the first place is often an ill-defined process. In contrast, unsupervised approaches do not rely on manual tags, and construct a partitioning of the corpus that suggests a fully data-driven taxonomy. Unsupervised approaches have only just begun to be explored for dialogue act classification, but early results from the computational linguistics literature suggest that they hold promise [10, 24], and a very recent finding in the educational data mining literature has begun to explore these techniques for learning-centered speech [25].

This paper presents a novel approach toward unsupervised dialogue act classification: *query-likelihood clustering*. This approach adapts an information retrieval (IR) technique based on query likelihood to first identify utterances that are similar to a target utterance. These results are then clustered to identify dialogue acts within a corpus in a fully unsupervised fashion. We evaluate the proposed technique on a corpus of task-oriented tutorial dialogue collected through a textual, computer-mediated dialogue study. How best to evaluate unsupervised techniques is an open research question since there is no “perfect” model that the results can be compared to. We therefore examine two complementary evaluation criteria that have been used in prior work: quantitative evaluation with respect to manual labels [10, 25], and detailed qualitative inspection of the clustering to determine whether it learned “natural” groupings of utterances [24]. The results demonstrate that query-likelihood clustering performs significantly better than majority baseline chance compared to manual labels. In addition, the proposed algorithm outperforms a recently reported unsupervised approach for speech act classification within a learning-centered corpus. Finally, qualitative analysis suggests that the clustering does group together many categories of utterances in an intuitive way, even highlighting in a fully data-driven fashion some ways in which the original hand-authored dialogue act taxonomy could be revised and improved in the future.

2. RELATED WORK

Dialogue act classification aims to model the intent underlying each utterance. Supervised dialogue act modeling has been well studied in the computational linguistics literature, applying techniques such as Hidden Markov Models [30] and Maximum Entropy classifiers [4][29]. For tutorial dialogue, promising approaches have included an extension of latent semantic analysis [28], a syntactic parser model [22], and vector-based classifiers [6].

Compared to the rich body of work on supervised dialogue act modeling, a much smaller body of work has focused on unsupervised approaches. A recent non-parametric Bayesian approach used Dirichlet Process Mixture Models [10], which attempt to identify the number of clusters non-parametrically. Another recent work on unsupervised classification of dialogue acts modeled a corpus of Twitter conversations using Hidden Markov Models combined with a topic model built using Latent Dirichlet Allocation [24]. This corpus was composed of small dialogues about many general subjects discussed on Twitter. In order for the dialogue act model not to be distracted by different topics, they separated content words from dialogue act cues with the help of the topic model. In our tutoring corpus, however, the content words reveal important information about dialogue acts. For example, the word "help" is generally found in utterances that are requesting a hint. Therefore, our model retains content words.

Rus et al. utilize clustering to classify dialogue acts within an educational corpus [25], forming vectors of utterances using the leading tokens (words and punctuation marks), and using string comparison as the similarity metric. As they mention, this string comparison may not be sufficient to generalize word types used within the same context. For example, 'hello' and 'hi' are different according to string comparison; however, they are part of the same dialogue act, in that they both serve as a greeting. Our clustering approach uses query likelihood to group similar words that can be used for the same intention, and we use a blended part-of-speech tag and word feature set which overcomes the challenge introduced by string comparisons. The results suggest that these extensions improve upon existing clustering techniques.

3. TUTORING CORPUS

The corpus consists of dialogues collected between pairs of tutors and students collaborating on the task of solving a programming problem as part of the JavaTutor project during spring 2007. The tutor and student interacted remotely with textual dialogue through computer interfaces. There were forty-three dialogues in total, with 1,525 student utterances (averaging 7.54 words per utterance) and 3,332 tutor utterances (averaging 9.04 words per utterance). This paper focuses on classifying the dialogue acts of student utterances only. Within an automated tutoring system, tutor utterances are system-generated and their dialogue acts are therefore known. The corpus was manually segmented and annotated with dialogue acts, one dialogue act per utterance, during prior research that focused on supervised dialogue act annotation and dialogue structure modeling [7]. While the manual dialogue act labels are not used in model training, they are used to evaluate the unsupervised clustering. Table 1 shows manually labeled tags and their frequencies. The Kappa for agreement on these manual tags was 0.76. An excerpt from the corpus is presented in Table 2.

Table 1: Dialogue act tags with examples and student frequencies from corpus

Tag	Act	Description	Freq
Q	Question	*A general question which is not specific to the task*	276
EQ	Evaluation Question	*A question about the task*	416
S	Statement	*A statement of fact*	211
G	Grounding	*Acknowledgement of previous utterance*	192
EX	Extra-Domain	*Any utterance that is not related to the task*	133
PF	Positive Feedback	*Positive assessment of knowledge or task*	116
NF	Negative Feedback	*Negative assessment of knowledge or task*	92
LF	Lukewarm Feedback	*Assessment having both positive and negative assessments*	32
GRE	Greeting	*Greeting words*	57

Table 2: Excerpt from the corpus (typographical errors originated in corpus)

	Utterance	Tag
Student:	so obviously here im going to read into the array list and pull what we have in the list so i can do my calculations	*S*
Tutor:	something like that, yes	*LF*
Tutor:	by the way, an array list (or ArrayList) is something different in Java. this is just an array.	*S*
Student:	ok	*G*
Student:	im sorry i just refer to it as a list because thats what it reminds me it does	*S*
Student:	stores values inside a listbox(invisible)	*S*
Tutor:	that's fine	*EX*
Tutor:	ok, so what are we doing here?	*EQ*
Student:	im not sure how to read into the array	*NF*

4. QUERY-LIKELIHOOD CLUSTERING

This section describes our novel approach of adapting information retrieval (IR) techniques combined with clustering to the task of unsupervised dialogue act classification. IR is the process of searching available resources to retrieve results that are similar to the query [3]. IR techniques are mostly used in search engines to retrieve results that are similar to given queries. In the proposed approach, the target utterance that is to be classified is used as a query and its similar utterances are gathered using query likelihood. Then, the query likelihood results are provided to the clustering technique.

4.1 Natural Language Preprocessing

At its core, query-likelihood information retrieval operates at the token, or word, level. In order to prepare the corpus for this application, several preliminary experiments were conducted to determine the appropriate type of preprocessing. It was observed that preprocessing is a crucial step in order to increase the discriminating cues extracted from the corpus.

Part-of-speech (POS) tagging is a technique for labeling each word according to its grammatical part of speech such as noun, verb, and adjective. This procedure allows us to generalize words to their functionalities in sentences. For example, the pronouns 'you' and 'it' are grouped in the same POS tag: PRP standing for personal pronoun. The generalization provided by this part-of-speech backoff can be useful in dialogue act classification [5, 6, 28]. We experimented on querying with both actual words and with full part-of-speech backoff. The best results were produced by a combination of words and POS tags. This hybrid approach replaces function words such as determiners ('the', 'a'), conjunctions ('and', 'but'), and prepositions ('in', 'after') with their POS tags. Content words were retained but stemmed (e.g., 'parameter' becomes 'paramet', 'completely' becomes 'complet') to reduce the number of distinct words in the vocabulary of the corpus under consideration. This choice was motivated by the observation that in this task-oriented domain, important information about the dialogue act resides in content words. For instance, the word 'confused' reveals important information about the state in which the student is and it is likely that the student might be requesting a hint.

It was noted that in this domain of computer science tutoring, the natural language contains special characters that indicate a semantically important entity related to the domain, such as short bits of programming code. Although they are important with regard to the tutoring task, they require additional preprocessing in order to be handled appropriately by automated natural language processing techniques. Therefore, code segments in the corpus were replaced with meaningful tags representing them. For instance, segments about array indexing, which may originally have appeared as 'x[i]' and been mishandled, were replaced with the text 'ARRAY_INDEXING'. If-statements, loops and arithmetic operations were all replaced in the corpus using similar conventions. All procedures in natural language processing is automated using parser therefore, the human-intervention was in deciding on the procedure to use (retain content words, replace function words) not in its implementation.

4.2 Query-Likelihood Representation

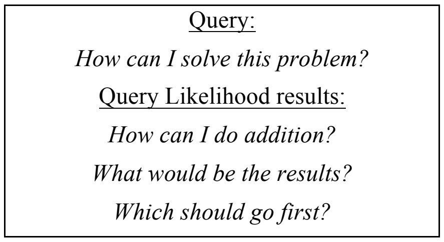
The query likelihood model treats each utterance as a document. The target utterance whose dialogue act is to be predicted becomes the query, and we apply information retrieval by querying the target utterance in the corpus. This query produces a ranked list of documents, from most likely to least likely, and this list is used to identify those utterances that are most similar to the target. The query likelihood implementation from the Lemur Project was used in this work [31].

The ordering of words contains important information about the structure of utterances. For example, the word pair 'I am' is mostly found in statements whereas if we regard them separately, 'I' can belong to a question such as 'What should I do next?' or 'am' can be part of a evaluating question 'Am I doing this correct?' (After preprocessing 'I am' becomes 'PRP (personal pronoun) VBP (present tense singular verb, non third person)' however, the same issue applies to the POS tags as well.) Because of the importance of word ordering on inferring the structure of utterances, we utilized a modified query approach that considers bigrams (pairs of adjacent words occurring in each utterance) rather than unigrams (individual words).

Table 3 displays several original utterances, their modified forms after POS backoff and stemming, and the query combinations that were submitted to the algorithm. The POS tag VBZ represents third person present tense singular verbs, DT represents determiners, TO is the same as the word 'to', VBD stands for past tense verbs, WDT and WRB are interrogatives, and MD represents modal verbs. Figure 1 shows an example query with a question and its similar utterances retrieved.

Table 3: Original utterances, their processed versions and query combinations

Utterance	POS+ stemming	Query combination
I'm reading it right now	VB read PRP right now	(VB read) (read PRP) (PRP right) (right now)
what is the basic structure to begin an array?	WDT VBZ DT basic structur TO begin DT array?	(WDT VBZ) (VBZ DT) (DT basic) (basic structur) (structur TO) (TO begin) (begin DT) (DT array) (array ?)
that was correct	WDT VBD correct	(WDT VBD) (VBD correct)
how do you think you should start it?	WRB do PRP think PRP MD start PRP?	(WRB do) (do PRP) (PRP think) (think PRP) (PRP MD) (PRP start) (start PRP) (PRP ?)


Query:

How can I solve this problem?

Query Likelihood results:

How can I do addition?

What would be the results?

Which should go first?

Figure 1: Sample query and its results

4.3 Clustering

The similarity results from querying were used as the distance metric in a *k*-means clustering algorithm. The implementation of this idea relies on creating binary vectors for similar utterances and then grouping those vectors. Each utterance that is present in the similarity list is represented as a 1, while the others are represented with a value of 0. In this way, each target utterance in the corpus is represented by a vector indicating the utterances that are similar to it. The entire unsupervised dialogue act classification algorithm is summarized in Figure 2.

```
Let $D$ be a corpus of utterances $D = \{u_1, u_2 \dots u_n\}$
Then the goal is:
$\forall u_i \in D$, identify $l_i$ = dialogue act label of $u_i$
Procedure:
    For each $u_i$
        1. Set target utterance $q_i = u_i$
        2. Let the query likelihood result of
              utterance $u_i$ be $R = (u_t, u_{t+1} \dots u_z)$
              such that $R$ is the result of
              $queryLikelihood(q_i)$
        3. Create vector of query results
              indicator variables
              $V_i = (v_1, v_2 \dots v_j \dots v_n)$ such that
                  $v_j = 1$ if $u_i \in R$
                  else $v_j = 0$
    Let the total vector be $V_T = (V_1, V_2, \dots, V_n)$
    Return clusters $C = \{c_1, c_2 \dots c_k\}$
              such that $C$ is the result of
              $k$-$means(V_T)$
```

Figure 2: Query-likelihood clustering algorithm.

5. EXPERIMENTS

The goal of the experiments is to apply the novel unsupervised technique of query-likelihood clustering to discover student dialogue act clusters within the corpus of tutorial dialogue. We utilize a two-pronged evaluation consisting of quantitative comparison in terms of accuracy on manual labels, as well as qualitative examination of the resulting clusters. This section first presents the model-learning process, including parameter tuning on a development set, and then presents quantitative and qualitative evaluations on the remainder of the corpus. We also compare performance of the proposed approach to a state-of-the-art unsupervised technique for speech act labeling in a learning-centered corpus.

5.1 Parameter Tuning

In order to train the unsupervised model, three parameters had to be set. Two of these parameters are used within the query phase and the last one applies to the clustering phase. The two parameters to be determined in the query phase are *b*, the blind relevance feedback threshold, and *n,* the number of top query likelihood results to be used while creating vectors for clustering. The parameter related to the clustering phase is the distance metric. In order to tune these parameters, a 25% validation set, constituting 10 randomly selected sessions, was drawn from the corpus. The parameter tuning was conducted in a sequential manner that allows the latter steps to use already fixed parameters.

Token weighting. Prior to tuning the parameters, an additional optional set of token weights was tuned for use during querying. This optional parameter was desirable based on observations that some POS tags should be weighted more than others for identifying similar results to a target utterance. For example, interrogatives (question words such as *what*, *where*, *when*, *how*, and *why*) and question marks are highly discriminating for question dialogue moves. Weights were learned for this subset of tokens using an incremental approach as shown in Table 4. First, the mean average precision values of query likelihood results without any weights were given. Then, weights for WDT (POS tag for *what* and *which*) were utilized within the experiments. Having determined the proper weight for WDT, different weights for WRB (POS tag for *why, where, how, when*) were tried. Finally, the weight for question marks was set.

Table 4: Mean average precision (MAP) results for weighting interrogative parts of speech and punctuation

Weights	MAP
no weight	0.1239
WDT = 10	**0.2359**
WDT = 100	0.2326
WRB = 10	**0.2358**
WRB = 100	0.2339
'?' = 10	0.2457
'?' = 100	**0.2567**

Relevance feedback threshold (*b*). In a typical query likelihood scenario, relevance feedback on the retrieved results is provided by human users and used to improve the model performance. However, in a fully unsupervised scenario, human relevance feedback is not available. Our unsupervised approach utilizes pseudo-relevance feedback (blind relevance feedback), which assumes that the top *b* documents retrieved are relevant to the query [26]. Taking the top *b* documents into account, the algorithm automatically finds words that are important for those documents and therefore may be useful for the query. In order to find the important words, relevance models for each retrieved document in the ranked list are computed, where a relevance model is the probability of features used in the query given the document. In our experiments, the features are composed of bigrams, which are pairs of adjacent words within an utterance. Therefore, the relevance model of a document is the probability of its bigrams given the whole utterance. Then, relevance models of the top *b* documents are sorted and the top terms are determined, which are used to expand the original query. Having chosen those words, the algorithm expands the initial query by appending the newly found terms and running the query again. This procedure of enriching the query with top relevant results is done in order to avoid sparse ranked lists. Like the other parameters described in this section, *b* was tuned on a development set. Table 5 shows the resulting best *b*-value of 30.

Table 5: *b*-value MAP results

b	**5**	**10**	**15**	**20**	**25**	**30**	**35**
MAP	0.343	0.34	0.342	0.346	0.351	**0.352**	0.351

Top query threshold (*n*). Given the *b*-value, we moved on to tuning *n*-values, which represent the number of top query likelihood results to be used in forming the vectors for subsequent clustering. A higher *n* value will treat a larger number of utterances retrieved during querying as "similar" to the target. A minimum of *n*=5 was chosen to sufficiently populate the vectors since the non-zero entries determine clusters, and we explored whether larger *n* values performed better; however, the optimal value was *n*=5 as shown in Table 6.

Table 6: *n*-value MAP results with set *b*-value

n-value	**5**	**10**	**15**	**20**	**30**	**100**
MAP	**0.517**	0.432	0.409	0.398	0.395	0.307

Distance metric. We experimented with multiple distance metrics to determine which metric performed best within this novel context. The candidate distance metrics included the widely used Euclidean and Manhattan distances. Also considered was another widely used similarity metric in text mining, cosine similarity, which increases as the two vectors have non-zero entries in the same positions. This approach is particularly effective for sparse vectors. The final candidate was Borda count [18], a metric that weights non-zero entries according to their ranks in query likelihood. In our scenario, the first utterance retrieved by query likelihood gets the highest rank. This approach reforms vectors in a weighted manner, after which Euclidean distance is applied.

Within the development set, cosine distance performed with highest accuracy at 43.43% compared to manual labels. Borda count achieved 42.63%, Euclidean distance 41.64%, and Manhattan distance 41.82%. Since cosine similarity is computed by the dot product of two vectors divided by the product of their magnitudes, it takes the size of vectors into account and provides a ratio of matching 1 values in the same position in both vectors. In this way, intersecting non-zero entries are valued with respect to the size of the vectors.

5.2 Accuracy in Identifying Manually Labeled Dialogue Acts

Having tuned the parameter values using the development set, we trained a model on the remainder of the corpus. This unsupervised model did not take manual labels into account during training, but we evaluate its performance with respect to manual labels. Due to the nature of unsupervised approach, the optimal number of clusters was not known. Therefore, we explore varying numbers of clusters using *k*-Means, a standard clustering algorithm that aims to cluster observations into *k* clusters so that each element has the highest similarity to every other element in the cluster [19]. In addition to *k*-means clustering, we experimented with a non-parametric Bayesian approach used in recent unsupervised dialogue act classification work in a non-tutoring domain [10].

We calculate the accuracy of the approach for classification by comparing to manual labels. Figure 3 presents the accuracy of the proposed approach using the *k*-means clustering algorithm in Weka [17]. To label a target utterance, the query likelihood results were retrieved for that utterance, and then its vector was provided to the clustering algorithm. The resulting cluster in which the target utterance resides was interpreted as the dialogue act label of the target utterance. The majority label of the cluster was taken as its dialogue act label. For comparison, the accuracy results are compared against the majority class baseline of 25.87%, Evaluation Question (EQ). This is the most commonly occurring student dialogue act in the corpus and therefore represents the expected accuracy of a model that performs equal to chance. Additionally, we compare our implementation against that of the recent approach of Rus et al. [25], which clusters utterances via word similarity using a specified number of leading tokens of each utterance.

The highest accuracy achieved by query-likelihood clustering was 41.84% with *k*=8. This accuracy therefore constitutes a 61.9% improvement over baseline chance. We experimented with the Rus et al. leading-tokens clustering using two to five leading tokens as they suggest. Five leading tokens performed best, yielding 34.90% accuracy at *k*=7.

In order to provide a comparison across corpora, we consider the results from Rus et al. on their corpora of educational games. Their highest accuracy was 37.9%, compared with a majority class baseline of 28.5% (statements). This is a 32.98% improvement over the majority baseline chance. Another algorithm tried by Rus et al. was Expectation Maximization, which achieved 37.9% accuracy on their combined corpus of educational games. This algorithm achieved its highest accuracy of 30.47% with four leading tokens on our corpus. We also experimented with Dirichlet process clustering as used by Crook et al. [10]. Dirichlet process clustering gave substantially lower results on our corpus after natural language preprocessing, with an accuracy of 24.48% compared to the 41.84% of query-likelihood clustering.[1]

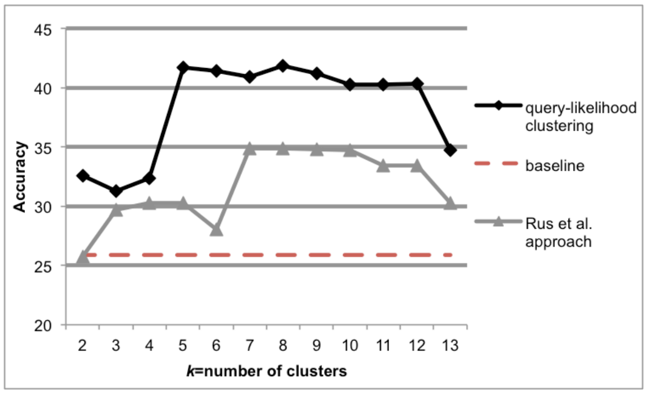


Figure 3: Accuracy results (%) for query-likelihood clustering, Rus et al. approach with 5 leading tokens and majority baseline.

Finally, in order to evaluate the relative contribution of the query-likelihood clustering components of automatic natural language preprocessing (POS tagging, stemming) and vector enhancement with information retrieval, we performed two experiments omitting each of these components from the procedure. Parameters were re-tuned for each experiment utilizing the same development set split used earlier (25% of the corpus). With the best-performing model size of *k*=8 clusters, omitting the natural language preprocessing step produced accuracy of 37.59% and omitting the information retrieval step produced accuracy of 35.68%, each of which is substantially lower than the query-likelihood clustering approach of 41.84%. However, these simpler approaches resulted in a smaller number of clusters for their best-fit models, achieving their highest accuracies of 41.06% and 40.45%, respectively, when *k*=4. This accuracy is only modestly lower than the query-likelihood clustering accuracy of 41.84%; however, a significant drawback is that the number of clusters is much smaller than are typically distinguished by dialogue act classification schemes, and would likely not result in sufficiently fine-grained distinctions to support dialogue management.

5.3 Analysis of Clusters

As shown in Table 7, the approach was particularly successful in clustering negative feedback utterances (NF), evaluation questions (EQ) and groundings (G). 64.06% of all NF utterances are grouped in one cluster (*Cluster 2*), while 64.09% of EQ utterances are in one cluster *(Cluster 5)* and 94.74% of all G utterances are in another cluster *(Cluster 1)*. The Q and EQ tagged utterances,

[1] The Dirichlet clustering implementation from mahout.apache.org was used for this analysis.

which are structurally very similar in that they both are questions, were grouped into one cluster (*Cluster 5)*, which constitutes 58.54% of all Q and EQ tagged utterances. In another cluster, 14.26% of all Q and EQ utterances were grouped together *(Cluster 6)*.

Table 7: Student utterance distributions over clusters using manual tags (majority dialogue act in bold for each cluster)

Clusters/ Tags	*C1*	*C2*	*C3*	*C4*	*C5*	*C6*	*C7*	*C8*
GRE	31	1	0	4	8	0	0	2
EX	22	21	5	16	8	0	1	21
Q	1	13	11	20	121	33	0	36
PF	57	16	6	7	0	1	0	5
S	11	32	**21**	**29**	5	0	0	**51**
G	**144**	0	0	1	0	0	**6**	1
EQ	1	18	17	14	**191**	**43**	0	14
NF	3	**41**	9	4	0	0	0	7
LF	1	15	2	3	0	0	0	1

In order to examine the structure of the clusters in more detail, Table 8 provides sample utterances from four of the eight clusters. The manual labels of the utterances are also given in parenthesis. Some clusters perform intuitively; for example, *Cluster 1* is dominated by grounding, with many positive feedback utterances as well. These positive feedback and grounding utterances have similar surface forms such as "yeah," "right," and "oh," and distinguishing them further will require modeling the broader dialogue structure (for example, a notion of dialogue history) in future work.

Cluster 2 is dominated by negative feedback (NF) with a large number of statements (S). This cluster captures the vast majority of negative feedback moves, indicating that these moves are highly distinguishable. Most of the statements were in negative tone or expressing confusion. For example, the utterance "Im not sure if it is asking if the PARAMETER is how ever far you are away from NUMBER or the actual number you are away from NUMBER," which is labeled as statement, shows confusion although it is a statement in terms of its intention.

Cluster 3 and *Cluster 4* are highly impure. Closer inspection reveals that *Cluster 3* mostly has broken utterances such as "correction digit", "is in", "digit" and some statements. *Cluster 4* contains statements and extra domain utterances that are primarily statements. Moreover, there are some implicit questions such as "thats another thing I was going to ask am I just storing the values in METHOD_NAME and sending them to PARAMETER" and "So I have to call upon the PARAMETER class and use a method in there right?". *Cluster 7* is highly sparse, containing only seven utterances. These tend to be highly similar in structure, such as "oh ok," and "oh dear." The very low distance between these utterances pulled them tightly into one cluster that was judged as distinct from the others.

Cluster 5 and *Cluster 6* are dominated by evaluation questions (EQ), with a substantial number of general questions (Q) as well. The dominance of EQ in both clusters may relate to its high frequency within the corpus. However, these two question acts tend to exhibit close structural similarity although their roles are different (asking for feedback versus asking a general conceptual question). For example, the utterance 'do i have to set it to a PARAMETER?' is a question; however, a deeper analysis shows that it is more specifically an evaluation question that requests feedback on the task.

Table 8: Utterances from selected clusters with manual tags

Cluster 1

- right (G)
- ahh (G)
- ok (G)
- yeah (G)
- yes (PF)
- heheh yeah that would work (PF)
- I see that (PF)
- gotcha (EX)
- Yes Giving me definitions to various commands and such (EX)
- Ohh yes substantially (EX)

Cluster 2

- not really yet (NF)
- im not completely sure about how to do this (NF)
- the parsing im not sure about (NF)
- to be honest im not even sure what an array is (NF)
- im not sure how to read into the array (NF)
- I don't know how to do this (NF)
- and I know there is more to this line but I cannot remember the command (NF)
- Not yet (NF)
- im not so good at ARRAY just yet (NF)
- but I'm not exactly sure how to do that (NF)
- Im not sure if it is asking if the PARAMETER is how ever far you are away from NUMBER or the actual number you are away from NUMBER (S)
- I am asking how to do whatever drawing I need to do in the METHOD_CALL method (S)

Cluster 5

- So what's wrong with this? (Q)
- Can't manually turn an integer into a string? (Q)
- Then how would I incorporate that with the METHOD_CALL? I think it's asking me to use that in some why but it's not supplying arguments to do so (Q)
- are we done extracting digits? (Q)
- how do i sum the digits? (Q)
- do i have to set it to a PARAMETER? (EQ)
- thats another thing I was going to ask am I just storing the values in METHOD_NAME and sending them to PARAMETER? (EQ)
- why is what I just highlighted underlined in red doesn't that mean its wrong? (EQ)
- does extracting have to do with METHOD_NAME? or anything (EQ)

Cluster 6

- What is the next step? (Q)
- What do I write in it? (Q)
- what do i do first? (Q)
- so what can i do to fix what i was doing? (Q)
- does that look ok? (EQ)
- is this correct? (EQ)
- is this what i need to do? (EQ)

The presence of impure and sparse clusters prompted an experiment to explore whether other models with similar but slightly lower overall accuracy would yield a more desirable clustering. Therefore, we explored using an information criterion, Hartigan's rule of thumb, that utilizes the number of parameters in the model. This metric identified ten clusters as optimal, with a slightly lower accuracy (40.28%) compared to eight clusters, and the sparse clusters remained. We also experimented with the X-Means algorithm that utilizes Bayesian Information Criterion for splitting clusters [23], which resulted in four clusters with 36.72% accuracy.

6. DISCUSSION

A strength of unsupervised approaches is that because they do not rely on any manually engineered tagging schemes, they reflect the structure of the corpus in a fully data-driven way. In our case, the results highlight challenges of utilizing pedagogically driven manual dialogue act classification taxonomies within automated approaches. For example, a cross-cutting issue with the clustering presented here is that EX dialogue acts are distributed almost equally across several clusters. In the manual tagging, EX is a catch-all tag for conversation that was not directly related to tutoring. This tag was applied at the structural level, so if a question such as 'Should I close the door?' was not task-related it would have been tagged EX, as would its answer, 'Yes.' This distinction was desirable from a pedagogical perspective, but from a linguistic perspective it conflates dialogue act with topic. Future work will explore combining unsupervised dialogue act modeling with unsupervised topic modeling in order to address this type of modeling challenge. From a dialogue act research perspective, it is important to consider the issue of conflating act with topic when devising manual tagging schemes that may become the target of automated approaches in later work.

While the proposed algorithm is promising in that it outperforms current unsupervised approaches for dialogue act modeling, it has several notable limitations. One limitation is algorithmic complexity, which is quadratic over the size of the corpus. This complexity is inherent in the binary representation of each utterance as a vector with similarity to other utterances. Another limitation of the proposed approach arises with clustering algorithms in general, which is that a significant amount of human intelligence is often required to decide on the number of suitable clusters for the corpus. Nonparametric approaches to automatically identifying the number of clusters performed worse than parametric approaches in the current analyses; however, nonparametric approaches in general are an important area for future study. Finally, the query-likelihood clustering approach does not consider higher-level dialogue structure; it clusters one utterance at a time. This limitation leads to trouble disambiguating utterances with similar surface features. A highly promising direction to address this limitation is to enhance the algorithm with structural features such as dialogue history.

7. CONCLUSION AND FUTURE WORK

We have presented a novel student dialogue act classification model, query-likelihood clustering, which classifies dialogue acts in an unsupervised fashion by adapting techniques from information retrieval with a clustering approach. Although the technique did not utilize manual labels for model training, it performed substantially better than baseline chance at classifying utterances when compared to manually applied dialogue act tags. Moreover, query-likelihood clustering outperformed several currently reported approaches in the recent computational linguistics and educational data mining literature. It discovered a best-fit model with eight clusters, a close correspondence to the nine dialogue acts based on the handcrafted dialogue act taxonomy.

This novel approach holds great promise for dialogue act classification. Several directions are particularly important for future work. Multimodal features, such as posture, gesture, or facial expression of students, may hold great potential for providing dialogue act cues. Additionally, future work should focus on modeling higher-level dialogue structure such as adjacency pairs and discourse structure within unsupervised frameworks, and on devising suitable novel techniques for joint dialogue act and topic modeling within task-oriented tutorial dialogue. Finally, evaluating unsupervised techniques is an open research question. Future work will focus on further developing research techniques for evaluating unsupervised dialogue act classification. Together these research directions will allow unsupervised classification of dialogue acts within large corpora.

8. ACKNOWLEDGMENTS

This work is supported in part by the North Carolina State University Department of Computer Science and the National Science Foundation through Grant DRL-1007962 and the STARS Alliance, CNS-1042468. Any opinions, findings, conclusions, or recommendations expressed in this report are those of the participants, and do not necessarily represent the official views, opinions, or policy of the National Science Foundation.

9. REFERENCES

[1] Aleven, V. and Koedinger, K.R. (2002). An effective metacognitive strategy: learning by doing and explaining with a computer-based Cognitive Tutor. *Cognitive Science*. 26, 2, 147–179.

[2] Austin, J.L. (1962). *How To Do Things With Words*. Oxford University Press.

[3] Baeza-Yates, R. and Ribeiro-Neto, B. (1999). *Modern Information Retrieval*. ACM Press.

[4] Bangalore, S., Fabbrizio, G. Di and Stent, A. (2008). Learning the Structure of Task-Driven Human-Human Dialogs. *IEEE Transactions on Audio, Speech and Language Processing*. 16, 7, 1249–1259.

[5] Becker, L., Basu, S. and Vanderwende, L. (2012). Mind the Gap : Learning to Choose Gaps for Question Generation. *North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 742–751.

[6] Boyer, K.E., Ha, E.Y., Phillips, R., Wallis, M.D., Vouk, M.A. and Lester, J.C. (2010). Dialogue Act Modeling in a Complex Task-Oriented Domain. *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue-SIGDIAL'10*, 297–305.

[7] Boyer, K.E., Phillips, R., Ingram, A., Ha, E.Y., Wallis, M., Vouk, M. and Lester, J. (2011). Investigating the Relationship Between Dialogue Structure and Tutoring Effectiveness: A Hidden Markov Modeling Approach. *The International Journal of Artificial Intelligence in Education (IJAIED)*, 65–81.

[8] Chen, L., Eugenio, B. Di, Fossati, D., Ohlsson, S. and Cosejo, D. (2011). Exploring Effective Dialogue Act Sequences in One-on-one Computer Science Tutoring Dialogues. *Proceedings of the 6th Workshop on*

Innovative Use of NLP for Building Educational Applications., 65–75.

[9] Chi, M., VanLehn, K. and Litman, D. (2010). Do micro-level tutorial decisions matter: applying reinforcement learning to induce pedagogical tutorial tactics. *International Conference on Intelligent Tutoring Systems*, 224–234.

[10] Crook, N., Granell, R. and Pulman, S. (2009). Unsupervised classification of dialogue acts using a Dirichlet process mixture model. *Proceedings of the 10th Annual Meeting of the Special Interest Group on Discourse and Dialogue - SIGDIAL'09*, 341–348.

[11] Dzikovska, M., Steinhauser, N., Moore, J.D., Campbell, G.E., Harrison, K.M. and Taylor, L.S. (2010). Content, Social, and Metacognitive Statements: An Empirical Study Comparing Human-human and Human-computer Tutorial Dialogue. *Proceedings of the European Conference on Technology Enhanced Learning*, 93–108.

[12] Dzikovska, M., Moore, J.D., Steinhauser, N., Campbell, G., Farrow, E. and Callaway, C.B. (2010). BEETLE II : a system for tutoring and computational linguistics experimentation. *Proceedings of the ACL 2010 System Demonstrations*, 13–18.

[13] D'Mello, S.K., Lehman, B. and Graesser, A.C. (2011). A Motivationally Supportive Affect-Sensitive AutoTutor. *R. A. Calvo and S. K. D'Mello (Eds.), New Perspectives on Affect and Learning Technologies*, pp. 113–126.

[14] Forbes-Riley, K. and Litman, D. (2009). Adapting to Student Uncertainty Improves Tutoring Dialogues. *Proceedings of the International Conference on Artificial Intelligence in Education*, 33–40.

[15] Graesser, A.C., Wiemer-Hastings, K., Wiemer-Hastings, P. and Kreuz, R. (1999). AutoTutor: A simulation of a human tutor. *Cognitive Systems Research*. 1, 1, 35–51.

[16] Ha, E.Y., Grafsgaard, J.F., Mitchell, C.M., Boyer, K.E. and Lester, J.C. (2012). Combining Verbal and Nonverbal Features to Overcome the 'Information Gap' in Task-Oriented Dialogue. *Proceedings of the 13th Annual SIGDIAL Meeting on Discourse and Dialogue, Seoul, Republic of Korea*, 247–256.

[17] Hall, M., National, H., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P. and Witten, I.H. (2009). The WEKA Data Mining Software: An Update. *ACM SIGKDD Explorations Newsletter*. 11, 1, 10–18.

[18] Jain, A., Nandakumar, K. and Ross, A. (2005). Score normalization in multimodal biometric systems. *Pattern Recognition*. 38, 12, 2270–2285.

[19] Kanungo, T., Member, S., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R. and Wu, A.Y. (2002). An Efficient *k*-Means Clustering Algorithm: Analysis and Implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 24, 7, 881–892.

[20] Kumar, R., Ai, H., Beuth, J. and Rosé, C.P. (2010). Socially Capable Conversational Tutors Can Be Effective in Collaborative Learning Situations. *Proceedings of the International Conference on Intelligent Tutoring Systems*, 156–164.

[21] Litman, D. and Silliman, S. (2004). ITSPOKE: An Intelligent Tutoring Spoken Dialogue System. *Demonstration Papers at NAACL HLT '10 Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

[22] Marineau, J., Wiemer-Hastings, P., Harter, D., Olde, B., Chipman, P., Karnavat, A., Pomeroy, V., Rajan, S. and Graesser, A. (2000). Classification of Speech Acts in Tutorial Dialog. *Proceedings of the Workshop on Modeling Human Teaching Tactics and Strategies at the Intelligent Tutoring Systems Conference*, 65–71.

[23] Pelleg, D. and Moore, A. (2000). X-means: Extending K-means with Efficient Estimation of the Number of Clusters. *Proceedings of the Seventeenth International Conference on Machine Learning*, 727-734.

[24] Ritter, A., Cherry, C. and Dolan, B. (2010). Unsupervised Modeling of Twitter Conversations. *Proceedings of NAACL HLT '10 Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 172–180.

[25] Rus, V., Moldovan, C., Niraula, N. and Graesser, A.C. (2012). Automated Discovery of Speech Act Categories in Educational Games. *Proceedings of International Conference on Educational Data Mining, 25-32*,

[26] Salton, G. and Buckley, C. (1990). Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*. 41, 4, 288–297.

[27] Searle, J.R. (1969). *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press.

[28] Serafin, R. and Di Eugenio, B. (2010). Dialogue Act Classification, Higher Order Dialogue Structure, and Instance-Based Learning. *Journal of Dialogue & Discourse. 1, 2, 81-104*.

[29] Sridhar, V.K.R., Bangalore, S. and Narayanan, S.S. (2009). Combining lexical, syntactic and prosodic cues for improved online dialog act tagging. *Computer Speech & Language*. 23, 4, 407–422.

[30] Stolcke, A., Ries, K., Coccaro, N., Shriberg, E., Bates, R., Jurafsky, D., Taylor, P., Martin, R., Ess-Dykema, C. Van and Meteer, M. (2000). Dialogue Act Modeling for Automatic Tagging and Recognition of Conversational Speech. *Association for Computational Linguistics*. 26, 3, 339–373.

[31] Strohman, T., Metzler, D., Turtle, H. and Croft, W.B. (2005). Indri : A language-model based search engine for complex queries. *Proceedings of the International Conference on Intelligent Analysis*.

A Spectral Learning Approach to Knowledge Tracing

Mohammad H. Falakmasir
Intelligent Systems Program, University of Pittsburgh
210 South Bouquet Street, Pittsburgh, PA,
falakmasir@cs.pitt.edu

Zachary A. Pardos
Computer Science AI Lab
Massachusetts Institute of Technology
77 Massachusetts Ave.
Cambridge, MA 02139
zp@csail.mit.edu

Geoffrey J. Gordon
Machine Learning Department
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
ggordon@cs.cmu.edu

Peter Brusilovsky
School of Information Sciences,
University of Pittsburgh,
135 North Bellefield Ave.,
Pittsburgh, PA 15260, USA
peterb@pitt.edu

ABSTRACT
Bayesian Knowledge Tracing (BKT) is a common way of determining student knowledge of skills in adaptive educational systems and cognitive tutors. The basic BKT is a Hidden Markov Model (HMM) that models student knowledge based on five parameters: prior, learn rate, forget, guess, and slip. Expectation Maximization (EM) is often used to learn these parameters from training data. However, EM is a time-consuming process, and is prone to converging to erroneous, implausible local optima depending on the initial values of the BKT parameters. In this paper we address these two problems by using spectral learning to learn a Predictive State Representation (PSR) that represents the BKT HMM. We then use a heuristic to extract the BKT parameters from the learned PSR using basic matrix operations. The spectral learning method is based on an approximate factorization of the estimated covariance of windows from students' sequences of correct and incorrect responses; it is fast, local-optimum-free, and statistically consistent. In the past few years, spectral techniques have been used on real-world problems involving latent variables in dynamical systems, computer vision, and natural language processing. Our results suggest that the parameters learned by the spectral algorithm can replace the parameters learned by EM; the results of our study show that the spectral algorithm can improve knowledge tracing parameter-fitting time significantly while maintaining the same prediction accuracy, or help to improve accuracy while still keeping parameter-fitting time equivalent to EM.

Keywords
Bayesian Knowledge Tracing, Spectral Learning.

1. INTRODUCTION
Hidden Markov Models and extensions have been one of the most popular techniques for modeling complex patterns of behavior, especially patterns that extend over time. In the case of BKT, the model estimates the probability of a student knowing a particular skill (latent variable) based on the student's past history of incorrect and correct attempts at that skill. This probability is the key value used by many cognitive tutors to determine when the student has reached mastery in a skill (also called a Knowledge Component, or KC) [17]. In an adaptive educational system, this probability can be used to recommend personalized learning activities based on the detailed representation of student knowledge in different topics.

In practice, there is a two-step process for inferring student knowledge. In the first step, an HMM is learned for each topic or skill within a tutoring system based on the history of students' interaction with the system. The output of this step is a set of parameters (basic parameters of BKT: prior, learn rate, forget, guess, and slip), which is used in the second step to estimate the mastery level of each student. A popular method for the first step, learning parameters from training data, is Expectation Maximization (EM). However, EM is a time consuming process, and previous studies [2,3,11,14] have shown that it can converge to erroneous learned parameters, depending on their initial values. To address these problems, we propose an alternate method: first we use a spectral learning method [4] to learn a Predictive State Representation [15] of the BKT HMM directly from the observed history of students' interaction. Then we use a heuristic to extract the parameters of BKT directly from the PSR. Our results show that the learned PSR captures the essential features of the training data, allowing a computationally efficient and practically effective prediction of BKT parameters. In particular, we decreased the time spent on learning the parameters of BKT by almost 30 times on average compared to EM, while keeping the mean accuracy and RMSE of predicting students' performance on the next question statistically the same. Furthermore, by initializing EM with our extracted parameters, we can obtain improvements in accuracy and RMSE.

This paper is organized as follows: Section 2 provides a background on BKT parameter learning and spectral learning of the parameters in PSRs. Section 3 describes our methodology and setting. In Section 4 we present the detailed results of our experiments and compare the BKT model with our model from several points of view. We provide analysis and justification of the results in Section 5. Finally, Section 6 is conclusion and future work.

2. BACKGROUND
In BKT we are interested in a sequence of student answers to a series of exercises on different skills (KCs) in a tutoring system [6]. BKT treats each skill separately, and attempts to model each skill-specific sequence using a binary model of the student's latent cognitive state (the skill is learned or unlearned). Treating state as Markovian, we therefore have five parameters to explain student mastery in each skill: probabilities for initial knowledge, knowledge acquisition, forget, guess, and slip. However, in standard BKT [6], it is typical to neglect the possibility of forgetting, leaving four free parameters.

The main benefit of the BKT model is that it monitors changes in student knowledge state during practice. Each time a student answers a question, the model updates its estimate of whether the student knows the skill based on the student's answer (the HMM observation). However, the typical parameter estimation algorithm for BKT, EM, is prone to converging to erroneous local optima depending on initialization. On the other hand, in the past few years, researchers have introduced a generalization of HMMs called Predictive State Representations (PSRs) [16] that can be extracted from the data using spectral learning methods [8]. The new learning algorithm uses efficient matrix algebra techniques, which avoid the local optima problems of EM (or any other algorithms based on maximizing data likelihood over the HMM parameter space) and run in a fraction of the time of EM. In this section we first review the EM parameter learning of BKT and then provide a brief background on spectral learning of PSRs.

2.1 EM Parameter Learning of BKT

The main problem with BKT parameter learning by EM is the initial values. The EM algorithm is an iterative process. In each iteration, we first estimate the distributions over students' latent knowledge states, and then update the BKT parameters to try to improve the expected log-likelihood of the training data given our latent state distribution estimates. As mentioned before, the iterative nature of EM means that it is prone to getting stuck in local optima. To remedy this problem, researchers often use multiple runs of EM from different starting points; however, the multiple runs can be time-consuming. Calculating the log likelihood of the model in each iteration also involves going through all the training data, which further exacerbates the runtime problem, especially with large data sets.

There are number of studies that try to handle the problems of EM parameter learning by different approaches. In basic BKT [6], the authors tried to solve the problem by imposing a plausible range of values for each parameter—for example setting the maximum value for the guess parameter to be 0.30. Similar approaches have been applied by [2] and [4]. Another study [12] tried to address the local optimum problem by modifying the structure of BKT and using information from multiple skills to estimate each student's prior in particular skills. The same group made an effort [13] to improve BKT by clustering students based on their performance and using different models for students in different clusters.

Beck & Chang [3] discussed another fundamental problem, called identifiability, with learning BKT parameters by maximum likelihood. In their work, they showed that different sets of BKT parameters could lead to identical predictions of student performance. There is still one set that is more plausible based on expert knowledge, but the other set with identical fit tends to predict that the students are more likely to answer a question wrong when they mastered the skill. They recommend the same approach of constraining the values of the parameters into a plausible range based on the domain knowledge. While these studies elucidated the problem of identifiability and gave rules of thumb to follow in order to arrive at plausible parameters, these rules are often specific to a particular domain and do not necessarily generalize. Moreover, constraining EM to move inside a pre-known parameter space is not trivial, and in many cases the optimizer ends up exceeding its iteration threshold walking along the boundaries of the parameter space without converging to the maximum likelihood value.

Pardos & Heffernan [11] suggested running a grid search over the EM parameter initialization space of BKT to try to find which initial values led to good or bad learned parameters. They analyzed the learned parameters and tried to find boundaries for the initial values not based on plausibility but based on the exact error. They showed that choosing initial guess and slip values that summed up to less than one tends to lead EM to converge toward the expert-preferred parameter set.

2.2 Spectral Learning of PSRs

A Predictive State Representation (PSR) [10] is a compact and complete description of a dynamical system. A PSR can be estimated from a matrix of conditional probabilities of future events (*tests* or *characteristic events*) given past events (*histories* or *indicative events*). If the true probability matrix is generated from a PSR or an HMM, then it will have low rank; so, *spectral* methods can approximate a PSR well from empirical estimates of the probabilities [4,5,8,15]. (In practice we estimate a similarity transform of the PSR parameters, known as a Transformed PSR [15].)

We use in particular the spectral algorithm of Boots & Gordon [5] [4]. They applied their method in several applications and compared the results with competing approaches. In particular, they tested the algorithm by learning a model of a high-dimensional vision-based task, and showed that the learned PSR captures the essential features of the environment effectively, allowing accurate prediction with a small number of parameters. Our work uses their published code.[1]

3. METHODOLOGY

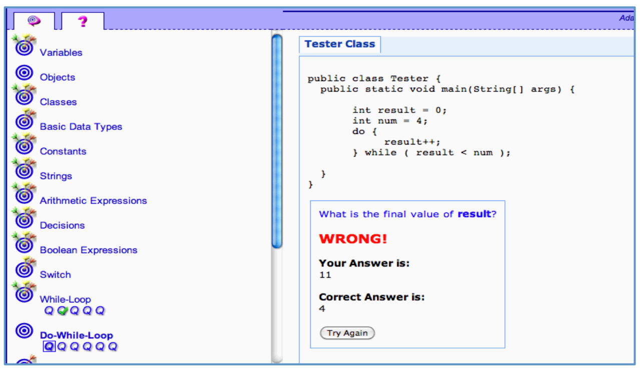
We propose replacing the parameter-learning step of BKT with a spectral method. In particular, we use spectral learning to discover a PSR from a small number of sufficient statistics of the observed sequences of student interactions. We then use a heuristic to extract an HMM that approximates the learned PSR and read the BKT parameters off of this extracted HMM. We can finally use these parameters directly to estimate student mastery levels, and evaluate prediction accuracy with our method compared to the standard EM/MLE method of BKT parameter fitting. We call the above method "spectral knowledge tracing" or SKT. We also evaluated using the learned parameters as initial values for EM in order to get closer to the global optimum. Due to the fact that spectral method does not attempt to maximize likelihood, and also some noise in the translation of the PSR to BKT parameters, the returned BKT parameters are close to the global maximum, but further improvement is available with a few EM iterations. The rest of the section presents a short description of the data along with a brief summary of our student model and analysis procedure.

3.1 Data Description

Our data comes from an online self-assessment tool QuizJET for Java programming. This tool is a part of an adaptive educational system JavaGuide [7] that keeps detailed track of students' interaction to provide adaptive navigation support. The system presents and evaluates parameterized questions to students (programming question templates filled in with random parameters); students can try different versions of the same question several times until they acquire the knowledge to answer them correctly or give up. There are a total of 99 question templates, categorized into 21 topics, with a maximum of 6 question templates within a topic.

We consider each topic as a KC and each question template as a Step toward mastery of the KC. Based on the definition of BKT and KC [6,17] we are only considering the first attempt of each student on each question template, assuming that if a student tried a question template several times until success, they will answer the next question within the topic correctly on the first attempt. This mapping is more coarse-grained than the original definition of KC since we are not dealing the data from an intelligent tutoring system. However, the question templates are designed in such a way that answering all of them correctly will result in mastery of the topic.

[1] http://www.cs.cmu.edu/~ggordon/spectral-learning/



```
public class Tester {
  public static void main(String[] args) {

       int result = 0;
       int num = 4;
       do {
            result++;
       } while ( result < num );

  }
}
```

What is the final value of **result**?

WRONG!

Your Answer is:
11

Correct Answer is:
4

Figure 1: Student view of a question template for the skill "Do-While-Loops".

Figure 1 shows a student view of an example question template. The student can select a topic from the left pane to expand the question templates under each topic. Then s/he can try answering any of the questions under the topic repeatedly whether s/he answers it right or wrong. The system has been in use in the introductory programming classes at the School of Information Sciences, University of Pittsburgh for more than four years. In our study we use data for 9 semesters from Spring 2008 to Fall 2012. Table 1 shows the distribution of records over the semesters.

Table 1: distribution of the records over the semesters.

Semester	#Students	#Topics (Templates) tried	#Records
Spring 2008	15	18 (75)	427
Fall 2008	21	21 (96)	1003
Spring 2009	20	21 (99)	1138
Spring 2010	21	21 (99)	750
Fall 2010	18	19 (91)	657
Spring 2011	31	20 (95)	1585
Fall 2011	14	17 (81)	456
Spring 2012	41	19 (95)	2486
Fall 2012	41	21 (99)	2017
Total	222	21 (99)	10519

The system had no major structural changes since 2008, but the enclosing adaptive system used some engagement techniques in order to motivate more students to use the system. This is the main reason the number of records is higher in the Spring and Fall semesters of 2012.

3.2 Student Model

A time-homogeneous, discrete Hidden Markov Model (HMM) is a probability distribution on random variables $\{(x_t, h_t)\}_{t\in\mathbb{N}}$ such that, conditioned on (x_t, h_t), all variables before t are independent of all those after t. The standard parameterization is the triple (T, O, π) where:

$$T \in \mathbb{R}^{m\times m}, \qquad T_{ij} = \Pr[h_t = i | h_{t-1} = j]$$

$$O \in \mathbb{R}^{n\times m}, \qquad O_{ij} = \Pr[x_t = i | h_t = j]$$

$$\pi \in \mathbb{R}^{m}, \quad \pi_j = \Pr[h_1 = j]$$

O is a mapping from hidden states to output predictions, and T is a mapping between hidden states. Considering our conditional independence properties, T, O, and π fully characterize the probability distribution of any sequence of states and observations [8]. Since the hidden states h_t are not directly observable from the training data, one often uses heuristics like EM to find $\hat{h}_t$, $\hat{T}$, $\hat{O}$ and $\hat{\pi}$ that maximize the likelihood of the samples and the current estimates. In the BKT setting, T is a 2×2 stochastic matrix, so it has two hidden parameters P(learn) and P(forget). O is also a 2×2 stochastic matrix, so it also has two hidden parameters P(guess) and P(slip). And, π is a length-2 probability distribution, so it has one hidden parameter P(init).

Our main contribution is to try extracting these matrices from a learned PSR, giving us the benefit of significantly decreasing training time and avoiding local optima. The details of the spectral algorithm for learning the PSR from the sequence of action-observation pairs are beyond the scope of this paper and can be found in [4]. The algorithm gets a sequence of students' first answers to different question templates within a topic, and builds a PSR using spectral learning. The key parameters of this particular implementation are window sizes used in creating state estimates; we set these to $n_{past} = 10$ and $n_{fut} = 6$. The outputs of the PSR learner are: first, the estimated PSR parameters $\hat{h}_0$, $\hat{A}_1$, and $\hat{A}_2$, and second a set of (noisy) state estimates $\hat{h}_t$, each of which represents a particular time point in the input sequence. We actually added dummy observations before the beginning and after the end of each observation sequence, in order to make the best use of our limited sample size; this means we get four matrices $\hat{A}_i$ from the PSR learner, corresponding to the two original observations plus the two dummy observations. We simply ignore the dummy observations when converting to an HMM.

Nominally, the PSR parameters are related to the HMM parameters by the equations $\hat{\pi} = \hat{h}_0$, $\hat{T} = \hat{A}_1 + \hat{A}_2$, $\hat{O}_i = \hat{A}_i\hat{T}^{-1}$. (Here $\hat{O}_i$ is the diagonal matrix with the ith column of $\hat{O}$ on its diagonal.) However, there is an ambiguity in PSR parameterization: for any invertible matrix S, we can replace each state $\hat{h}_t$ by $S\hat{h}_t$, as long as we replace $\hat{A}_i$ by $S\hat{A}_iS^{-1}$ for $i = 1{,}2$. When we use the modified parameters to compute likelihoods, each pair $S^{-1}S$ cancels, leaving the predictions of the PSR unchanged. So, we have to choose the right transformation S to be able to find parameters $\hat{T}$ and $\hat{O}$ that satisfy the conditions of BKT (each element should be a probability between 0 and 1, and columns should sum to 1).

To pick the transformation matrix S, we designed a heuristic that looks at the state estimates $\hat{h}_t$: we attempt to guess which points in the learned state space correspond to the unit vectors (1,0) and (0,1) in the desired transformation of the learned state space. (We call these the "transformation points.") Given the transformation points, the matrix S is determined. Our heuristic runs in time linear in the length of the input sequence of correct/incorrect observations. Figure 2 shows an overview of the transformation process and Figure 3 shows the details of the heuristic.

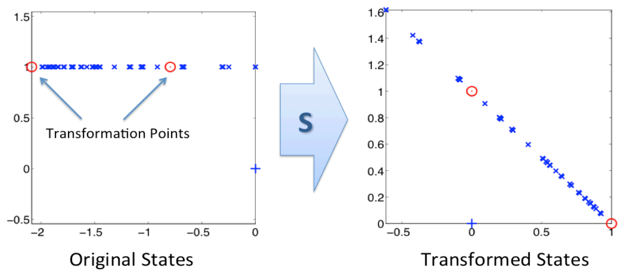


Figure 2: Overview of the transformation scheme.

```
Algorithm FindTransformationPoints(PSR output States)
Find the minimum and maximum values among the
predictive states (mi, ma)
Calculate p = distance between the maximum value
among predictive states and the initial state (s1)
Let n = size(predictive states)
Let step = p / n
For i=1 to n
    Fix the first transformation point to $mi - step$
    Set second transformation point to $s_1 + i \times step$
    Calculate $S$ by linear regression from
    transformation points to (1,0) and (0,1)
    Transform the PSR and calculate $\hat{T}$ and $\hat{O}$
    If $\hat{T}$ and $\hat{O}$ have all elements between 0 and 1
        Break
End
```

Figure 3: Our heuristic to find the transformation points

One slightly subtle point is that, due to noise in the parameter estimates, no matter how we choose the transformation S, the matrices $\hat{O}_i = \hat{A}_i\hat{T}^{-1}$ may not be diagonal. In this case, we simply zero out the off-diagonal elements and renormalize.

3.3 Analysis Procedure

To evaluate our new parameter extraction method, we compared the results of our method with EM learning of BKT parameters as a baseline. We compare both runtime and the ability to predict students' correct/incorrect answer to the next question; for the latter, we calculate both Root Mean Squared Error (RMSE) and prediction accuracy (percent correct). We hypothesize that our spectral method has better performance compared to EM in regard to the time spent on extracting the parameters, while keeping the same accuracy and RMSE of predicting the students' answer to the next question. Since the parameters learned from the PSR are an approximation of the actual global best-fit set of BKT parameters, we also hypothesize that if we use the them as the initial parameters of EM, it will result in a better model in both accuracy and RMSE.

4. RESULTS

For the purpose of mimicking how the model may be trained and deployed in a real world scenario, we learn the model from the first semester data and test it on the second semester, learn the model from the first and second semester data and test it on the third semester, and so on. In total, we calculated results for 155 topic-semester pairs. All analysis was conducted in Matlab on a laptop with a 2.4 GHz Intel® Core i5 CPU and 4 GB of RAM.

4.1 EM Results

In our experiments it took around 36 minutes for EM to fit the parameters, which is on average 15 seconds for each topic-semester pair. In 2 out of 155 cases, EM failed to converge within the 200-iteration limit. The average accuracy of predicting a student's answer to the next question using the parameters learned by EM is 0.650 with RMSE of 0.464. Figure 4 shows the boxplot of the parameters learned by EM. The average values for prior, learn, forget, guess and slip are: 0.413, 0.162, 0.019, 0.431, 0.295.

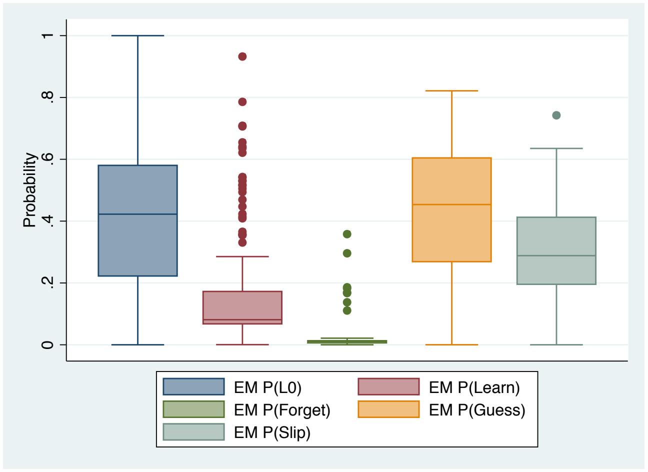


Figure 4: Boxplot of the parameters learned by EM

4.2 SKT Results

It took 1 minute 16 seconds in total for the spectral method to learn the parameters for all semesters and topics; that is almost 30 times faster than EM. The average accuracy of predicting student answer to the next question is 0.664 and RMSE is 0.463. Figure **5** shows the boxplot of the parameters learned by SKT. The average values for prior, learn, forget, guess and slip are: 0.526, 0.268, 0.302, 0.397, 0.271. Note that these values are substantially different from those learned by EM, which means that the calculated student mastery levels will also be different.

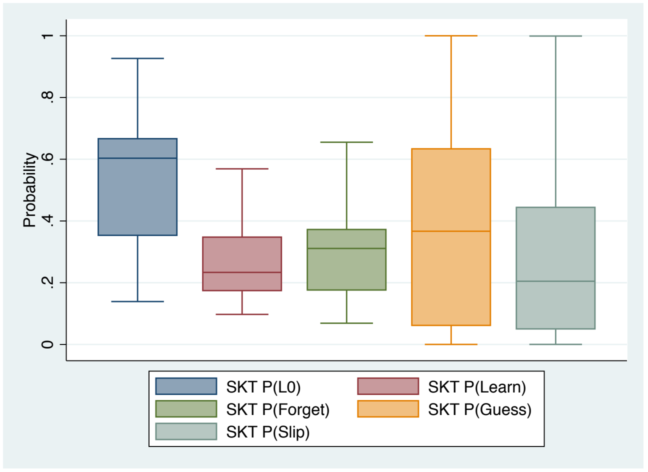


Figure 5: Boxplot of the parameters learned by spectral method

4.3 SEM Results

When we initialized EM with the spectrally learned parameters, the total time was 10 minutes and 40 seconds; that is still substantially faster than plain EM. As expected, the average accuracy of predicting a student's answer to the next question increased to 0.706, and RMSE decreased to 0.422, better than both previous models. Figure 6 shows the boxplot of the refined parameters. The average values for prior, learn, forget, guess and slip are: 0.492, 0.381, 0.360, 0.391, 0.292.

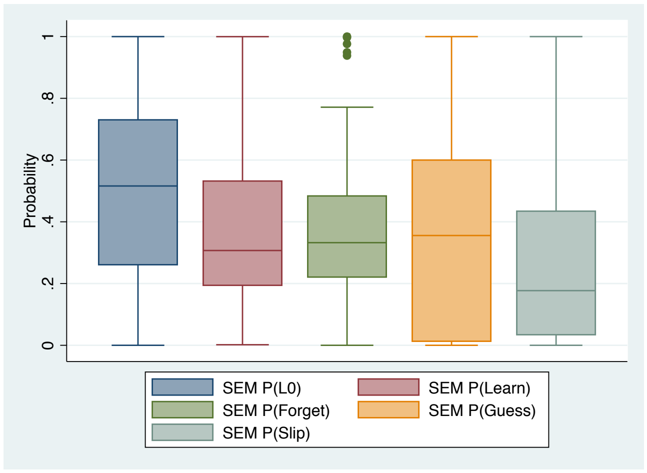


Figure 6: Boxplot of the parameters learned SEM

4.4 Comparison

4.4.1 Time

To get a better understanding of the time complexity of EM and SKT and their relation, we show a semilog plot of the times (Figure 7). We measure the elapsed time of parameter learning using the tic and toc functions of Matlab. Both methods have a similar growth rate as we increase the size of the training data: as we can see in the Figure, the slope of the fitted line for the EM time (green points) is almost the same as the slope of the fitted line for the SKT time (red points). We also tried locally weighted scatter plot smoothing (LOWESS) to compare the runtimes (Figure 8).

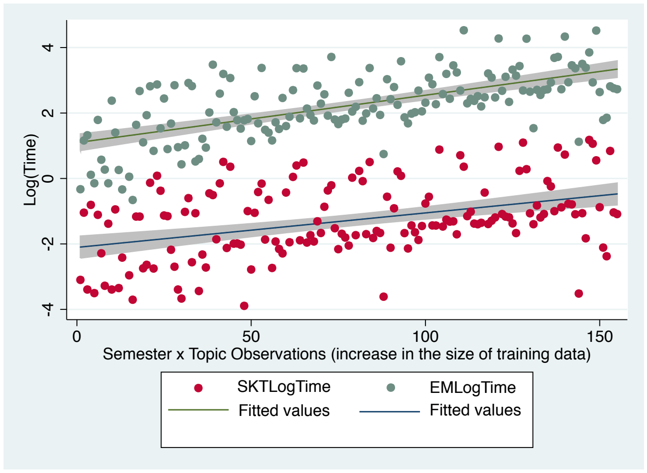


Figure 7: Scatter plot of log(time) with a fitted line

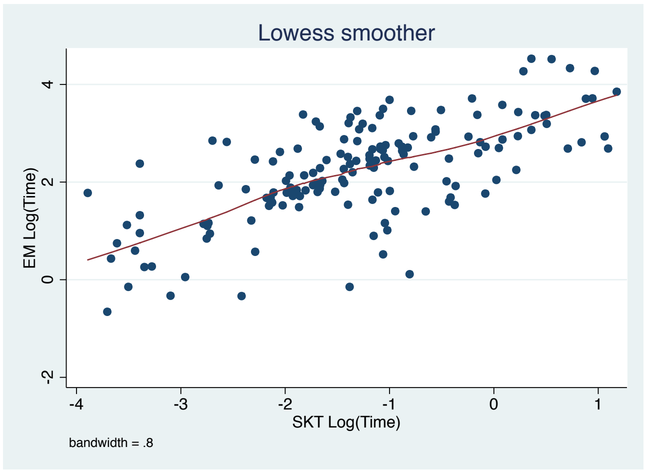


Figure 8: Regression of the Log(time)

The LOWESS plot confirms our intuition that the EM time grows at least linearly compared to the SKT time. To test that hypothesis we tried linear regression on the log-log plot. A 95% confidence interval for the intercept is [2.82, 3.18], which excludes an intercept of 0; a 95% interval for the slope is [.51, .70], which excludes a slope of 1. This can be interpreted as: the time spent learning parameters using EM is on average at least $e^{2.82} \approx 16.77$ times greater than the time spent learning the parameters using SKT, and the scaling behavior of EM is likely to be worse (the ratio gets higher as the data gets larger).

4.4.2 Accuracy and RMSE

Figure 9 and Figure 10 show the histogram of prediction accuracy and RMSE for the 3 models. By looking at the histograms, we can say that the results are approximately normally distributed with about the same variance, but different means.

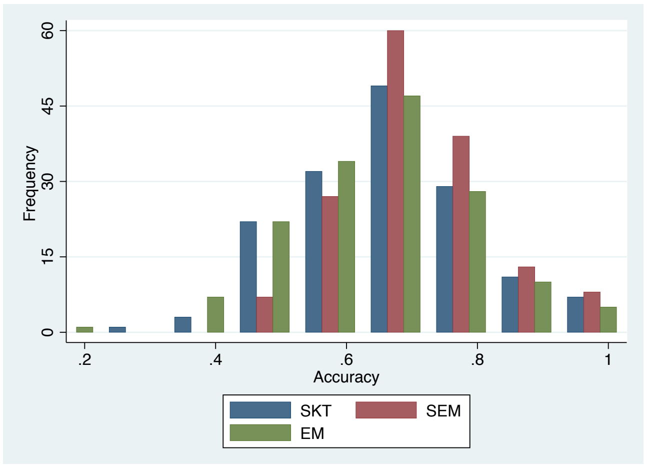


Figure 9: Histogram of Prediction Accuracy

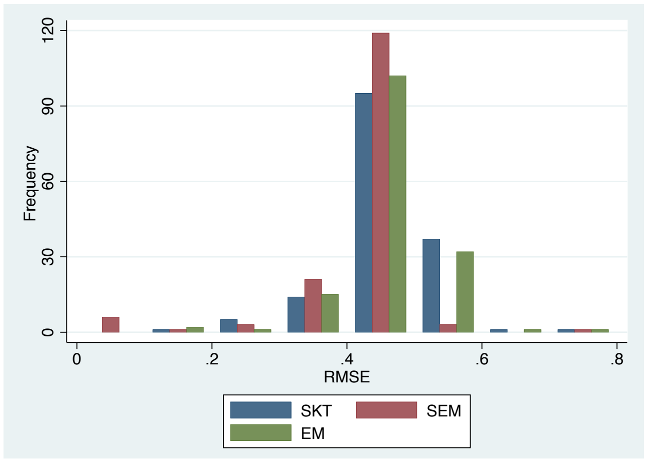


Figure 10: Histogram of the Prediction average RMSE

Regarding prediction accuracy, both of our methods significantly improved the prediction results (p=0.017 SKT vs. EM, p<<0.001 SEM vs. EM, paired t-test, 153 degrees of freedom). Regarding RMSE, the spectrally learned parameters do not result in a significant improvement compared to BKT, but the combination of SKT with EM leads to a significantly better (lower) RMSE compared to BKT (p<<0.001, paired t-test, 153 dof). Table 2 shows the summary of the results. Figure 11 and Figure 12 show the boxplot of the prediction accuracy and RMSE respectively.

Table 2: Summary of the results

Method	Accuracy	RMSE
BKT	0.649 (baseline)	0.465 (baseline)
SKT	0.664 (p=0.017)	0.464 (p=0.348)
SEM	0.706 (p<<0.01)	0.422(p<<0.01)

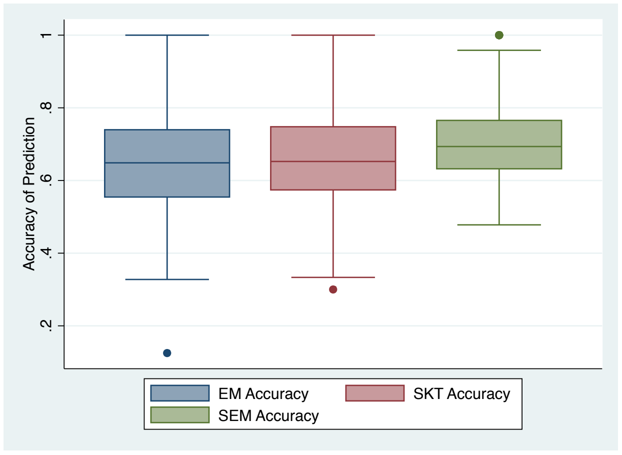


Figure 11: Boxplot of the accuracy

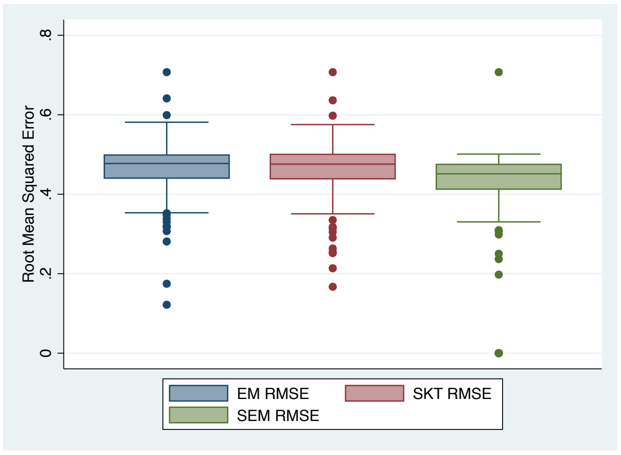


Figure 12: Boxplot of the RMSE

5. DISCUSSION

Based on the results of our study, we found that the spectrally learned parameters can be used directly in the BKT setting, and decrease the time spent on learning parameters by a factor of almost 30 while keeping the same performance in regard to prediction accuracy and RMSE. On the other hand, if we use the spectrally learned parameters to initialize the BKT EM optimization, we can get significantly improved results and still have the advantage of shorter time spent on learning the parameters.

In a setting with a huge number of students and lots of data over several semesters, e.g., an adaptive educational system, the spectrally learned parameters are more helpful in keeping the time spent on building the model for each topic tractable. However, in a more delicate environment, like a cognitive tutor, in which the parameters of BKT are the main basis of the system, we can use the combination method, SEM, and build a more accurate student model in order to predict mastery in different skills.

6. CONCLUSION AND FUTURE WORK

In this paper we presented a novel spectral method for learning the parameters of BKT directly from students' sequences of correct/incorrect responses. One direction for future work would be to compare our method (learn a PSR and extract HMM parameters) to recent algorithms for directly learning an HMM by spectral methods [1], and perhaps combine ideas from these methods with our heuristic.

Another future direction is that, since spectral algorithms have recently been used to learn the parameters of different types of graphical models [9], the results of our study open a new direction for future research on learning complex latent variable models (variations of BKT) directly from student performance data.

From a practical point of view, the results of our study will help us improve our adaptive educational system. Currently, JavaGuide uses a knowledge accumulation approach, based on the total number of correct answers, to estimate students' mastery within each topic for adaptation purposes. The SEM model can be used to improve the system by providing a more accurate (in regard to predicting the student answer to the next question) estimate of student knowledge.

7. ACKNOWLEDGMENTS

This work is partially supported by the Director's Interdisciplinary Graduate Fellowship, and was an extension of a project initiated at the 8th Annual 2012 LearnLab Summer School at CMU.

8. REFERENCES

1. Anandkumar, A., Ge, R., Hsu, D., Kakade, S.M., and Telgarsky, M. Tensor decompositions for learning latent variable models. (2012).
2. Baker, R.S.J., Corbett, A.T., and Aleven, V. More Accurate Student Modeling Through Contextual Estimation of Slip and Guess Probabilities in Bayesian Knowledge Tracing. *Intelligent Tutoring Systems*, Springer (2008), 406–415.
3. Beck, J.E. and Chang, K. Identifiability: A Fundamental Problem of Student Modeling. *User Modeling 2007 4511*, (2009), 137–146.
4. Boots, B., Siddiqi, S.M., Gordon, G.J., and Byron Boots, S.M.S. Closing the learning-planning loop with predictive state representations. *The International Journal of Robotics Research 30*, 7 (2009), 954–966.
5. Boots, B., Gordon, G.J. Predictive State Temporal Difference Learning. In J. Lafferty, C.K.I. Williams, J. Shawe-Taylor, R.S. Zemel and A. Culotta, eds., *Advances in Neural Information Processing Systems 23*. 2010, 271–279.
6. Corbett, A.T. and Anderson, J.R. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modelling and UserAdapted Interaction 4*, 4 (1995), 253–278.
7. Hsiao, I.-H., Sosnovsky, S., and Brusilovsky, P. Guiding students to the right questions: adaptive navigation support in an E-Learning system for Java programming. *Journal of Computer Assisted Learning 26*, 4 (2010), 270–283.
8. Hsu, D., M Kakade, S., Zhang, T., and Daniel Hsu, S.M.K. A Spectral Algorithm for Learning Hidden Markov Models. *Journal of Computer and System Sciences 78*, 1 (2008), 1–22.
9. Ishteva, M., Song, L., Park, H., Parikh, A., and Xing, E. Hierarchical Tensor Decomposition of Latent Tree Graphical Models. *The 30th International Conference on Machine Learning (ICML 2013)*, (2013).
10. Michael L. Littman, R.S.S., Singh, S.P., Littman, M.R.S.S., S Sutton, R., and P Singh, S. Predictive Representations of State. *Neural Information Processing Systems 14*, 14 (2001), 1555–1561.
11. Pardos, Z.A. and Heffernan, N.T. Navigating the parameter space of Bayesian Knowledge Tracing models: Visualizations of the convergence of the Expectation Maximization algorithm. *EDM*, www.educationaldatamining.org (2010), 161–170.
12. Pardos, Z.A. and Heffernan, N.T. *Modeling individualization in a bayesian networks implementation of knowledge tracing*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.
13. Pardos, Z.A., Trivedi, S., Heffernan, N., Sárközy, G.N., and Zachary A. Pardos, S.T. Clustered Knowledge Tracing. *Intelligent Tutoring Systems ITS 12*, Springer Berlin Heidelberg (2012), 405–410.
14. Pavlik, P.I., Cen, H., and Koedinger, K.R. Performance Factors Analysis --A New Alternative to Knowledge Tracing. *Proceedings of the 2009 Conference on Artificial Intelligence in Education: Building Learning Systems That Care: From Knowledge Representation to Affective Modelling*, (2009), 531–538.
15. Rosencrantz, M., Gordon, G., and Thrun, S. Learning low dimensional predictive representations. *Twenty-first international conference on Machine learning - ICML '04*, ACM Press (2004), 88.
16. Singh, S., James, M.R., and Rudary, M.R. Predictive state representations: a new theory for modeling dynamical systems. *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, AUAI Press (2004), 512–519.
17. Vanlehn, K. The Behavior of Tutoring Systems. *International Journal of Artificial Intelligence in Education 16*, 3 (2006), 227–265.

Optimal and Worst-Case Performance of Mastery Learning Assessment with Bayesian Knowledge Tracing

Stephen E. Fancsali, Tristan Nixon, and Steven Ritter
Carnegie Learning, Inc.
437 Grant Street, Suite 918
Pittsburgh, PA 15219, USA
(888) 851.7094 {x219, x123, x122}
{sfancsali, tnixon, sritter}@carnegielearning.com

ABSTRACT
By implementing mastery learning, intelligent tutoring systems aim to present students with exactly the amount of instruction they need to master a concept. In practice, determination of mastery is imperfect. Student knowledge must be inferred from performance, and performance does not always follow knowledge. A standard method is to set a threshold for mastery, representing a level of certainty that the student has attained mastery. Tutors can make two types of errors when assessing student knowledge: (1) false positives, in which a student without knowledge is judged to have mastered a skill, and (2) false negatives, in which a student is presented with additional practice opportunities after acquiring knowledge. Viewed from this perspective, the mastery threshold can be viewed as a parameter that controls the relative frequency of false negatives and false positives. In this paper, we provide a framework for understanding the role of the mastery threshold in Bayesian Knowledge Tracing and use simulations to model the effects of setting different thresholds under different best and worst-case skill modeling assumptions.

Keywords
Cognitive Tutor, intelligent tutoring systems, knowledge tracing, student modeling, mastery learning

1. INTRODUCTION
Carnegie Learning's Cognitive Tutors (CTs) [12] and other intelligent tutoring systems (ITSs) adapt to real-time student learning to provide efficient practice. Such tutors are structured around cognitive models, based on the ACT-R theory of cognition [1-4], that represent knowledge in a particular domain by atomizing it into knowledge components (KCs). CTs for mathematics, for example, present students with problems that are associated with skills that track mathematics KCs in cognitive models. Content is tailored to student knowledge via run-time assessments that probabilistically track student knowledge/mastery of skills using a framework called Bayesian Knowledge Tracing (BKT) [8].

Even in cases in which BKT mastery learning judgments are based on parameters that perfectly match student parameters (e.g., with idealized, simulated student data), assessment of mastery or knowledge is imperfect; student performance need not perfectly track knowledge. In this context, mastery learning assessment is a kind of classification problem. Like all classifiers, an ITS is subject to two types of errors when assessing student knowledge: (1) false positives, in which a student without knowledge is judged to have mastered a skill, and (2) false negatives, in which a student is presented with additional practice opportunities after acquiring knowledge.

A false positive judgment results in pushing a student too quickly through the curriculum. Students pushed too quickly may be asked to demonstrate or use knowledge that they have not yet acquired fully. False negative judgments result in pushing a student too slowly, so the risk is that valuable instructional time is taken teaching KCs that are already mastered, rather than learning new KCs.

Depending on instructional objectives and course design, these two types of errors may not be equally important. If we present a mixed-practice curriculum in which a student will receive more practice on KCs in the future, it may be acceptable to focus on minimizing false negatives. However, if a student will receive only a single block of practice on a KC, particularly if it constitutes important pre-requisite knowledge for later material, then we will strongly prefer to minimize false positives, even at the expense of incurring additional over-practice.

Since detecting mastery typically requires a number of correct trials following the student's attainment of knowledge, a certain amount of "lag" between the point where a student acquires knowledge of a skill and the point where a tutor detects student mastery may be inevitable. We illustrate progression to mastery in Figure 1, dividing opportunities into three phases: learning before the student acquires knowledge of the skill (from opportunity 1 to K), "lag" practice opportunities immediately after knowledge acquisition (K to L), and over-practice after this lag, but before mastery judgment at opportunity M.

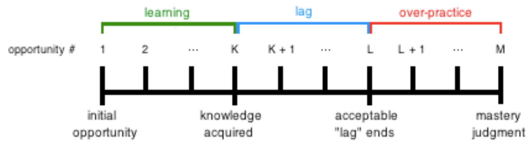


Figure 1. Progression to mastery (judgment) over M student-skill opportunities divided into three "phases"

Despite imperfect assessment of the student, adaptive tutors attempt to minimize the number of opportunities at which students practice skills they have already mastered, so they can focus student practice on skills they have yet to master. We investigate the impact of several factors on the efficiency of practice, focusing especially on the threshold used for student mastery assessments.

We provide a framework for thinking about inherent trade-offs between the two types of CT assessment errors. We quantify the notions of "lag" and over-practice and investigate their relationships with the BKT probability threshold for mastery learning, mastery learning skill parameters, and the dynamics of the student population (or sub-populations) being modeled.

Recent work also focuses on the efficiency of student practice given various methods of individualizing student parameters (e.g., [7] [9]). Increased efficiency has been quantified as the amount of time saved (e.g., by improved cognitive models), without negatively impacting learning [3]. Despite concerns to the contrary, recent work suggests that over-practice is not necessary for long-term retention [7]. Other work conceptualizes the problem of efficient practice roughly as we do, quantifying efficiency and over-practice in terms of expected counts of student opportunities [9]. This work differs in several important ways from past work, especially by focusing on *simulated* data for a variety of skills and quantifying a notion of an acceptable lag while not focusing on individualization.

2. BAYESIAN KNOWLEDGE TRACING

BKT [8] provides a method to track student knowledge acquisition, and it is the basis of the mastery learning implementation in the Cognitive Tutor. For each skill, a student can be in one of two knowledge states: "unknown" or "known." At each opportunity to practice a skill, the student generates an observable (correct or incorrect) response. Four parameters comprise the model of student behavior. The first two are called learning parameters, and the last two are performance parameters:

- $P(L_0)$: initial probability skill is known at first opportunity to practice it
- $P(T)$: probability that student learns the skill (i.e., transition from the unknown to the known state) after an opportunity to practice the skill
- $P(G)$: probability that student produces a correct response at an opportunity despite not knowing the skill ("guessing")
- $P(S)$: probability that student produces an incorrect response at an opportunity despite knowing the skill ("slipping").

Corbett and Anderson [8] provide a well-known algorithm to estimate a student's knowledge state in response to each observable student action and given parameters. The Cognitive Tutor implements run-time mastery assessment using such an algorithm. Mastery of a skill is usually declared when the algorithm determines that a student has 95% probability of being in the known state for the skill. We treat this mastery threshold as a tunable parameter that controls the relative frequency of the two types of mastery assessment errors.

One way to think about false positive errors is as the proportion of students for whom at least one false positive occurred, i.e., the proportion of students for whom mastery of a skill was judged pre-maturely by the run-time algorithm. This provides an accounting of how many students the system moves on to practice new skills too quickly.

The regular occurrence of a modest "lag" (i.e., of false negatives) is both expected and vital to a tutoring system remaining "conservative" in the sense that it infrequently commits false positive errors. Some lag may be required because of uncertainty inherent in the BKT model. We can never be completely sure that correct performance results from student knowledge, rather than a guess. As we observe repeated correct performances, we become more certain that the behavior results from underlying knowledge, rather than just guessing. It is this transition from uncertainty to certainty that is the source of what we will call the "acceptable lag" after knowledge acquisition. The mastery threshold represents the point at which we consider the system to be certain enough to conclude that the student has mastered the skill.

A well-calibrated, adaptive tutoring system should not frequently prescribe large amounts of "over-practice." We quantify the notion of the acceptable lag as well as over-practice and assess the proportion of students that experience over-practice.

3. SIMULATION REGIME

We use the BKT model to generate idealized data for simulated students in a manner comparable to [6] and [11]. For example, if $P(L_0) = 0.5$, $P(T) = 0.35$, $P(G) = 0.1$ and $P(S)=0.1$, then the simulation would, for each simulated student, place the student in the known state initially with a probability of 0.5. Students in the known state would then generate correct responses with a 0.45 probability $[P(L_0)*(1-P(S))]$. Those in the unknown state would generate correct responses with probability 0.1, and have a 0.35 probability of transitioning into the known state. Percent correct on the first opportunity for all students simulated with this skill would be 0.5 $[P(L_0)*(1-P(S))+(1- P(L_0))*P(G)]$.

Since we know exactly when each virtual student transitioned into the known state, we can compare the point where this occurred to the judgment of the BKT run-time mastery algorithm, which can only observe the generated student actions. We apply this testing paradigm to scenarios where the runtime system uses the same BKT parameters as the generating model ("best-case"), and to a couple of scenarios where they are significantly different ("worst-cases").

We simulate data over skills represented by 14 unique parameter quadruples, a subset of those identified in [13] as representative of broad clusters of skills deployed in Cognitive Tutor mathematics curricula[1]. We ascertain the number of "lagged" opportunities we expect students to see, the frequency that the number of lagged opportunities can reasonably be considered over-practice (i.e., beyond the acceptable lag), and the frequency of pre-mature mastery judgment, for one best-case and two worst-case scenarios.

4. RESULTS

There are several ways of thinking about best and worst-case scenarios; we do not exhaust the space of possibilities. We begin by considering a best-case scenario.

4.1 Best Case: Homogeneous, Matching Student Population

In our first round of simulations, we simulate response data, for four mastery threshold probabilities (75%, 90%, 95%, and 98%). We assume that students are homogeneous with respect to a skill, meaning that student behavior is generated probabilistically from the same set of BKT parameters for all students.

For each skill parameter set, we simulate 10,000 students in this way, for up to thirty opportunities per student. Since we are implementing mastery learning, the actual number of opportunities generated by a simulated student depends on when and if the student reaches mastery. In this best-case scenario, the system judges mastery by the same skill parameters that are used

[1] Ritter, et al. [13] identify a total of 23 "cluster" skill quadruples inferred from empirical data collected for thousands of skills in CT curricula. We discard seven quadruples with $P(L_0) > 0.75$. Two quadruples are discarded that cover little of the empirical parameter space and seem to have implausible values for P(G) and P(S) (cf. [5]).

to generate student behaviors. That is, the system is correctly modeling student skills.

Since these are simulated students, we know when students learn each skill (by transitioning from the unknown to known state) in addition to their observed behavior, so we compare the internal knowledge state at opportunities to those at which the BKT run-time algorithm judges a student to have reached a sufficiently high probability of knowing each skill. Given the large sample of students and number of simulated opportunities, individual simulations for each mastery threshold should be comparable.

4.1.1 Efficient Practice

Figure 2 provides frequencies with which values of the median[2] number of lagged or over-practice opportunities occur over the 14 skills for four mastery thresholds. As we increase the mastery threshold, we expect the ordinary student to see more lagged opportunities (i.e., opportunities after knowledge acquisition). At the 95% mastery threshold, for example, we expect the median student to see one to four lagged opportunities on most skills. At the 98% threshold, more skills have a median lag of five or more opportunities.

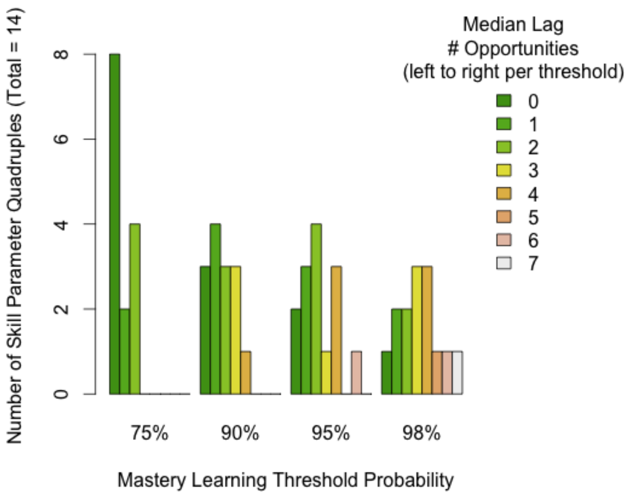


Figure 2. Frequency (# of skills) of median student lag opportunities (i.e., those beyond knowledge acquisition) [14 skills simulated for 10,000 students at each threshold; student BKT parameters match run-time mastery learning parameters.]

Figure 3 provides distributions over skills of the frequency with which student pre-mature mastery judgment (false positives) occurs for the four mastery thresholds. Coupled with Figure 2, we see a trade-off between pre-mature mastery judgments, which decrease, and lagged opportunities, which increase, as we increase the mastery threshold. We expect no more than 5% (indeed, generally less than 5%) of students to be pre-maturely judged as having acquired skill knowledge/mastery at the 95% and 98% thresholds.

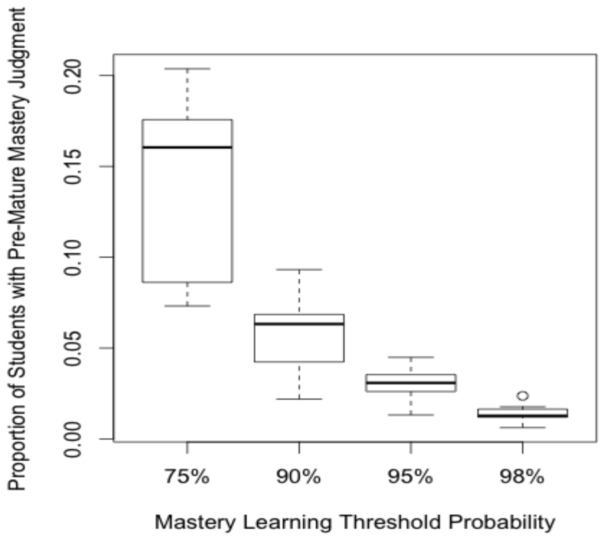


Figure 3. Distribution of proportion of simulated students (per skill) pre-maturely judged to have skill mastery [14 skills simulated for 10,000 students at each threshold; student BKT parameters match run-time mastery learning parameters]

4.1.2 Over-Practice

We seek to quantify over-practice, and the extent to which ideal students endure it, as a function of mastery thresholds and CT mastery learning parameters.

4.1.2.1 Acceptable Lag After Knowledge Acquisition

Recall that the second phase in the progression to mastery in Figure 1 begins at knowledge acquisition (opportunity K) and continues until the end of what we have called the acceptable lag at opportunity L. We define this acceptable number of lagged opportunities for each particular skill and mastery threshold so that we can quantify over-practice as opportunities after knowledge acquisition beyond an acceptable lag.

We begin by noting that properties of the BKT model entail that the inferred probability of student skill knowledge never falls to zero for non-zero $P(L_0)$, $P(T)$, and $P(G)$. Rather, each skill parameter quadruple implies a theoretical minimum probability of knowledge. We can estimate the theoretical minimum probability of knowledge per skill by simulating "runs" of consecutive incorrect student opportunities and noting the asymptotic value to which the probability of knowledge decreases.

Figure 4 shows how the BKT estimated probability of knowledge for the skill with $P(L_0) = 0.631$, $P(T) = 0.11$, $P(G) = 0.282$, and $P(S) = 0.228$, decreases over a series of consecutive incorrect responses from $P(L_0)$ to its theoretical minimum at roughly 0.161 in about six opportunities.

We determine the number of consecutive correct opportunities required to take a student from the theoretical minimum probability of knowledge, for each skill, to the mastery threshold probability. The length of this run is the acceptable amount of lagged opportunities[3]; any simulated student that encounters a lag

[2] With few exceptions, in all simulations we report, for each skill, the median, mean, and modal number of lagged opportunities are relatively close in value. The median takes on an integral value in these simulations, making it more readily interpretable.

[3] This definition presumably provides an upper bound for this value. Other definitions may be appropriate (e.g., based on analysis of empirical data), but we leave this topic for future research.

of opportunities with length greater than the acceptable number is considered to encounter over-practice for that skill. For the skill with theoretical minimum illustrated in Figure 4, the acceptable lags at the 75%, 90%, 95%, and 98% thresholds are 3, 4, 4, and 5, respectively. We determine the proportion of students who encounter over-practice and compare this to the proportion of students with pre-mature mastery judgment.

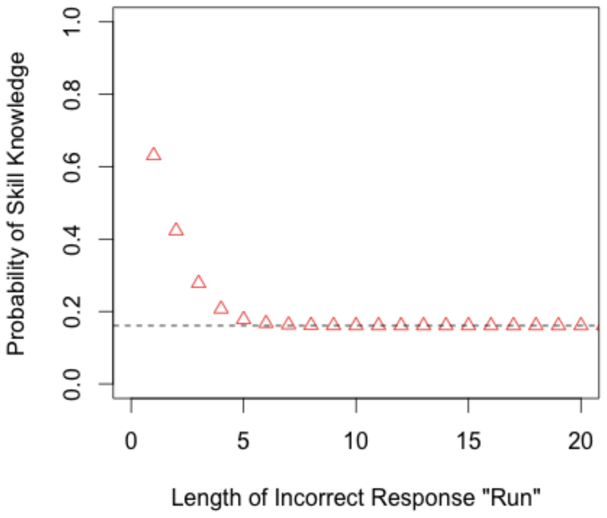


Figure 4. Illustration of theoretical minimum probability of skill knowledge over "run" of consecutive incorrect responses for a skill [$P(L_0) = 0.631$; $P(T) = 0.11$; $P(G) = 0.282$; $P(S) = 0.228$]; dashed-line marks approximate asymptote at 0.161.

4.1.2.2 Frequency and Magnitude of Over-Practice

Figure 5 shows that by increasing the threshold for mastery we tend to increase the proportion of students to whom over-practice opportunities are provided[4] (as well as the variability of this proportion over skills). This illustrates the trade-off between pre-mature mastery judgment and over-practice (in addition to the noted trade-off between pre-mature mastery and lagged practice opportunities).

Notably, increasing the threshold does not drastically increase the *number* of over-practice opportunities the median simulated student is expected to see, only the probability that a student will get some over-practice. The median number of over-practice opportunities per student-skill interaction with over-practice is 1 for the 75%, 90%, and 95% thresholds, and 2 for the 98% threshold. While over-practice is assigned for roughly 30% of students at the traditional 95% threshold, the median student, for most skills, does not experience much over-practice. The traditional 95% mastery criterion seems to embody a conservative tradeoff: some students will receive a small amount of over-practice, but pre-mature mastery judgment is mostly avoided.

[4] It is not clear to us why the median proportion for skills at the 98% threshold is lower than the 95% threshold. These median values are closer (with the median at 98% greater than that at 95%) under other conditions we describe later.

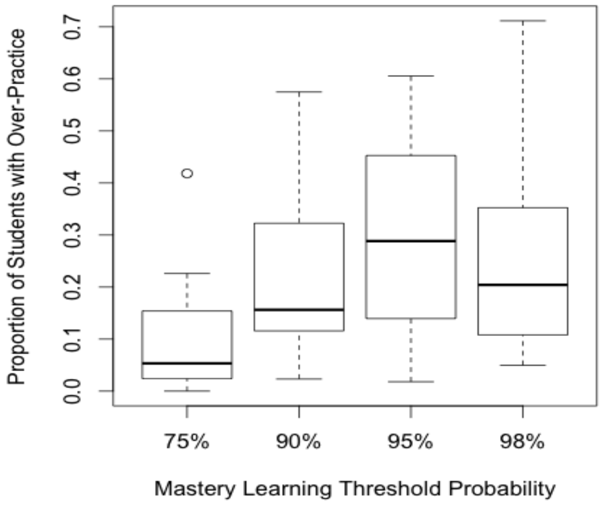


Figure 5. Distribution of proportion of simulated students (per skill) assigned over-practice opportunities (i.e., at least one opportunity beyond the acceptable lag for a particular skill) [14 skills simulated for 10,000 students at each threshold; student BKT parameters match run-time mastery learning parameters.]

4.2 Worst-Case #1: Homogenous, Non-Matching Student Population

Next, we consider one type of worst-case scenario. Simulated students are drawn from a homogenous population, but these student parameters uniformly mismatch the BKT parameters used for run-time mastery assessment. That is, the system is doing the poorest possible job of modeling the student's learning parameters. For the same 14 skills, we specify mismatched student generating parameters in the following manner where P_S stands for mismatched student parameters and P_M corresponds to mastery assessment parameters that will be used:

- $P_S(L_0) = 1 - P_M(L_0)$
- $P_S(T) = 1 - P_M(T)$
- $P_S(G) = 0.5 - P_M(G)$
- $P_S(S) = 0.5 - P_M(S)$

We generate data, in the same manner as the previous section, for 10,000 students for up to thirty opportunities.

4.2.1 Efficient Practice

Figure 6 provides the frequency with which particular median lagged opportunity counts occur over the 14 skills at the four mastery thresholds. We see a shift toward more skills with greater median counts of lagged opportunities, especially at the 95% and 98% thresholds. Coupled with Figure 7, we see evidence of the same trade-off between pre-mature mastery judgment and lag opportunities as in the best-case scenario, but we find (mostly) far lower proportions of students at each threshold with pre-mature mastery judgment.

Figure 7 also makes apparent that two skills have substantially greater proportions of students with pre-mature mastery. These correspond to two of the 14 skills with $P(T) > 0.8$. Since they have a high mastery learning $P_M(T)$ parameter and simulated students have $P(T) < 0.2$, the mastery learning assessment naturally counts students as acquiring knowledge pre-maturely with greater frequency.

For this scheme of generating worst-case, mismatching student parameters (and the corresponding 14 skills' parameters), we again find evidence that BKT mastery assessment is generally conservative, erring on the side of providing students with more opportunities after knowledge acquisition, rather than pre-maturely judging mastery.

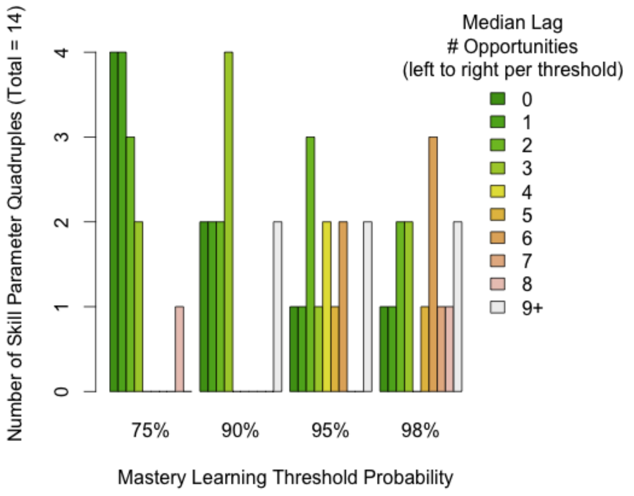


Figure 6. Frequency (count of skills) of median student opportunities beyond knowledge acquisition for four run-time mastery threshold probabilities [Student BKT parameters uniformly "mismatch" 14 run-time mastery learning parameters.]

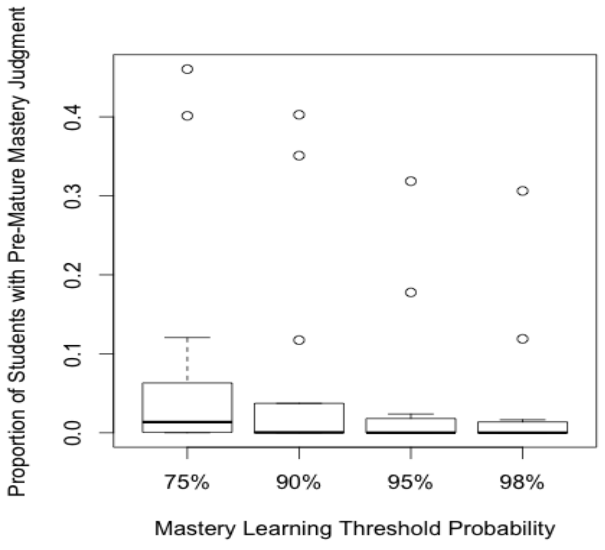


Figure 7. For "mismatched" student skill parameters, distribution of proportion of simulated students (per skill) pre-maturely judged to have skill mastery, grouped by run-time mastery thresholds

4.2.2 *Over-Practice*

Compared to the best case "matching" parameter scenario, Figure 8 shows that the proportions of students that experience over-practice at each mastery threshold are far greater (and increase with increasing mastery threshold). However, the amount of over-practice through which simulated students must work remains modest; the median student that experiences over-practice sees 2 over-practice opportunities per skill over all the skills at the 75% threshold, 3 at the 90% and 95% threshold, and 4 at the 98% threshold. Again, we do not witness particularly onerous over-practice in general, despite the mismatch of student parameters and mastery assessment parameters.

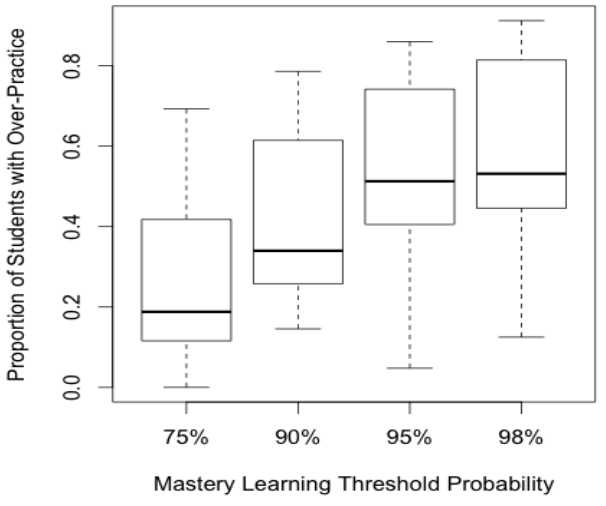


Figure 8. For "mismatched" student skill parameters, distribution of proportion of simulated students (per skill) assigned over-practice opportunities

4.3 Worst Case #2: Completely Heterogeneous Population, Random Student Parameters

We now consider simulating maximally heterogeneous populations of simulated students for the same 14 skills. Mastery learning parameters correspond to the 14 skill parameter quadruples, but for each skill and each student, BKT parameters for a generating model are randomly sampled as follows:

- $P_S(L_0)$, $P_S(T)$ ~ Uniform(0.0, 1.0)
- $P_S(G)$, $P_S(S)$ ~ Uniform(0.0, 0.5).

This corresponds to testing each CT skill parameter quadruple for robustness against a worst-case in which there are no stable sub-populations of students; data for each student are drawn from a different, random generating model. We consider how such a scenario would affect the CT's ability to assign efficient practice based on BKT mastery assessment.

4.3.1 *Efficient Practice*

For worst-case #2, the pattern of trade-offs in efficient practice are the same as in the previous two cases, but frequencies of median lagged opportunities (Figure 9) and proportions of students judged for mastery pre-maturely (Figure 10) fall in between the best-case scenario and the previous worst-case scenario. There is also a larger variance in the distribution of the proportion of students per skill that get pre-mature mastery judgments than in either of the previous two scenarios we have considered.

This accords with our expectations, as randomly generated parameters will sometimes be very close (and other times far removed) from each skill's mastery learning parameters. Thus, a maximally heterogeneous population is really a mixture of best-case students, worst-case students, and students somewhere in the "middle" of the two. Further, given random generation of student parameters we reasonably expect an increase in variance in pre-mature mastery judgment compared to either of the homogeneous simulated student populations we have considered. As with the mismatching student population, the two outlier skills for premature mastery judgment are those with P(T) > 0.8, but the proportion for these two skills is smaller than in the case of the mismatching population, as we expect.

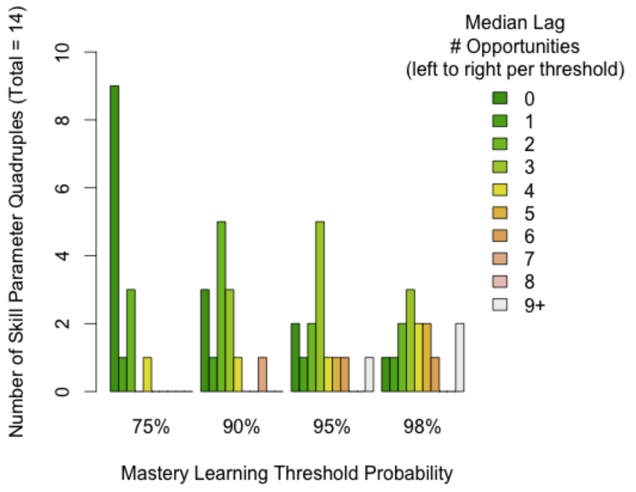


Figure 9. Frequency (count of skills) of median student over-practice opportunities for four run-time mastery threshold probabilities with random student BKT parameters & 14 "cluster" skill quadruples as mastery parameters

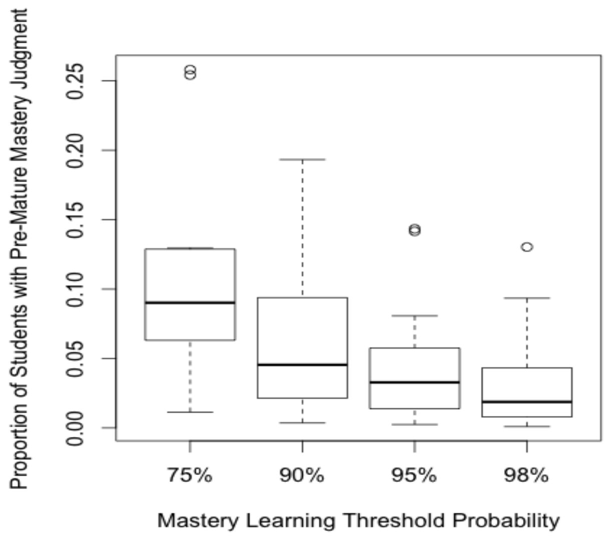


Figure 10. For random student skill parameters, distribution of proportion of simulated students (per skill) pre-maturely judged to have skill mastery

4.3.2 Over-Practice

As in the previous two cases, we see that the proportions of students experiencing over-practice per skill generally increase as we increase the mastery threshold probability (Figure 11). Further, most values of these proportions fall roughly in between the medians for the previous two cases. Median counts of over-practice opportunities over all skills at each mastery threshold are also similar to those in the other scenarios (median = 2 for 75%, 90%, and 95%; median = 3 for 98%).

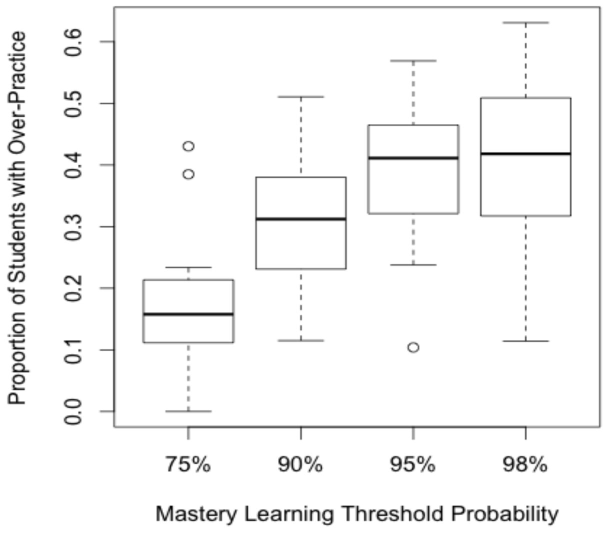


Figure 11. For random student skill parameters, distribution of proportion of simulated students (per skill) assigned over-practice opportunities

5. VISUALIZATION WITH ROC CURVES

One way to conceptualize assessing student skill mastery is as the problem of "detecting" learning from a noisy signal. With errors in mastery assessment cast in terms of false positives and false negatives, or Type I and Type II errors, a natural way to visualize mastery learning classification of student-skill opportunities and trade-offs between false positives and false negatives, as we adjust the mastery threshold, is via Receiver Operating Characteristic (ROC) curves.

Figure 12 provides scatterplots of true positive rate versus false positive rate (roughly ROC curve graphs) for each skill from our simulations with random student skill parameters, grouped by mastery threshold probability. The cluster of points at the bottom left (along with the dearth of points, two outliers aside, in the center and far right of the graph) indicate the relative "conservativeness" of BKT mastery learning for the skill parameters we consider. These points represent true positive rates that are relatively low mostly because of increased false negative errors, corresponding to opportunities that lag student knowledge acquisition (whether over-practice beyond an acceptable lag or not).

Recall that we take a greater proportion of false negatives than false positives to be a virtue of a conservative tutoring system that decreases the risk of pushing students along too quickly. Two outliers on the graph (the two right-most points of the graph) represent skills each with false positive rates near 0.5 and 1, for the 90% and 75% mastery thresholds, respectively. Those points correspond to the same two skills we identified in §4.2.1 and §4.3.1 with P(T) > 0.8. We find a similar cluster of points (and overall structure) in the same graph constructed over best-case simulations. High P(T) means we assume that students learn quickly, so these results may indicate that such an assumption for a skill is more likely to produce high false positive rates, especially for lower mastery thresholds, regardless of the makeup of the student population.

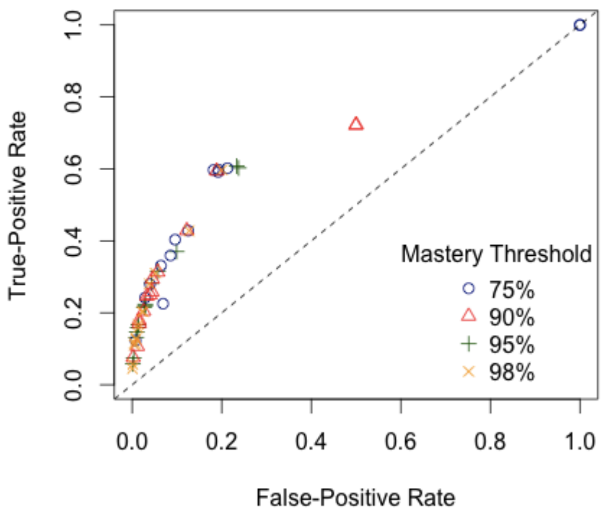


Figure 12. Scatterplots of true positive rate versus false positive rate (roughly ROC graphs) for each skill from simulations with random student skill parameters, grouped by mastery threshold probability

Since rates are calculated over student practice opportunities, false positive instances are limited to one per student per skill, because the student no longer practices a skill after mastery judgment. However, there can be many false negative errors for a single student. Simulating students without mastery learning, in general, would allow us to ascertain more balanced false positive rates for BKT mastery learning.

Analogous graphs can be constructed at the level of students. This introduces some complications, since the assessment of the classification of a student as a true positive presumably depends not just on the performance of the mastery algorithm, but also on the estimation of acceptable lag for that skill. Many students will be counted as true positives, despite the fact that the tutor is committing false negative errors within the acceptable lag at the transaction level. As such, we should expect a lower false negative rate at the student level than at the opportunity level. We leave such extensions for future research, but note that there are important similarities between the type of analysis we provide and "signal detection for learning."

6. DISCUSSION

The trade-off, as a function of mastery probability threshold, of student pre-mature mastery judgment (false positives), lagged skill opportunities, and over-practice (false negatives) is consistent across different best-case and worst-case skill modeling assumptions. The value of the mastery probability threshold and skill modeling assumptions influence the magnitude of these error rates when calculated as proportions of students pre-maturely judged to have achieved mastery or subjected to over-practice. However, we find that for the median student subjected to over-practice, regardless of the skill modeling scenario, the amount of over-practice as a count of opportunities is not, on the surface, particularly onerous. Thus, BKT and a variety of skill parameters are generally robust to committing the types of errors we have quantified, with the exception of two outlier skills we discovered with $P(T) > 0.8$ for which pre-mature mastery judgment occurred more frequently in the two worst-case scenarios. This suggests that without strong empirical evidence that certain skills are learned quickly, we should err on the side of setting lower values of P(T) for mastery learning.

Over all three simulation scenarios, we find that the conventional 95% mastery threshold probability leads to pre-mature mastery judgment for under 5% of students per skill for the majority of cases. Further, onerous over-practice is generally not assigned to students at these thresholds, making it less likely that student time will be "crowded out" by practice for skills they have already mastered and making it more likely that they will cover more course material over all. We emphasize that our results are limited to the skill parameter quadruples we considered that are broadly representative of those deployed in the CT [13], but broader patterns (e.g., that using $P(T) > 0.8$ leads to more false positives) seem to emerge and should be studied further.

Beyond this computational, pragmatic justification for the conventionally deployed mastery threshold, we provide a framework for thinking about the BKT mastery threshold as a parameter that can be tuned according to course developers' appetite for risk, in the sense of trading off false positives for false negatives, and how each type of error will affect student learning, course completion, and other instructional outcomes. There is potential that a moderate amount of over-practice might have additional value in preventing future forgetting. This framework and these (or similar) results could be used to calibrate tutors to optimize student practice for future retention. Finally, we provided a better theoretical understanding of optimal performance of BKT-based mastery learning.

This work calls for extension in several ways. Beyond considering our relatively limited best-case and worst-case scenarios, we should investigate a greater range of average-case possibilities. For example, students with diverse prior knowledge, learning rates, and other learning characteristics use real-world ITSs. How much fine-tuning of run-time mastery learning parameters, to student sub-populations or even individual students (e.g., [10], [11]), is necessary to prevent both over-practice of skills and pre-mature mastery judgment?

Future work should also address a broader, more exhaustive range of BKT parameter quadruples. Those we analyze here are important because they are representative of real-world data collected over thousands of students and skills, but we should seek a better understanding of the full parameter space and how different parameter combinations interact with factors like student sub-population composition and mastery learning thresholds, a greater number of which should also be systematically explored and tested. Finally, depending on the investigator and educator interests, the nature of particular curricula, and other concerns, exploring alternative conceptualizations of over-practice and under-practice (cf. [7]) (and their connection to our work based on practice opportunities counts) is an interesting avenue for future research.

7. ACKNOWLEDGMENTS

The authors acknowledge the helpful comments of Robert Hausmann, R. Charles Murray, Brendon Towle, and Michael Yudelson on an early draft of this paper.

8. REFERENCES

[1] Anderson, J.R. 1983. *The architecture of cognition*. Harvard UP, Cambridge, MA.

[2] Anderson, J.R. 1990. *The adaptive character of thought*. Erlbaum, Hillsdale, NJ.

[3] Anderson, J.R. 1993. *Rules of the mind*. Erlbaum, Hillsdale, NJ.

[4] Anderson, J.R., Bothell, D., Byrne, M.D., Douglass, S., Lebière, C., Qin, Y. 2004. An integrated theory of the mind. *Psychological Rev.* 111, (2004), 1036-1060.

[5] Baker, R.S.J.d., Corbett, A.T., Aleven, V. 2008. More accurate student modeling through contextual estimation of slip and guess probabilities in Bayesian Knowledge Tracing. In *Proceedings of the 9th International Conference on Intelligent Tutoring Systems* (Montreal, Canada, 2008). 406-415.

[6] Beck, J., Chang, K. 2007. Identifiability: a fundamental problem of student modeling. In *Proceedings of the 11th International Conference on User Modeling* (Corfu, Greece, 2007). 137-146.

[7] Cen, H., Koedinger, K.R., Junker, B. 2007. Is over-practice necessary? – Improving learning efficiency with the Cognitive Tutor through educational data mining. In *Proceedings of the 13th International Conference on Artificial Intelligence in Education* (Los Angeles, USA, 2007). 511-518.

[8] Corbett, A.T., Anderson, J.R. 1995. Knowledge tracing: modeling the acquisition of procedural knowledge. *User Modeling & User-Adapted Interaction* 4, (1995), 253-278.

[9] Lee, J.I., Brunskill, E. 2012. The impact of individualizing student models on necessary practice opportunities. In *Proceedings of the 5th International Conference on Educational Data Mining* (Chania, Greece, 2012), 118-125.

[10] Pardos, Z.A., Heffernan, N.T. 2010. Modeling individualization in a Bayesian networks implementation of Knowledge Tracing. In *Proceedings of the 18th International Conference on User Modeling, Adaptation and Personalization* (Hawaii, USA, 2010), 255-266.

[11] Pardos, Z.A., Heffernan, N.T. 2010. Navigating the parameter space of Bayesian Knowledge Tracing models: Visualizations of the convergence of the Expectation Maximization algorithm. In *Proceedings of the 3rd International Conference on Educational Data Mining* (Pittsburgh, USA, 2010), 161-170.

[12] Ritter, S., Anderson, J.R., Koedinger, K.R., Corbett, A.T. 2007. Cognitive Tutor: applied research in mathematics education. *Psychonomic Bulletin & Rev.* 14, 2 (2007), 249-255.

[13] 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.

Automatically Recognizing Facial Expression: Predicting Engagement and Frustration

Joseph F. Grafsgaard[1], Joseph B. Wiggins[1], Kristy Elizabeth Boyer[1], Eric N. Wiebe[2], James C. Lester[1]

[1]Department of Computer Science [2]Department of STEM Education
North Carolina State University, Raleigh, NC, USA

{jfgrafsg, jbwiggi3, keboyer, wiebe, lester}@ncsu.edu

ABSTRACT

Learning involves a rich array of cognitive and affective states. Recognizing and understanding these cognitive and affective dimensions of learning is key to designing informed interventions. Prior research has highlighted the importance of facial expressions in learning-centered affective states, but tracking facial expression poses significant challenges. This paper presents an automated analysis of fine-grained facial movements that occur during computer-mediated tutoring. We use the Computer Expression Recognition Toolbox (CERT) to track fine-grained facial movements consisting of eyebrow raising (inner and outer), brow lowering, eyelid tightening, and mouth dimpling within a naturalistic video corpus of tutorial dialogue (*N*=65). Within the dataset, upper face movements were found to be predictive of engagement, frustration, and learning, while mouth dimpling was a positive predictor of learning and self-reported performance. These results highlight how both intensity and frequency of facial expressions predict tutoring outcomes. Additionally, this paper presents a novel validation of an automated tracking tool on a naturalistic tutoring dataset, comparing CERT results with manual annotations across a prior video corpus. With the advent of readily available fine-grained facial expression recognition, the developments introduced here represent a next step toward automatically understanding moment-by-moment affective states during learning.

Keywords

Facial expression recognition, engagement, frustration, affect, computer-mediated tutoring

1. INTRODUCTION

Over the past decade, research has increasingly highlighted ways in which affective states are central to learning [6, 21]. Learning-centered affective states, such as engagement and frustration, are inextricably linked with the cognitive aspects of learning. Thus, understanding and detecting learner affective states has become a fundamental research problem. In order to identify students' affective states, researchers often investigate nonverbal behavior. A particularly compelling nonverbal channel is facial expression, which has been intensely studied for decades. However, there is still a need to more fully explore facial expression in the context of learning [6].

Recent research has identified facial expressions that are related to self-reported and judged learning-centered affective states [1, 7, 9, 18, 25], which typically include boredom, confusion, engaged concentration, and frustration. However, more research is needed to fully explore the relationships between facial movement and learning-centered affective states. For instance, timing and intensity of facial expressions have only just begun to be explored in the context of learning [18].

The Facial Action Coding System (FACS) [10] has been widely used to study detailed facial movements for decades. FACS enumerates the possible movements of the human face as facial *action units*. Thus, FACS is an objective measure used to identify facial configurations before interpreting displayed affect. Because FACS quantifies facial movements present in displays of emotion, it allows researchers to identify facial components of learning-centered affect, which have been found to be different from those in everyday emotions [4, 6, 7, 9, 18, 27]. Identifying these action units is a time-intensive manual task, but a variety of computer vision tools are in current use, most often focusing on tracking facial feature points [4, 27]. Facial feature tracking tools recognize the presence of a face and then locate facial features such as the corners of the mouth and eyes. Generally, there are two distinct families of tools: low-level tools that track facial features [3] (e.g., which way the head is turned and where points are positioned) and tools that provide affective interpretations [17, 23, 24] (e.g., smiling, emotions). However, the tool used in this study, the Computer Expression Recognition Toolbox (CERT), offers a mid-level alternative. CERT produces intensity values for a wide array of FACS facial action units, thus enabling fine-grained analyses of facial expression [19].

This paper presents an automated facial recognition approach to analyzing student facial movements during tutoring and an examination of the extent to which these facial movements correspond to tutoring outcomes. The novel contributions are two-fold. First, the output of the facial action unit tracking tool, CERT, was validated through comparing CERT output values with manual FACS annotations. The results indicate excellent agreement at the level of presence versus absence of facial movements. Naturalistic video is challenging for computer vision techniques, and this validation is the first of its kind on a naturalistic tutoring video corpus. Second, models were constructed to examine whether the intensity and frequency of facial expressions predict tutoring outcomes. The results show that several specific facial movements predict tutoring outcomes.

For instance, brow lowering intensity (i.e., the magnitude of the CERT output value) was associated with reduced perception of the tutoring session as being worthwhile, and greater self-reported frustration. Additionally, frequency of mouth dimpling predicts increased learning gains and self-reported task success. These results represent a next step toward large-scale analyses and understanding of learning-centered affective states in tutoring.

2. RELATED WORK

D'Mello and colleagues have a longstanding line of research into the mechanisms of facial expression and learning-centered affective states. In recent years, stable correlations between specific facial action units and self-reported or judged affective states have been identified [7, 9]. Brow lowering (AU4) and eyelid tightening (AU7) were correlated with confusion, while inner and outer brow raising (AU1, AU2) were correlated with frustration.

Another prominent line of research is that of Baker and colleagues. After extensively observing student nonverbal behaviors during interactions with tutoring systems, they developed a protocol for judging students' affective states, such as boredom or engagement [1, 22]. This has enabled lightweight annotation of affective states across a wide variety of classrooms. Automated tools extend these approaches to studying student facial expressions, with potential to confirm current hypotheses across large-scale datasets.

The intelligent tutoring systems community has also begun integrating real-time facial expression tracking into studies of learning-centered affective states [5, 8]. These studies are a parallel line of research to that of understanding student affect. Incorporating nonverbal behavior tracking into intelligent tutoring systems is a necessary step toward meaningful real-time affective interventions. There has also been recent research that may lead to robust sensor-free affect detection [2]. Such an approach identifies patterns of behavior in log data that are associated with observed affective states. Then, models are built from the log data alone to predict affective states.

In prior research toward automated analysis of learning-centered affect, the creators of CERT applied the tool to video corpora taken during demanding tasks [18, 25]. Particularly, the facial expressions of children were investigated in order to compile a set of facial expressions relevant to the younger population [18]. These studies inform the use of automated facial expression recognition. A key difference in the present study is that we are presenting a comparatively much larger scale of analysis (over 80 times the duration of video). Additionally, we conducted a novel validation that compared values of CERT output with manual FACS annotations across a naturalistic tutoring video corpus.

3. TUTORING VIDEO CORPUS

The corpus consists of computer-mediated tutorial dialogue for introductory computer science collected during the 2011-2012 academic year. Students (*N*=67) and tutors interacted through a web-based interface that provided learning tasks, an interface for computer programming, and textual dialogue. The participants were university students in the United States, with average age of 18.5 years (*stdev*=1.5). The students voluntarily participated for course credit in an introductory engineering course, but no prior computer science knowledge was assumed or required. Each student was paired with a tutor for a total of six sessions on different days, limited to forty minutes each session. Recordings of the sessions included database logs, webcam video, skin conductance, and Kinect depth video. This study analyzes the webcam video corpus. The student workstation configuration is shown in Figure 1. The JavaTutor interface is shown on the next page in Figure 3.

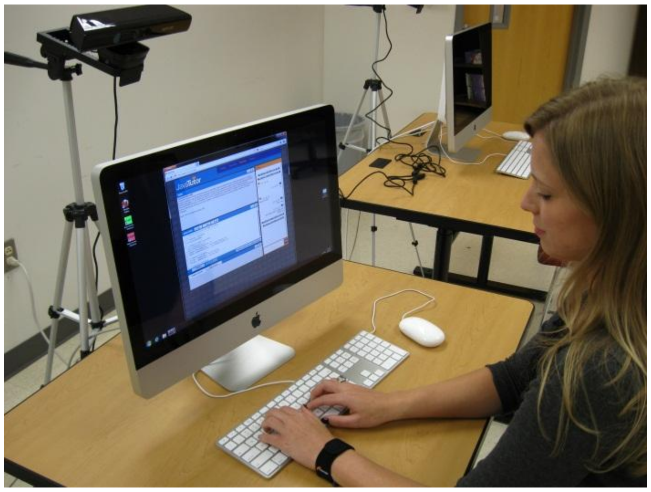


Figure 1. Student workstation with depth camera, skin conductance bracelet, and computer with webcam

Before each session, students completed a content-based pretest. After each session, students answered a post-session survey and posttest (identical to the pretest). The post-session survey items were designed to measure several aspects of engagement and cognitive load. The survey was composed of a modified User Engagement Survey (UES) [20] with Focused Attention, Endurability, and Involvement subscales, and the NASA-TLX workload survey [16], which consisted of response items for Mental Demand, Physical Demand, Temporal Demand, Performance, Effort, and Frustration Level. Student survey items relevant to the results presented in Section 4 are shown in Figure 2. Students were intentionally not asked about a wider set of emotions in order to avoid biasing their future interactions.

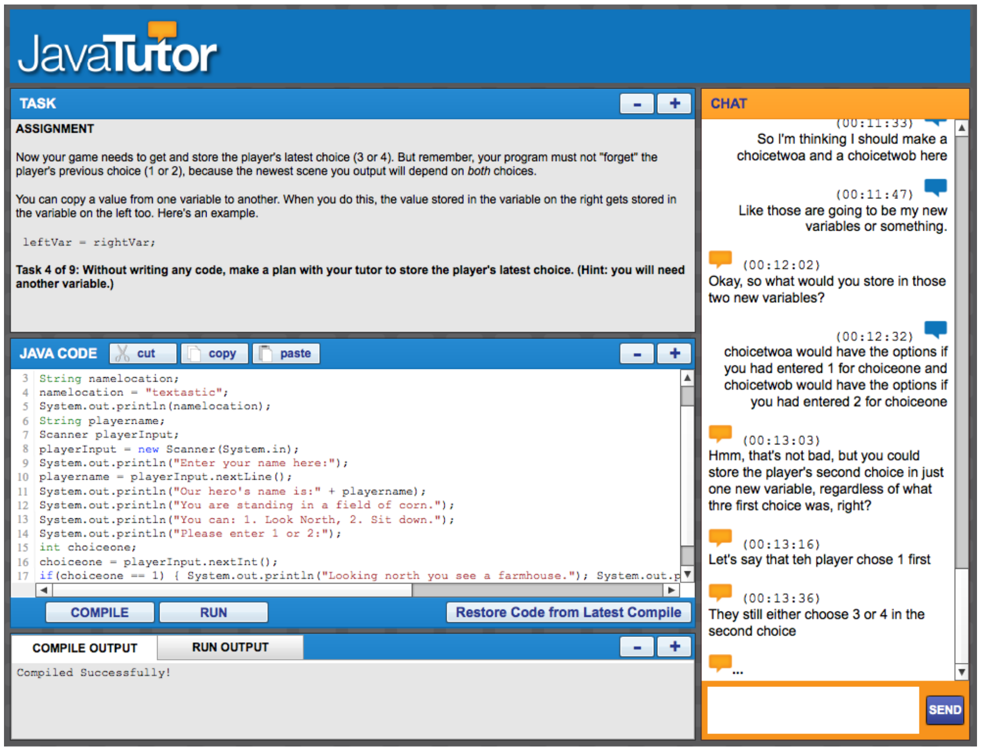
Endurability (UES):
- Working on this task was worthwhile.
- I consider my learning experience a success.
- My learning experience was rewarding.
- I would recommend using JavaTutor to my friends and family.

Temporal Demand (NASA-TLX):
- How hurried or rushed was the pace of the task?

Performance (NASA-TLX):
- How successful were you in accomplishing what you were asked to do?

Frustration Level (NASA-TLX):
- How insecure, discouraged, irritated, stressed, and annoyed were you?

Figure 2. Subset of student post-session survey items



ASSIGNMENT

Now your game needs to get and store the player's latest choice (3 or 4). But remember, your program must not "forget" the player's previous choice (1 or 2), because the newest scene you output will depend on *both* choices.

You can copy a value from one variable to another. When you do this, the value stored in the variable on the right gets stored in the variable on the left too. Here's an example.

```
leftVar = rightVar;
```

Task 4 of 9: Without writing any code, make a plan with your tutor to store the player's latest choice. (Hint: you will need another variable.)

```java
String namelocation;
namelocation = "textastic";
System.out.println(namelocation);
String playername;
Scanner playerInput;
playerInput = new Scanner(System.in);
System.out.println("Enter your name here:");
playername = playerInput.nextLine();
System.out.println("Our hero's name is:" + playername);
System.out.println("You are standing in a field of corn.");
System.out.println("You can: 1. Look North, 2. Sit down.");
System.out.println("Please enter 1 or 2:");
int choiceone;
choiceone = playerInput.nextInt();
if(choiceone == 1) { System.out.println("Looking north you see a farmhouse."); System.out.p
```

```
Compiled Successfully!
```

(00:11:33)
So I'm thinking I should make a choicetwoa and a choicetwob here

(00:11:47)
Like those are going to be my new variables or something.

(00:12:02)
Okay, so what would you store in those two new variables?

(00:12:32)
choicetwoa would have the options if you had entered 1 for choiceone and choicetwob would have the options if you had entered 2 for choiceone

(00:13:03)
Hmm, that's not bad, but you could store the player's second choice in just one new variable, regardless of what thre first choice was, right?

(00:13:16)
Let's say that teh player chose 1 first

(00:13:36)
They still either choose 3 or 4 in the second choice

...

Figure 3. The JavaTutor interface

The tutoring video corpus is comprised of approximately four million video frames totaling thirty-seven hours across the first tutoring session. Two session recordings were missing due to human error (*N*=65). The recordings were taken at 640x480 pixel resolution and thirty frames per second. CERT successfully tracked faces across a great majority of the tutoring video corpus (*mean*=83% of frames tracked, *median*=94%, *stdev*=23%).

3.1 Facial Expression Recognition

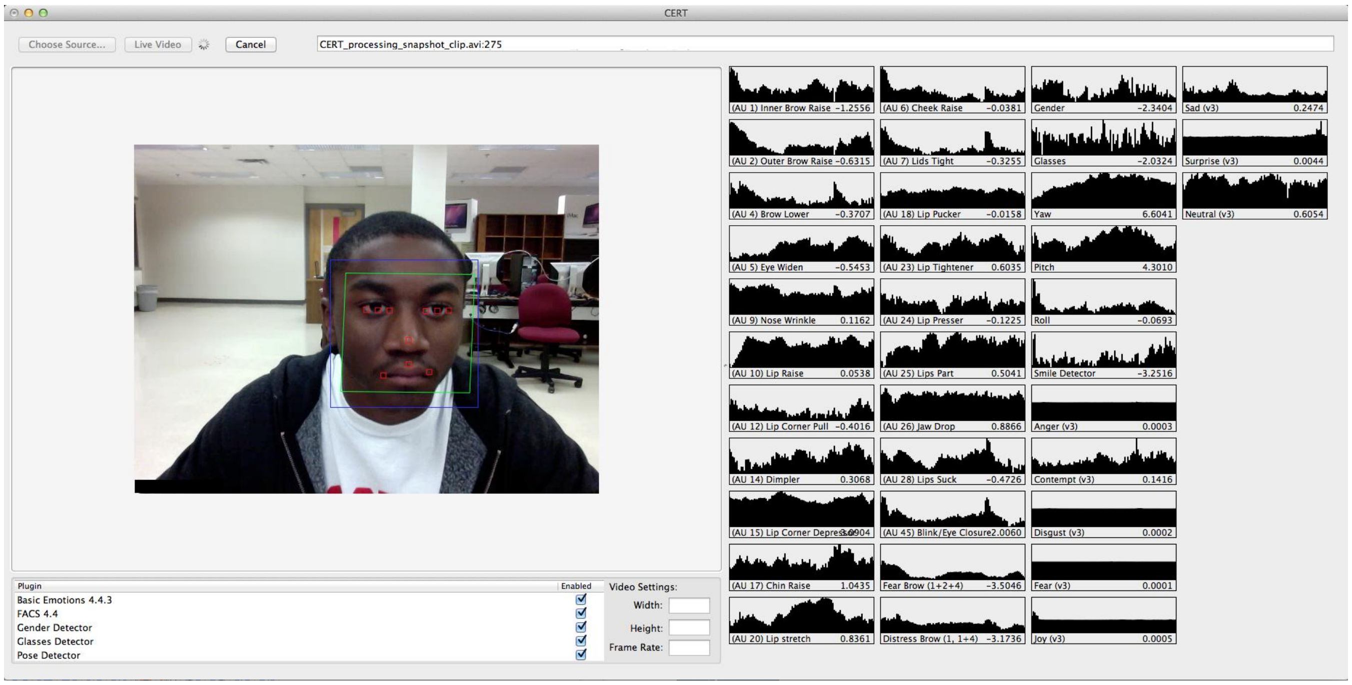
The Computer Expression Recognition Toolbox (CERT) [19] was used in this study because it allows frame-by-frame tracking of a wide variety of facial action units. CERT finds faces in a video frame, locates facial features for the nearest face, and outputs weights for each tracked facial action unit using support vector machines. For a detailed description of the technology used in CERT, see [26].

Based on observations from prior studies [12, 13], we selected a subset of the 20 facial action units that CERT detects as the focus of the present analyses. This set of facial action units was informed by a prior naturalistic tutoring video corpus [13], used in this study as a validation set, consisting of approximately 650,000 FACS-annotated video frames and seven tutoring sessions. In this corpus, sixteen facial action units were annotated. The five most frequently occurring action units each occurred in over 10% of the facial expression events. The remaining facial action units occurred substantially less frequently. The five frequently occurring action units were selected for the further analysis presented here on the new corpus. Table 1 shows the relative frequency of each action unit's participation in discrete facial expression events and the number of frames annotated with each action unit from the validation corpus.

A screenshot of CERT processing is shown in Figure 4. In the course of processing videos with CERT, we noted that the range of output values can vary between individuals due to their hair, complexion, or wearing eyeglasses or hats. This has also been noted by the creators of CERT [26]. In order to better capture instances of facial expression displays, we introduce an adjustment procedure for individual tracking differences. First, the average output value for each student was computed for each action unit. These values correspond to individual baselines of facial expression. The average output value per session was subtracted for each action unit, resulting in individually adjusted CERT output. This adjustment was applied to all CERT values presented in this paper. Automatically recognized instances of the selected action units are shown in Figure 5, with corresponding adjusted CERT output. While any positive output value indicates that CERT recognizes an action unit, we used an empirically determined threshold of 0.25 to reduce the potential for false positives. This threshold was based on observations of CERT output in which action unit instances that were more than slightly visible corresponded with output values above 0.25. CERT successfully tracked faces across a large majority of the validation corpus (*mean*=76% of frames tracked, *median*=87%, *stdev*=23%).

Table 1. The five most frequent facial action units in the validation corpus [13]

Facial action unit	Frames	Event Freq.
AU1: Inner Brow Raiser	12,257	15.5%
AU2: Outer Brow Raiser	15,183	21.7%
AU4: Brow Lowerer	127,510	18.6%
AU7: Lid Tightener	9,474	13.2%
AU14: Dimpler	14,462	24.2%

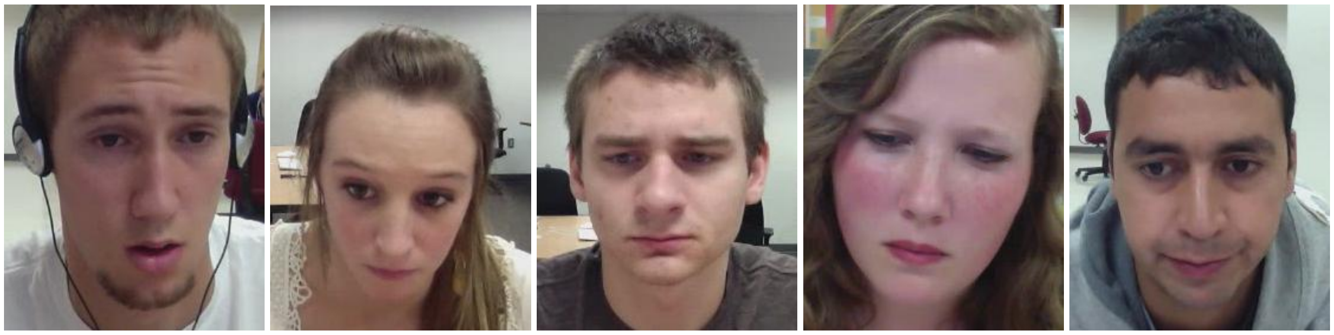


CERT_processing_snapshot_clip.avi:275

Plugin	Enabled
Basic Emotions 4.4.3	✓
FACS 4.4	✓
Gender Detector	✓
Glasses Detector	✓
Pose Detector	✓

```
(AU 1) Inner Brow Raise  -1.2556
(AU 2) Outer Brow Raise -0.6315
(AU 4) Brow Lower  -0.3707
(AU 5) Eye Widen  -0.5453
(AU 9) Nose Wrinkle  0.1162
(AU 10) Lip Raise  0.0538
(AU 12) Lip Corner Pull  -0.4016
(AU 14) Dimpler  0.3068
(AU 15) Lip Corner Depress3.0904
(AU 17) Chin Raise  1.0435
(AU 20) Lip stretch  0.8361
(AU 6) Cheek Raise  -0.0381
(AU 7) Lids Tight  -0.3255
(AU 18) Lip Pucker  -0.0158
(AU 23) Lip Tightener  0.6035
(AU 24) Lip Presser  -0.1225
(AU 25) Lips Part  0.5041
(AU 26) Jaw Drop  0.8866
(AU 28) Lips Suck  -0.4726
(AU 45) Blink/Eye Closure2.0060
Fear Brow (1+2+4)  -3.5046
Distress Brow (1, 1+4)  -3.1736
Gender  -2.3404
Glasses  -2.0324
Yaw  6.6041
Pitch  4.3010
Roll  -0.0693
Smile Detector  -3.2516
Anger (v3)  0.0003
Contempt (v3)  0.1416
Disgust (v3)  0.0002
Fear (v3)  0.0001
Joy (v3)  0.0005
Sad (v3)  0.2474
Surprise (v3)  0.0044
Neutral (v3)  0.6054
```

Figure 4. Screenshot of CERT video processing



AU1(0.80) AU2(-0.12) **AU4(0.25)** AU7(-0.23) AU14(-0.06)	AU1(0.17) **AU2(0.27)** AU4(0.08) AU7(-0.09) AU14(-0.53)	AU1(-0.02) AU2(0.07) **AU4(0.47)** AU7(0.08) AU14(-0.85)	AU1(-0.22) AU2(-0.27) AU4(0.11) **AU7(0.26)** AU14(-0.04)	AU1(-0.02) AU2(0.00) AU4(-0.04) AU7(0.03) **AU14(0.46)**
AU1 and AU4: Inner brow raiser and brow lowerer	**AU2: Outer brow raiser**	**AU4: Brow lowerer**	**AU7: Lid tightener**	**AU14: Dimpler**

Figure 5. Automatically recognized facial action units (bold values are above selected threshold of 0.25)

3.2 Validation

CERT was developed using thousands of posed and spontaneous facial expression examples of adults outside of the tutoring domain. However, naturalistic tutoring data often has special considerations, such as a diverse demographic, background noise within a classroom or school setting, no controls for participant clothing or hair, and facial occlusion from a wide array of hand-to-face gesture movements. Therefore, we aim to validate CERT's performance within the naturalistic tutoring domain. CERT's adjusted output was compared to manual annotations from a validation corpus, as described in Section 3.1.

The creators of CERT have applied the tool to the problem of understanding children's facial expressions during learning. To validate CERT's output, they compared it with manual FACS annotations across 200 video frames [18]. However, the goal in this analysis is to validate CERT's performance across a validation corpus of approximately 650,000 video frames. It is important to know whether average CERT output values for video frames with a specific facial movement are different from those without that facial movement. If the values are differentiable, then CERT may be an appropriate tool for general use at a large scale. If the values cannot be distinguished, then CERT is likely to provide many false positives and false negatives. Thus, this novel validation analysis provides needed insight into how well CERT performs across an entire corpus. The design of the validation analysis is shown in Figure 6.

Adjust CERT Output: Adjusting CERT output values with a baseline for each individual allows for comparison across students

↓

Binary Split on AU: Output values for each student are divided between AU-present and AU-not-present and then averaged; these values serve as the input variable for logistic regression

↓

Build Predictive Model: A logistic regression model is built from all students' average values for AU-present and AU-not-present, producing predicted categories of AU/Not-AU

↓

Compare Predictions: Compute Cohen's Kappa and accuracy from the logistic regression model predictions and manual tags to evaluate how well AU/Not-AU was predicted

Figure 6. Design of the validation analysis

Adjusted CERT output was computed for each video frame as described in Section 3.1. The CERT output values were then averaged within five binary splits, one for each facial action unit under consideration. Each binary split was comprised of frames with a specific facial action present and frames without that particular action unit, as labeled in the validation corpus. For example, to evaluate performance on brow lowering (AU4), video frames were divided between presence or absence of AU4 via the manual annotations. Once the binary split was performed, the frames were further subdivided by student. Thus, each student has an average value for frames with a specific action unit present and an average value for frames without that action unit. Logistic regression models were constructed using the average value as the sole parameter. One logistic regression model was built per action unit, for a total of five. The binary response variable categories (action unit present/action unit absent) were produced from each regression model. The predicted categories were compared to the categories from manual annotation, yielding Cohen's κ and percent accuracy.

The validation results show that CERT output has an excellent capability to distinguish facial expression events from baseline across the validation corpus, yielding an average κ across the five action units of 0.82. Naturalistic data is challenging for computer vision techniques, so the validation analysis confirms the accuracy of CERT facial expression recognition. Table 2 displays the validation results.

Table 2. Comparison of agreement on validation corpus [13] Manual FACS vs. logistic regression of CERT output

FACS Coder	AU1	AU2	AU4	AU7	AU14
Manual κ^*	0.88	0.82	0.79	0.78	0.73
CERT κ^*	0.86	0.86	0.68	1	0.71
CERT Accuracy*	93%	93%	85%	100%	86%

*Manual κ on face events; CERT evaluated on avg. output

In order to explore the effectiveness of the correction for individual differences described in Section 3.1, the validation analysis was performed again, this time without corrected output values. With raw CERT output, the logistic regression models could not distinguish between the average values for AU-present versus AU-not-present (Table 3). Thus, agreement with the manual annotations was poor. The validation analyses illustrate that CERT output should be corrected with average values if a comparison across individuals is desired. This correction is straightforward to apply in post-processing. In a real-time application of such a tool, a running average could be computed at each video frame.

Table 3. Secondary validation analysis on raw CERT output

	AU1	AU2	AU4	AU7	AU14
CERT κ	0.14	0.29	0.05	0.29	0.29
CERT Accuracy	57%	64%	54%	64%	64%

A difficulty that remains for facial expression recognition is face occlusion, where the face is covered by an object, hand, etc. One source of face occlusions is hand-to-face gestures [14], where one or two hands touch the lower face. These gestures are particularly prominent in our tutoring video corpus, as students often place a hand to their face while thinking or cradle their head in both hands while apparently tired or bored. These gestures can result in loss of face tracking or incorrect output. Accordingly, our analyses considered only video frames where face tracking and registration were successful (i.e., where CERT produced facial action unit output). Examples of both types of occlusion errors are shown in Figure 7. The CERT adjusted output values for the mostly occluded face frame (in the left image) are [AU1 = 1.34, AU2 = 0.65, AU4 = 0.62, AU7 = 0.30, AU14 = -1.17]. If these values are interpreted with the 0.25 threshold, then they represent presence of multiple action units, but that is clearly not the case when viewing the video. CERT was unable to find the student's face in the partially occluded frame (in the right image), though the presence of brow lowering is apparent. While hand-to-face gestures present a significant complication in naturalistic tutoring data, there has been preliminary progress toward automatically detecting these gestures [14], so their effect may be mitigated in future facial expression tracking research.

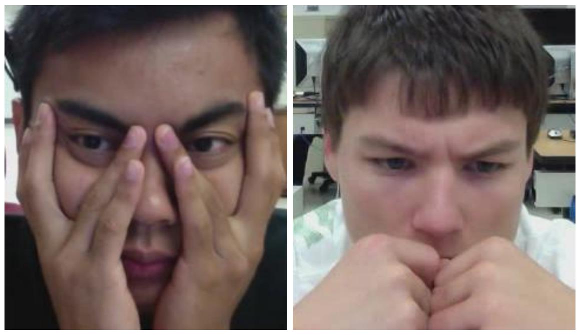


Figure 7. Facial recognition errors due to gestures: mostly occluded (left) and partially occluded (right)

4. PREDICTIVE MODELS

Automated facial expression recognition enables fine-grained analyses of facial movements across an entire video corpus.

With such tracking, there is potential to discover previously unidentified ways in which both frequency [7] and intensity [18] of facial expressions inform diagnosis of student affective states. A first step toward this possibility is to quantify facial expressions as they occurred throughout tutoring and compare these with tutorial outcomes. Therefore, predictive models of both affective and learning outcomes were built leveraging both the average intensity and frequency of facial movements. Refer to Figure 5 for example images of the facial action units.

Predictive models were constructed using minimum Bayesian Information Criterion (BIC) in forward stepwise linear regression, using JMP statistical software. These models are conservative in how they select predictive features because the explanatory value of added parameters must offset the BIC penalty for model complexity. Tutoring outcomes (affective and learning) were the dependent variables. Therefore, a model was constructed to predict each of the post-session survey scales and normalized learning gain (ten in total). The models for which facial action unit features were significantly explanatory are described below.

4.1 Facial Action Units and Affective Outcomes

Endurability was the student's self-report of whether he or she found the tutoring session to be worthwhile and whether he or she would recommend JavaTutor tutoring to others. Endurability was predicted by inner brow raising (AU1) intensity and brow lowering (AU4) intensity. AU1 was a positive predictor, while AU4 was negative. After adjusting for degrees of freedom (i.e., the number of model parameters), the model effect size was *r* = 0.37. The model is shown in Table 4.

Table 4. Stepwise linear regression model for Endurability

Endurability =	**Partial R^2**	**Model R^2**	***p***
-10.58 * *AU4_Intensity*	0.088	0.088	0.004
6.60 * *AU1_Intensity*	0.075	0.162	0.023
16.61 (intercept)			<0.001
RMSE = 10.01% of range in Endurability scale			

Temporal demand captures the student's self-report of whether he or she felt rushed or hurried during the session. Temporal demand was negatively predicted by outer brow raising (AU2) frequency; that is, students with higher frequency of this action unit reported feeling more rushed during the session. The adjusted model effect size was *r* = 0.23. The model is shown in Table 5.

Table 5. Stepwise linear regression model for Temporal Demand

Temporal Demand =	**Partial R^2**	**Model R^2**	***p***
-103.15 * *AU2_Freq*	0.068	0.068	0.037
34.90 (intercept)			<0.001
RMSE = 19.69% of range in Temporal Demand scale			

Performance was the student's self-report of how successful he or she felt in accomplishing the task. Performance was positively predicted by frequency of mouth dimpling (AU14), so students who displayed AU14 more frequently reported a higher sense of performance. The adjusted model effect size was *r* = 0.26. The model is shown in Table 6.

Table 6. Stepwise linear regression model for Performance

Performance =	**Partial R^2**	**Model R^2**	***p***
64.65 * *AU14_Freq*	0.081	0.081	0.022
72.74 (intercept)			<0.001
RMSE = 8.50% of range in Performance scale			

Frustration was the student's self-report of how insecure, agitated or upset he or she was during the tutoring session. Frustration was positively predicted by intensity of brow lowering (AU4); that is, students who displayed more intense AU4 reported feeling more insecure, agitated, or upset. The adjusted model effect size was *r* = 0.29. The model is shown in Table 7.

Table 7. Stepwise linear regression model for Frustration

Frustration =	**Partial R^2**	**Model R^2**	***p***
77.27 * *AU4_Intensity*	0.098	0.098	0.011
-15.34 (intercept)			0.165
RMSE = 17.05% of range in Frustration scale			

4.2 Facial Action Units and Learning Gain

We considered whether facial movements predicted learning gains. Normalized learning gain was computed using the following formula if posttest score was greater than pretest score:

$$NLG = \frac{Posttest - Pretest}{1 - Pretest}$$

Otherwise, normalized learning gain was computed as follows:

$$NLG = \frac{Posttest - Pretest}{Pretest}$$

Normalized learning gain was predicted by outer brow raising (AU2) intensity and mouth dimpling (AU14) frequency. AU2 was a negative predictor and AU14 was a positive predictor; that is, lower AU2 intensity corresponded to lower learning gain, while greater AU14 frequency corresponded to higher learning gain. The adjusted model effect size was *r* = 0.43. The model is shown in Table 8.

Table 8. Stepwise linear regression model for Normalized Learning Gain

Norm. Learn Gain =	**Partial R^2**	**Model R^2**	***p***
-2.29 * *AU2_Intensity*	0.145	0.145	<0.001
2.13 * *AU14_Freq*	0.064	0.208	0.031
0.73 (intercept)			0.053
RMSE = 29.49% of range in Normalized Learning Gain			

5. DISCUSSION

The results highlight that specific facial movements predict tutoring outcomes of engagement, frustration, and learning. Particular patterns emerged for almost all of the facial action units analyzed. We discuss each of the results in turn along with the insight they provide into mechanisms of engagement, frustration, and learning as predicted by facial expression.

Average intensity of brow lowering (AU4) was associated with negative outcomes, such as increased frustration and reduced desire to attend future tutoring sessions. Brow lowering (AU4) has been correlated with confusion in prior research [7, 9] and interpreted as a thoughtful state in other research [12, 18]. Here, the average intensity of brow lowering is found to be a positive predictor of student frustration and a negative predictor of students finding the tutoring session worthwhile. It may be that the tutor and student were unable to overcome student confusion, resulting in frustration instead of deep learning [9]. This interpretation is compatible with the theory of cognitive disequilibrium, which maps possible transitions from confusion to deep learning when a new concept is successfully acquired or to frustration when the concept cannot be reconciled with the student's present understanding. It is also possible that in some cases, AU4 displays represent an angry or agitated affective state. AU4 is a key component of the prototypical display of anger [11]. Further study that accounts for student progress through the programming task may reveal whether there is a significant cognitive aspect to this result.

Average intensity of inner brow raising (AU1) was positively associated with students finding the tutoring session worthwhile. At first glance, this finding seems to be in marked contrast to prior research that implicated both inner and outer brow raising as components of frustration displays [7]. However, intensity of the facial expressions was not considered in the prior work. AU1 is also a component of prototypical expressions of surprise or sadness [11]. From among these possible affective states—frustration, sadness, and surprise—surprise may be most likely to explain higher ratings of endurability. Students may have found the tutoring session to be surprising because it was a first exposure to computer programming. Surprise displays were observed while processing the videos through CERT and there were numerous such displays in the validation corpus. However, further study is required to disambiguate this result.

Lower frequency of outer brow raising (AU2) predicted a lesser sense of being hurried or rushed; in contrast, greater intensity of displays of AU2 predicted reduced learning gains. Outer brow raising (AU2) has been associated with frustration in prior research [7]. As frustrated students may not achieve high learning gains, the intensity of AU2 may be indicative of frustration. However, AU2 was not predictive of students' self-reported frustration levels, so this may be capturing a subtly different phenomenon. An alternative interpretation comes from research into facial expressions of anxiety, in which "fear brow" facial movements were found to occur more often during anxiety [15]. The prototypical "fear brow" includes AU1, AU2, and AU4 present in combination [11]. An example of this facial expression is shown as AU2 in Figure 5. Greater anxiety during tutoring may result in feeling rushed or hurried and may also negatively impact learning. Thus, anxiety is consistent with the results for AU2. However, the other action units expected in facial expressions of anxiety, AU1 and AU4, did not have the same results. This is likely due to the conflicting nature of brow raising and brow lowering, as the CERT values for AU1 and AU4 may be reduced during their combined movement in the "fear brow" (see Figure 5). Further analyses of combined facial movements would provide insight into this complication of automated facial expression recognition.

Frequency of mouth dimpling (AU14) predicted increased student self-reports of task success, as well as increased learning gains. There have not been conclusive associations of mouth dimpling (AU14) and learning-centered emotions. However, this action unit has been implicated as being involved in expressions of frustration [7] and concentration [18]. In this study, frequency of AU14 was positively predictive of both self-reported performance and normalized learning gains. While the effect appears to be fairly subtle (effect size below 0.3 for both), it appears to be a display of concentration. This leads to the interesting question of whether AU4 or AU14 better represents a thoughtful, contemplative state. Further research in this vein may resolve the question.

While eyelid tightening (AU7) was not added to any of the predictive models, there appear to be reasons for this. Observation of CERT processing and the results of the validation analysis indicate a way to adjust CERT's output of AU7, enabling refined study of the action unit. AU7 is an important facial movement to include, as it has been correlated with confusion [7]. Our proposed method for correcting AU7 output was informed by observing that CERT tends to confuse AU7 with blinking or eyelid closing. In prior manual annotation efforts, we explicitly labeled AU7 only when eyelid movements tightened the orbital region of the eye (as in the FACS manual). Thus, manual annotation seems more effective due to this complication of eye movements. However, note that CERT's AU7 output perfectly agreed with manual annotations in our validation analysis. Thus, CERT clearly tracks eyelid movements well. The problem may be that CERT's AU7 output is overly sensitive to other eyelid movements. One way to mitigate this problem may be to subtract other eye-related movements from instances of AU7. For instance, if AU7 is detected, but CERT also recognizes that the eyelids are closed, the detected AU7 event could be discarded.

The results demonstrated predictive value not only for frequency of facial movements, but also intensity. The relationship between facial expression intensity and learning-centered affect is unknown, but perhaps action unit intensity is indicative of higher-arousal internal affective states. Additionally, it is possible that intensity will inform disambiguation between learning-centered affective states that may involve similar action units (e.g., confusion/frustration and anxiety/frustration). Lastly, intensity of facial movements may be able to aid diagnosis of low arousal affective states. For instance, a model of low intensity facial movements may be predictive of boredom, which current facial expression models have difficulty identifying.

6. CONCLUSION

This paper presented an automated facial recognition approach to analyzing student facial movements during tutoring using the Computer Expression Recognition Toolbox (CERT), which tracks a wide array of well-defined facial movements from the Facial Action Coding System (FACS). CERT output was validated by comparing its output values with manual FACS annotations, achieving excellent agreement despite the challenges imposed by naturalistic tutoring video. Predictive models were then built to examine the relationship between intensity and frequency of facial movements and tutoring session outcomes. The predictive models highlighted relationships between facial expression and aspects of engagement, frustration, and learning.

This novel approach of fine-grained, corpus-wide analysis of facial expressions has great potential for educational data mining. The validation analysis confirmed that CERT excels at tracking specific facial movements throughout tutoring sessions. Future studies should examine the phenomena of facial expression and learning in more detail. Temporal characteristics of facial expression can also be examined, such as how rapidly an expression appears and how quickly it vanishes. Additionally, with these results in hand, it will be important to conduct an analysis of the broader set of facial action units tracked by CERT to build a comprehensive understanding of the interplay between learning and affect.

ACKNOWLEDGMENTS

This work is supported in part by the North Carolina State University Department of Computer Science and the National Science Foundation through Grant DRL-1007962 and the STARS Alliance Grant CNS-1042468. Any opinions, findings, conclusions, or recommendations expressed in this report are those of the participants, and do not necessarily represent the official views, opinions, or policy of the National Science Foundation.

REFERENCES

[1] Baker, R.S.J. d., D'Mello, S.K., Rodrigo, M.M.T. and Graesser, A.C. 2010. 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.

[2] Baker, R.S.J. d., Gowda, S.M., Wixon, M., Kalka, J., Wagner, A.Z., Salvi, A., Aleven, V., Kusbit, G.W., Ocumpaugh, J. and Rossi, L. 2012. Towards Sensor-Free Affect Detection in Cognitive Tutor Algebra. *Proceedings of the 5th International Conference on Educational Data Mining*, 126–133.

[3] Baltrusaitis, T., Robinson, P. and Morency, L.-P. 2012. 3D Constrained Local Model for Rigid and Non-Rigid Facial Tracking. *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2610–2617.

[4] Calvo, R.A. and D'Mello, S.K. 2010. Affect Detection: An Interdisciplinary Review of Models, Methods, and Their Applications. *IEEE Transactions on Affective Computing*. 1, 1, 18–37.

[5] Cooper, D.G., Muldner, K., Arroyo, I., Woolf, B.P. and Burleson, W. 2010. Ranking Feature Sets for Emotion Models used in Classroom Based Intelligent Tutoring Systems. *Proceedings of the 18th International Conference on User Modeling, Adaptation, and Personalization*, 135–146.

[6] D'Mello, S.K. and Calvo, R.A. 2011. Significant Accomplishments, New Challenges, and New Perspectives. *New Perspectives on Affect and Learning Technologies*. R.A. Calvo and S.K. D'Mello, eds. Springer. 255–271.

[7] D'Mello, S.K., Craig, S.D. and Graesser, A.C. 2009. Multi-Method Assessment of Affective Experience and Expression during Deep Learning. *International Journal of Learning Technology*. 4, 3/4, 165–187.

[8] D'Mello, S.K. and Graesser, A. 2010. Multimodal Semi-automated Affect Detection From Conversational Cues, Gross Body Language, and Facial Features. *User Modeling and User-Adapted Interaction*. 20, 2, 147–187.

[9] D'Mello, S.K., Lehman, B., Pekrun, R. and Graesser, A.C. Confusion Can Be Beneficial for Learning. *Learning & Instruction*. (in press)

[10] Ekman, P. and Friesen, W. V. 1978. *Facial Action Coding System*. Consulting Psychologists Press.

[11] Ekman, P., Friesen, W. V. and Hager, J.C. 2002. *Facial Action Coding System: Investigator's Guide*. A Human Face.

[12] Grafsgaard, J.F., Boyer, K.E. and Lester, J.C. 2011. Predicting Facial Indicators of Confusion with Hidden Markov Models. *Proceedings of the 4th International Conference on Affective Computing and Intelligent Interaction*, 97–106.

[13] Grafsgaard, J.F., Boyer, K.E. and Lester, J.C. 2012. Toward a Machine Learning Framework for Understanding Affective Tutorial Interaction. *Proceedings of the 11th International Conference on Intelligent Tutoring Systems*, 52–58.

[14] Grafsgaard, J.F., Fulton, R.M., Boyer, K.E., Wiebe, E.N. and Lester, J.C. 2012. Multimodal Analysis of the Implicit Affective Channel in Computer-Mediated Textual Communication. *Proceedings of the 14th ACM International Conference on Multimodal Interaction*, 145–152.

[15] Harrigan, J.A. and O'Connell, D.M. 1996. How Do You Look When Feeling Anxious? Facial Displays of Anxiety. *Personality and Individual Differences*. 21, 2, 205–212.

[16] Hart, S.G. and Staveland, L.E. 1988. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. *Human Mental Workload*. P.A. Hancock and N. Meshkati, eds. Elsevier Science. 139–183.

[17] Kaliouby, R. and Robinson, P. 2005. Generalization of a Vision-Based Computational Model of Mind-Reading. *Proceedings of the First International Conference on Affective Computing and Intelligent Interfaces*, 582–589.

[18] Littlewort, G., Bartlett, M.S., Salamanca, L.P. and Reilly, J. 2011. Automated Measurement of Children's Facial Expressions during Problem Solving Tasks. *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, 30–35.

[19] Littlewort, G., Whitehill, J., Wu, T., Fasel, I., Frank, M., Movellan, J. and Bartlett, M. 2011. The Computer Expression Recognition Toolbox (CERT). *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*, 298–305.

[20] O'Brien, H.L. and Toms, E.G. 2010. The Development and Evaluation of a Survey to Measure User Engagement. *Journal of the American Society for Information Science and Technology*. 61, 1, 50–69.

[21] Picard, R.W., Papert, S., Bender, W., Blumberg, B., Breazeal, C., Cavallo, D., Machover, T., Resnick, M., Roy, D. and Strohecker, C. 2004. Affective Learning — A Manifesto. *BT Technology Journal*. 22, 4, 253–269.

[22] Rodrigo, M.M.T. and Baker, R.S.J.d. 2011. Comparing Learners' Affect while using an Intelligent Tutor and an Educational Game. *Research and Practice in Technology Enhanced Learning*. 6, 1, 43–66.

[23] Ruf, T., Ernst, A. and Kublbeck, C. 2011. Face Detection with the Sophisticated High-speed Object Recognition Engine (SHORE). *Microelectronic Systems*. 243–252.

[24] den Uyl, M.J. and van Kuilenburg, H. 2008. The FaceReader: Online Facial Expression Recognition. *Proceedings of Measuring Behavior 2005*, 589–590.

[25] Whitehill, J., Bartlett, M. and Movellan, J. 2008. Automatic Facial Expression Recognition for Intelligent Tutoring Systems. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 1–6.

[26] Wu, T., Butko, N.J., Ruvolo, P., Whitehill, J., Bartlett, M.S. and Movellan, J.R. 2012. Multi-Layer Architectures for Facial Action Unit Recognition. *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics*, 42, 4, 1027-1038.

[27] Zeng, Z., Pantic, M., Roisman, G.I. and Huang, T.S. 2009. A Survey of Affect Recognition Methods: Audio, Visual, and Spontaneous Expressions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. 31, 1, 39–58.

Investigating the Solution Space of an Open-Ended Educational Game Using Conceptual Feature Extraction

Erik Harpstead, Christopher J. MacLellan, Kenneth R. Koedinger,
Vincent Aleven, Steven P. Dow, Brad A. Myers
Human-Computer Interaction Institute
Carnegie Mellon University
Pittsburgh, PA
{eharpste, cmaclell, koedinger, aleven, spdow, bam}@cs.cmu.edu

ABSTRACT
The rich interaction space of many educational games presents a challenge for designers and researchers who strive to help players achieve specific learning outcomes. Giving players a large amount of freedom over how they perform a complex game task makes it difficult to anticipate what they will do. In order to address this issue designers must ask: what are students doing in my game? And does it embody what I intended them to learn? To answer these questions, designers need methods to expose the details of student play. We describe our approach for automatic extraction of conceptual features from logs of student play sessions within an open educational game utilizing a two-dimensional context-free grammar. We demonstrate how these features can be used to cluster student solutions in the educational game *RumbleBlocks*. Using these clusters, we explore the range of solutions and measure how many students use the designers' envisioned solution. Equipped with this information, designers and researchers can focus redesign efforts to areas in the game where discrepancies exist between the designers' intentions and player experiences.

Keywords
Educational Games, Representation Learning, Context-Free Grammars, Clustering

1. INTRODUCTION
Educational games are a growing sub-field of instructional technology. Researchers see video games as a compelling medium for instruction because they can offer students the ability to practice new skills within an authentic context that poses little personal risk [7]. These promising aspects of games have led many educational game designers to create "open games", which allow students to exercise creativity in how they solve problems. [12,25]. Open educational games are a form of exploratory learning environment and commonly use ill-defined problems as part of their designs [11,19]. While the tendency toward open experiences is compelling for educational game designers, it presents problems when analyzing student learning, a necessary part of designing activities to foster robust learning.

When designing an open game experience, the designer surrenders a degree of control over the nature and progression of the experience to the player [10]. This openness can be problematic to the designers of educational game experiences who are concerned that students receive some type of intended instruction and achieve a desired learning outcome. Educational game designers require a detailed picture of how students are playing a game in order to know if disparities exist between the designers' intentions and player experiences; and, if such disparities do exist, designers need to know where to focus redesign efforts.

To facilitate designers' and researchers' analysis of open educational games we propose a methodology for extracting conceptual features from student log data. We demonstrate our methodology in *RumbleBlocks*, an educational game designed to teach basic concepts of structural stability to young children [5]. The method takes as input logs of student gameplay and yields a set of conceptual features which describe student solutions. While some aspects of our approach are specific to *RumbleBlocks*, the general concept should be applicable to open educational games.

To automatically generate features in *RumbleBlocks*, we use a four-step process that converts the log data from student play into feature vectors. This process, which is the primary contribution of this paper, consists of discretizing the log data containing student solutions; generating a grammar from the discretized logs; using the grammar to parse each solution; and converting the resultant parse trees into vectors that concisely represent the structural components of the solutions. In the following sections, we first describe the game *RumbleBlocks* and then provide the details of the four-step process to extract features. Afterwards, we show the results of using the extracted features to cluster student solutions, which enables the identification of misalignment between designer intentions and student actions.

1.1 RumbleBlocks
RumbleBlocks is an educational game designed to teach basic structural stability and balance concepts to children in kindergarten through grade 3 (5-8 years old) [5]. It focuses primarily on three basic principles of stability: objects with wider bases are more stable, objects that are symmetrical are more stable, and objects with lower centers of mass are more stable. These principles are derived from the National Research Council's Framework for New Science Educational Standards [21] and other science education curricula for the target age group.

The game follows a sci-fi narrative where the player is helping a group of aliens who become stranded when their mother ship is damaged. Each level (see Figure 1 for an example level) consists of an alien stranded on a cliff with their deactivated space ship lying on the ground. The player must use an inventory of blocks to build a structure that is tall enough to reach the alien. In Figure 1, the player is dragging a third block (the highlighted square block) from the inventory (top left) to the tower-under-construction (bottom, center). Additionally, the player's structure must also cover a series of blue "energy balls" floating in space which are narratively used to power the space ship, but serve to both guide and constrain the players' designs. Once the student is confident in their design, they can place the spaceship on top of their tower triggering an earthquake that serves as a test of the tower's stability. If, at the end of the quake, the tower is still standing and the spaceship is still on top, the student passes the level and proceeds to the next level; otherwise they start the level over again.

Beyond the limits imposed by the energy ball mechanic and the types of available blocks, students are not very constrained in the

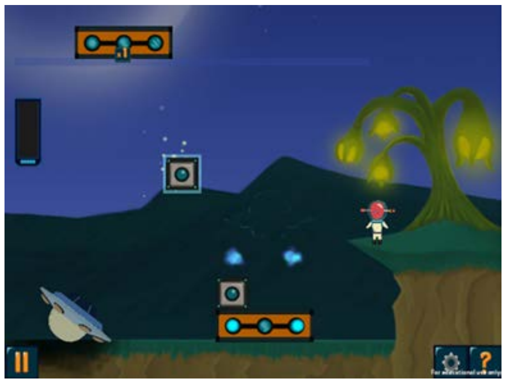


Figure 1. An example level from *RumbleBlocks*. The alien is stranded on the cliff and players must build a tower which is tall enough to reach him while also covering all blue "energy balls" to power his spaceship.

designs they can create. Each level in *RumbleBlocks* is designed to emphasize a particular principle of structural stability, and thus has a particular solution that was envisioned by the designers. However, students are not required to use it, and it is even possible that students may find a solution that is better than the one that the designer envisioned. Throughout development, the designers formed an intuition for the different groups of answer types being used by students, but they lacked methods for understanding how similar two answers were, and how many different answers were possible for each level. While it would have been possible to render all student solutions into screenshots, it would have been infeasible to manually comb through the thousands of towers generated by students.

To address this issue of understanding the kinds of solutions students are using, we have developed a method for extracting the conceptual features of game states in *RumbleBlocks* utilizing a two-dimensional context-free grammar. These features allow the designers and researchers of *RumbleBlocks* to examine the different sub-patterns that players are using in building their towers. The conceptual features can be used as a way of comparing different towers and evaluating how often students produce the answer which designers expected. It also enables us to zero in on the solutions they did not expect. To demonstrate the utility of these features we perform a clustering analysis, which assigns towers to groups, which correspond to the different unique solution that are possible on each level of the game. Designers can use this analysis to better understand the space of student solutions.

2. CONCEPTUAL FEATURE EXTRACTION

The first challenge in using *RumbleBlocks* data, or any educational game data, is to convert it into a form that is amenable to analysis. This task is not easy because a single state, or tower, in *RumbleBlocks* is both continuous and two-dimensional. Previous work has used an empirical measure of symmetry, width-of-base, and center-of-mass (human selected features) to describe a tower and has shown that these features can be predictive of student outcomes [9]. These features give a useful abstract evaluation of students' solutions; however, they are not descriptive enough to provide insight regarding specific patterns in student solutions. Without a more detailed description, it is hard for a designer to understand where new interventions need to be implemented to better facilitate student learning. In this work, we seek to remedy this problem by automatically extracting fine-grained conceptual features using unsupervised learning directly from two-dimensional descriptions of towers. These features allow us to investigate the solution space at a higher level of detail.

Our conceptual feature extraction process takes as input log files from all student play sessions and outputs all student towers as feature vectors that represent the towers' structural components. The process consists of the four steps illustrated in Figure 2 and discussed in turn in the next sections. First, we discretize the representations of all towers in the raw log files using a two-dimensional grid. Second, we generate grammatical rules based on the discretized representations using a novel algorithm of our own design: the Exhaustive Rule Generator (ERG), which induces a two-dimensional grammar returning an exhaustive set of rules capable of parsing the entire set. Third, the discrete representations are parsed using the rules generated by ERG, which returns a set of parse trees describing each tower in a hierarchical fashion. Finally, we process the parse trees to generate a set of feature vectors that denote which concepts from the grammar are present within each tower.

2.1 Discretization

The first step in the conceptual feature extraction process is discretization, or gathering meaningful data from the logs and converting it from a continuous two-dimensional space into a discrete two-dimensional space. The input to this step is the raw student log data, which contains action-by-action traces of student play sessions at replay fidelity. The logs generated by *RumbleBlocks* are intended to be post-processed through a replay analysis engine [9] which allows researchers to play logs back through an active instance of the game engine in order to extract information from live game states. Using this approach we are able to access information on individual game objects, such as collision information or bounding box dimensions, without having to log everything at the time of play. Since the logs are being replayed within the same game engine, the replayed game states are consistent with what students experienced.

To convert the continuous data from *RumbleBlocks* into discrete data we utilized a binning process. To bin a tower, the coordinates of the extents of each block's bounding box (the smallest rectangle which can be drawn around the block, a property accessible in the active game state) are translated such that the bottom left corner of the tower is at position (0,0). After translation, all of the edge coordinates of each block are divided by the size of the

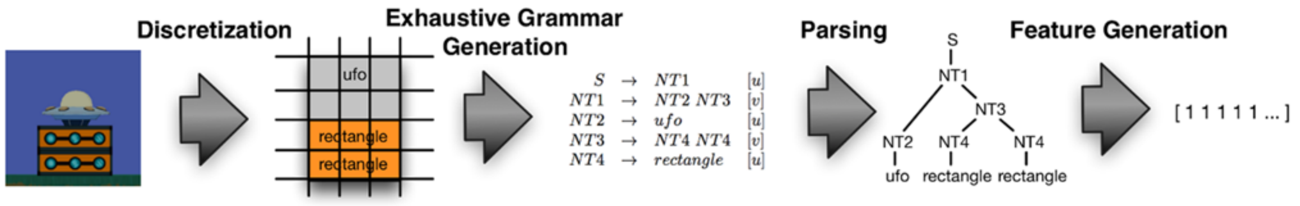


Figure 2. The Conceptual Feature Extraction Process.

smallest block (a square), creating a unit grid. Finally, the edges of blocks are rounded to their nearest integer positions (e.g., an x position of 1.6 would be rounded to 2), in effect “snapping” blocks to grid positions, which helps to ensure that clear divisions can be drawn between blocks because some blocks are slightly out of alignment. After binning, we output the discretized towers as a set of blocks described by their type and converted left, right, top and bottom values. The block type is the concatenation of the original block’s shape (cube, rectangle etc.) and its rotation about the z-axis rounded to the nearest 15 degrees (for example the “rectangle” block with a 90 degree angle would now have “rectangle90” as its type). Thus, the final tower is discrete and comprised of blocks binned to a unit grid.

2.2 Exhaustive Rule Generation (ERG)

Once all of the student log data has been converted into discretized towers, we can automatically generate features describing the spatial aspects of these towers using two-dimensional context-free grammars. These grammars, which have been used to perceive structure in pictures, are an extension of 1D grammars for strings [4]. The grammar used in our approach are simplification of probabilistic two-dimensional context-free grammars, which have been used in previous work to teach an artificial agent to learn to perceive tutor interfaces [16]. Our approach is slightly different than this previous work in that we do not need to choose a single best parse of a tower but instead want to extract all of the spatial features present in the tower. This makes the rule probabilities from [16] unnecessary and so we omit them. Additionally, the towers in the *RumbleBlocks* task are much more complicated than the grid layout of the tutoring system interfaces explored in the previous work. Despite these differences, the spirit of our work is the same. We are using context-free grammars to perform representation learning.

Before explaining how we automatically generate a grammar we give a description of how they are structured. A two-dimensional context-free grammar is represented by a 4-tuple *G* = <*S*,*V*,*E*,*R*>. *S* is the start symbol, which in our case represents the concept of a complete tower. *V* represents the set of nonterminal symbols, which represent the structural components of a tower. In the trivial case these nonterminals represent terminals, i.e. individual blocks or space, but more complicated nonterminals represent intermediate structures, e.g. a pair of blocks stacked on one another, or even entire towers. *E* is the set of terminal symbols, which in our task represent the blocks and filler space. Finally, *R* is the set of rules, which describe how nonterminal symbols can decompose into other terminal and nonterminal symbols, as well as the direction (vertical, horizontal, or unary) in which they decompose. Because our rules capture the relative positions between blocks and the spaces between them (vertically and horizontally adjacent), we do not need to store position information. To clarify, our rules have the following form:

NT → *BOTTOM TOP* [*vertical*]

NT → *LEFT RIGHT* [*horizontal*]

NT → *block* [*unary*]

Where *NT*, *BOTTOM*, *TOP*, *LEFT*, and *RIGHT* are nonterminal symbols, i.e. $\in V$, and *block* is a terminal symbol, i.e. $\in E$. The *vertical* rule can be used to parse the two structures, *BOTTOM* and *TOP*, into the *NT* structure if they are vertically adjacent, horizontally aligned (the values of their left extents and right extents are equal), have equal width, and if the *BOTTOM* structure is below the *TOP* structure. Similarly, the *horizontal* rule can be used to parse the two structures, *LEFT* and *RIGHT*, into the *NT* structure if they are horizontally adjacent, vertically aligned, have equal height, and if the *LEFT* structure is to the left of the *RIGHT* structure. Finally, the *unary* rule allows the *block* symbol to be parsed into *NT*; no additional constraints apply for unary rules. We utilize Chomsky Normal Form to represent our rules because it allows for polynomial time parsing using the CKY algorithm [6], so every nonterminal decomposes into a pair of nonterminals or a single terminal symbol. Note that for convenience, we also have unary start rules that point to nonterminals representing entire towers. Even though this is in violation of Chomsky Normal Form, we only have these special rules at the top-most level so it does not have an effect on parsing complexity. See Figure 3 for an example of how a grammar can be used to parse a tower.

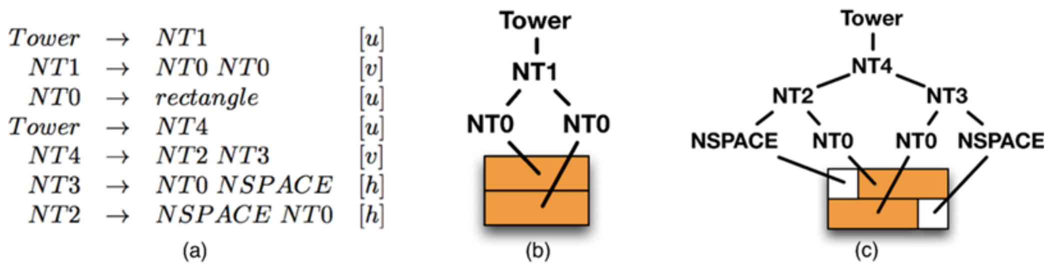
Before we can parse discretized *RumbleBlocks* towers we need to generate a grammar capable of parsing the set of towers. One difficulty is that most towers are not initially parsable because their blocks don’t align cleanly, which is needed for matching vertical and horizontal grammar rules. To deal with the problem that not all towers are completely rectangular in shape, we introduce a new ‘*space*’ terminal symbol that has unit size, i.e. takes up one grid cell, and fill in all of the negative space in a tower with these symbols.

While introducing ‘*space*’ symbols enables us to parse towers that have space in them, it also causes an additional problem. First, we plan on automatically generating new nonterminals for blocks that are adjacent to one another. Because there are so many ways to pair up ‘*space*’ symbols we end up bloating the grammar with unnecessary nonterminals that all reduce to space. Furthermore, this explosive number of nonterminal symbols also pair up with meaningful block symbols causing the grammar to grow even larger. To prevent grammar bloat we seed our initial grammar with the following recursive space rules:

NSPACE → *space* [*unary*]

NSPACE → *NSPACE NSPACE* [*vertical*]

NSPACE → *NSPACE NSPACE* [*horizontal*]



$$\begin{array}{rcll} Tower & \rightarrow & NT1 & [u] \\ NT1 & \rightarrow & NT0\ NT0 & [v] \\ NT0 & \rightarrow & rectangle & [u] \\ Tower & \rightarrow & NT4 & [u] \\ NT4 & \rightarrow & NT2\ NT3 & [v] \\ NT3 & \rightarrow & NT0\ NSPACE & [h] \\ NT2 & \rightarrow & NSPACE\ NT0 & [h] \end{array}$$

Figure 3. An example of how grammar (a) can be used to describe towers (b and c). The extra space and alignment rules of the grammar are omitted for clarity.

We also ensure that no additional nonterminals that reduce solely to space are introduced during automatic grammar generation.

Once we augment the towers with '*space*' symbols we use the novel Exhaustive Rule Generator (ERG) Algorithm (see Algorithm 1) to recursively generate a nonterminal for every pair of adjacent structures. The input to the algorithm is a set of towers and a start symbol. The algorithm starts by creating an empty grammar (seeded with recursive space rules), adds terminals for all of the blocks and the space symbol, creates an empty collection of remembered structures (used to ensure multiple nonterminals are not generated for the same structure), and iterates through the set of towers adding rules for each tower using the recursive *Rule-Gen* procedure.

The *Rule-Gen* procedure takes a single tower, a grammar, and a collection of remembered structures as inputs. It starts by checking if there is already a nonterminal that describes the tower, if such a nonterminal exists it is returned. Next, the algorithm checks if the tower only contains space, if so the algorithm returns the special *NSPACE* symbol (so any generated grammar integrates with the recursive space rules). If neither condition is met then a new nonterminal is generated with a unique name and added to the grammar. If the structure consists of a single terminal then a unary rule is added decomposing the new nonterminal into the terminal symbol. An entry in the hash table is created for that tower and the nonterminal is returned. If the structure contains more than one terminal, it is divided at each location (both horizontal and vertical) where the structure can be divided into two sub-structures (without splitting a terminal). For each division, the *Rule-Gen* procedure is called on the sub-structures and a rule is added mapping the new nonterminal to the nonterminals representing each sub-structure. The direction of this rule is determined by the direction of the division. After adding rules for all divisions, an entry is added to the collection mapping the structure to the new nonterminal and the nonterminal is returned.

Algorithm 1: EXHAUSTIVE-RULE-GENERATOR($Towers, S$)

```
procedure RULE-GEN($Structure, Grammar, Remembered$)
  if $Structure \in Remembered$
    then return ($Remembered[Structure]$)

  if IS-ONLY-SPACE($Structure$)
    then return ($'NSPACE'$)

  $Head \leftarrow$ UNIQUE-NONTERMINAL-NAME()
  ADD-NONTERMINAL($Head, Grammar$)

  if IS-TERMINAL($Structure, Grammar$)
    then { $Terminal \leftarrow$ GET-TERMINAL($Structure, Grammar$)
           ADD-RULE(($Head \rightarrow Terminal[unary]$), $Grammar$)
           $Remembered[Structure] \leftarrow Head$
           return ($Head$)

  for each ($SubA, SubB, Direction$) $\in$ ALL-DIVISIONS($Structure$)
    do { $NtA \leftarrow$ RULE-GEN($SubA, Grammar, Remembered$)
         $NtB \leftarrow$ RULE-GEN($SubB, Grammar, Remembered$)
         ADD-RULE($Grammar$, ($Head \rightarrow NtA, NtB[Direction]$))
  $Remembered[Structure] \leftarrow Head$
  return ($Head$)

main
  $Grammar \leftarrow$ EMPTY-GRAMMAR()
  ADD-TERMINALS($Towers, Grammar$)
  $Remembered \leftarrow$ EMPTY-COLLECTION()
  for $Tower \in Towers$
    do { $Nt \leftarrow$ RULE-GEN($Tower, Grammar, Remembered$)
         ADD-RULE(($S \rightarrow Nt[unary]$), $Grammar$)
         return ($Grammar$)
```

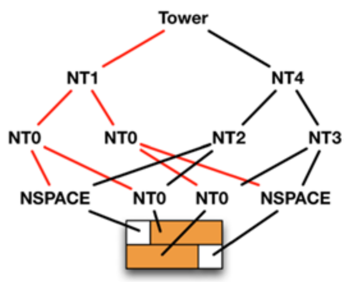


Figure 4. The two possible parses of tower *(c)* after alignment rules are added. Notice that the rules in the red tree are now similar to the rules in tower *(b)*'s parse tree.

The result of the ERG algorithm is a grammar that contains a nonterminal for every structure present in the set of towers. However, one subtle problem remains. Two towers that are nearly similar, but are unaligned and consequently have an additional '*space*' somewhere in the tower end up sharing no intermediate nonterminal symbols in their parses, see the differences between towers *(b)* and *(c)* in Figure 3. This is a problem because we are using nonterminals to model spatial features common across towers. To counter this effect, we introduce a set of "alignment rules" for every nonterminal *NT* in our grammar:

NT → *NT NSPACE* [*vertical*]

NT → *NSPACE NT* [*horizontal*]

NT → *NT NSPACE* [*horizontal*]

These rules triple the number of grammar rules, but add additional parses to towers so that they share common structure with other similar but differently aligned towers, see Figure 4. We have two horizontal rules so that we can have additional space on the left and right of a symbol, but we only have one vertical rule because we can have additional negative space on the top of a block, but not on the bottom, because blocks in *RumbleBlocks* are subject to gravity and any space below a block would be filled by the block falling into a new position. It is important to note that while these rules enable the towers to share similar structure, it does not give them identical parses. This enables us to relate similar structures using their parse trees without having to worry about truly different towers being lumped together.

2.3 Parsing

After generating a grammar, we can use it to parse the towers and determine all of the nonterminal symbols that can be derived from each tower. We use a modified version of the CKY algorithm [6] that functions over two dimensions instead of one. This algorithm, which utilizes dynamic programming, is an approach to bottom-up parsing in polynomial time. One feature of the CKY algorithm is that the amount of time required to compute all parses of a tower is the same as the amount of time required to compute one parse. Using this approach, we produce all of the parses for every tower in our set.

2.4 Feature Vector Generation

Once we have all of the parse trees, we convert them into feature vectors. This converted format is useful because the vector representation is more concise and easier to manipulate when doing analyses. To create a feature vector we create a one-dimensional vector with an integer value for every nonterminal in the grammar. These values are initialized to be 0 but are set to 1 for every nonterminal that appears in at least one of a given tower's parse trees, similar to previous work [17]. Thus, a feature vector is a concise description of all the structures that are present in all of the parses of a given tower. Once we have generated these feature vectors, we can use them to perform a variety of analyses as we will demonstrate next.

3. Data

The data we present here comes from a large formative evaluation of *RumbleBlocks*, which was performed in two local area elementary schools. The sample includes play sessions from 174 students from grades K-3 (5-8 years old) who played the game for a total of 40 min across 2 sessions. The game contained 39 different levels, each intended to target a specific principle of stability through the use of the energy balls as scaffolding. Players played an average of 17.8 unique levels (σ =7.2), as not all students completed the entire game. Additionally, because students are allowed to retry levels in which they fail, the data can contain multiple attempts by a student on each level (μ =1.24, σ =.68). In total, the dataset contains 6317 unique structures created by students.

Due to constraints of the conceptual feature extraction process some data had to be excluded from analysis. The parsing process requires that blocks be aligned to a grid such that clear separations can be drawn between them—because of this it was necessary to omit any structures where the binning process caused blocks to overlap the same grid cell (less than 0.2% of data). Additionally, rotating a block will sometimes cause its bounding box to intersect with adjacent grid cells, because the bounding box expands to encompass the maximum left, right, top, and bottom values of the block's geometry rather than rotating with it. To address these issues of grid overlap we exclude any record that contained blocks whose dimensions intersected or any blocks whose z-axis rotation was not a multiple of 90, after rounding to the nearest 15 degrees. Overall these constraints exclude ~3.5% of our sample.

The final grammar generated from the dataset by the ERG algorithm contains 13 terminals, 6,010 nonterminals, and 30,923 rules. Each nonterminal was used an average of 50.59 times (σ =240.2) across all towers. The average number of levels in which a given nonterminal was used was 3.09 (σ =4.14). The average number of nonterminals per towers was 49.96 (σ =40.23). Reporting statistics on the number of nonterminals within an average parse or number of parses within an average tower is complicated by the inclusion of alignment rules which add some arbitrary number of parses to each tower.

4. CLUSTER ANALYSIS

In order to demonstrate the utility of these conceptual features to guide the design process in educational games, we performed a clustering analysis of student solutions in *RumbleBlocks*, to discern how many solutions students were demonstrating. Clustering takes a series of data points, in our cases represented by conceptual feature vectors, and assigns them to groups based on how similar the points are. Clustering similar to ours has been used by Andersen and Liu et al. to group game states as a way of exploring common paths that players take through a game [18]. Our approach differs from theirs in that our features are machine learned rather than defined by designers. This allows us to observe emergent patterns in play without biasing the results with human input.

4.1 Method

As we were interested in what kinds of solutions students were using on each level, we performed clustering of solutions on a level-by-level basis, which will yield groups of similar student solutions. Within each level we utilized the k-means clustering algorithm (we use the scikit-learn implementation [22]). This algorithm takes as input a set of data and a parameter k, where each datum is described by an n-dimensional vector and k specifies the number of desired clusters. The output is a set of labels assigning each datum to a particular cluster. The algorithm works by using the k-means++ approach [3] to select initial centroids for the clusters such that they are generally distant from one another. This initialization algorithm guarantees that the solution found will be $O(log\ k)$ competitive to the optimal solution. Given the initial centroid positions, the data points are then assigned to the clusters based on which centroid they are nearest to, as measured by the Euclidian distance between the n-dimensional vectors of the point and the centroid. Once the points are assigned, the positions of the centroids are updated relative to the points they encompass. This process (also called hard expectation-maximization) is then repeated until quiescence. Although the worst-case running time is known to be super polynomial in the size of the input, in practice the algorithm finds solutions reasonably quickly [2]. For a given run of k-means we repeat this process 10 times and select the model that has the best fit to the data, which is measured by the within cluster sum squared distance from every point to its centroid. Running the algorithm multiple times helps to avoid local maximums and accounts for the inherent non-deterministic nature of the algorithm.

As we are also interested in how many solutions are present in the data, not just which solutions are similar, we therefore must determine the correct number of clusters to use, in essence choosing a good value for k. To identify the number of clusters present in the data, we use the G-means algorithm, which acts as a wrapper around the k-means algorithm [8]. This approach starts by running k-means on the entire dataset with k initialized to 1. The algorithm then takes the clusters of points returned by the previous k-means and attempts to divide each of them into two further sub-clusters, again using k-means with k=2. A vector is then drawn between the two new sub-clusters' centroids, which represents the dimension over which the two clusters are separated. The algorithm then projects all the points from both sub-clusters onto this single dimension of separation and checks to see if they have a Gaussian distribution using the Anderson-Darling statistic (with $p < 0.01$). If the distribution is found to not be Gaussian, the original value of k is incremented and the process is repeated for all clusters. Once all of the clusters are found to have a Gaussian distribution, the final k value is returned, representing a good number of groups in the dataset. This approach has been shown to be more effective than BIC at deciding the correct value for k [8]. Because k-means returns different clusters on different runs, we run the G-means algorithm 10 times and return the mode k value as the most likely value for k.

Before using the machine clustering to conduct analyses, we must first ensure that it is creating reasonable clusters. As a test of the validity of the clusters, we had two independent coders hand cluster three levels to generate a gold standard with which to compare the machine clustering results ($\kappa = 0.88$). Additionally, we want to evaluate the effectiveness of our approach by comparing it to a naïve method of automatic grouping. The naïve method we used was to group the towers by direct equability, i.e. assigning all towers that have identical discrete representations to the same group. This allows us to see how much closer our approach gets to human results than a naïve machine approach.

The selected three levels were chosen because they were part of an in-game counterbalanced pre-posttest, which did not use the

Table 1. Clustering measures (completeness, homogeneity, v-measure, and adjusted rand index) means and standard deviations after 10 iterations of clustering. Note that equality clustering is constant and so has no standard deviation.

Level	Comparison	Completeness (SD)	Homogeneity (SD)	V-Measure (SD)	Adj. Rand Index (SD)
com_11_noCheck (n=251)	k-means	.74 (.06)	.57 (.10)	.63 (.04)	.51 (.08)
	equality	.55 (NA)	.99 (NA)	.71 (NA)	.23 (NA)
s_13_noCheck (n=249)	k-means	.83 (.02)	.63 (.04)	.72 (.02)	.47 (.04)
	equality	.60 (NA)	.99 (NA)	.75 (NA)	.16 (NA)
wb_03_noCheck (n=254)	k-means	.63 (.02)	.80 (.02)	.71 (.02)	.42 (.02)
	equality	.53 (NA)	.99 (NA)	.69 (NA)	.28 (NA)

energy ball mechanic, making them less constrained and likely to have more variable answers, and therefore pose a greater challenge in terms of accurate clustering. Additionally, because all students were required to play them as part of the pre-post design, these levels have some of the largest sample sizes of all levels.

In comparing different clusterings we report the completeness, homogeneity, V-Measure [24], and Adjusted Rand Index (ARI) [23] on these three levels for machine clustering and direct equality using human clustering as a gold standard. These measures each evaluate different aspects of clustering results and are standard metrics of clustering quality. The Completeness score measures how well records in the same class are clustered together, i.e., how well the clustering put items that should be together in the same group. The Homogeneity score measures how well records that are different are separated, i.e. when the elements within a given cluster are all the same. Because these measures are in opposition to each other, we report the V-measure, which gives a harmonic balance between the completeness and homogeneity scores. Finally, ARI is a measure of clustering accuracy adjusted for chance. The measure has a range of [-1, 1] and approaches 0 when guessing.

After testing the clustering on a subset of hand-coded levels, we also wanted to gauge the validity of the approach on all levels. To measure validity we make the assumption that if two towers are highly similar they are also likely to both stand or fall in the earthquake, though some noise is to be expected due to indeterminacies in the game's physics engine. Taking this assumption, we can again use homogeneity as a way of calculating how consistent the success/failure designation is within a cluster. Comparing the homogeneity scores of the machine clustering and the random clustering of the towers (using the same number of clusters as determined by G-means) can tell us if the machine clustering is significantly better than that expected by chance. This metric can be interpreted as a sanity check to ensure that the clustering is actually working on levels that have not been hand labeled.

After evaluating clustering validity, we can use clustering to get a sense of how often players are using designer envisioned solutions. In designing the levels, the game designers tried to make each level focus on one of the three targeted principles of stability (low center of mass, wide base, symmetry). That is, the designers' intention is that on each level, the configuration of the energy dots and the block inventory, are such that the student is led to a solution that exemplifies the particular principle targeted at that level. It is fine, and probably desirable, if levels allows for multiple (and unforeseen) solutions. However, what we hope to avoid is levels that have a large number of unforeseen solutions that do *not* address the particular principle that the level is intended to target.

To perform this alignment analysis we had one of the designers of *RumbleBlocks* generate a play session log that represented the "answer key" for each level. We then determined which of the clusters the intended solution would be grouped into on each level and compared the number of towers in that group to the total number of towers for that level. This information can help us get a sense of the alignment between what designer expectant students to do and what players actually do. Having this information can help the designers know where to focus future redesign efforts to best target discrepancies.

4.2 Results

When looking at the measures of clustering effectiveness in Table 1 we see that the k-means algorithm was able to outperform straight equality grouping in ARI and completeness. This can be interpreted to mean that k-means clustering is making a higher percentage of correct decisions in grouping structures, suggesting that the results of clustering can be validly used in further analysis. In all instances, the equality grouping performs better than k-means clustering in homogeneity score because if direct equality is used to assign group labels the resulting groups will be, by definition, perfectly homogeneous. In many instances, this causes the V-measure to also be better because V-measure evenly weights for completeness and homogeneity. Overall these results can be interpreted to mean that clustering along conceptual features of towers provides reasonable grouping accuracy when compared to human clustering.

When clustering was performed across all levels, the mean homogeneity of the k-means clusters was found to be significantly greater than the homogeneity from random grouping of student solutions using a two-sample t-test ($p < .001$). Assuming that similar towers would stand or fall together, this further supports the idea that the clustering algorithm is not separating similar student solutions.

Overall the clustering algorithm generated an average of 8 clusters per level ($\sigma = 3.98$), compared to the average number of groups as determined by equality grouping 56 ($\sigma = 45.77$). The smallest number of clusters (2) was seen in the tutorial level, which contains only 1 block and the spaceship allowing for very little difference between solutions. The highest number of clusters (17) was found in a later level (centerOfMass_07) which contains 5 larger blocks and 6 energy balls allowing for nuanced differences in solution styles.

Our analysis of what percentage of solutions appear similar to the designers' intended solutions shows a high degree of variability, see Figure 5. Some levels, like the tutorial and other earlier levels, are found near the higher end of the spectrum because as introductory levels they do not allow for a large number of solutions. However, the levels on the lower end of the spectrum indicate that few students actually created the towers envisioned by the designers. These levels warrant a closer investigation to ascertain what other kinds of solutions students are producing. For example, upon further inspection of the solutions to centerOfMass_07, designed to target the principle of low center of mass, we discovered

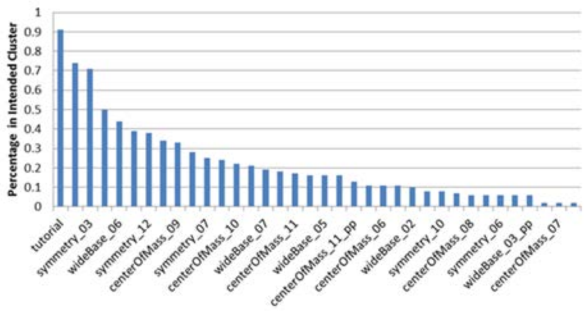


Figure 5. Percentage of use of the envisioned solution on a level for each level.

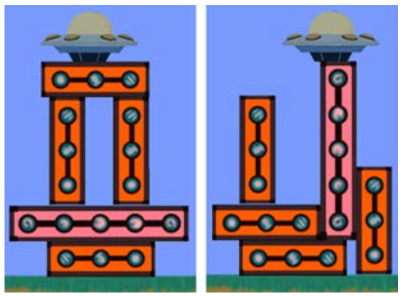


Figure 6. An example of mismatch with designer expectation and student solution from the centerOfMass_07 level. The designer's answer is on the left.

that a large number of student solutions that did not typify the level's key principle (See Figure 6). While a number of these solutions did not actually survive the earthquake the variety of atypical solutions points to the need for more guidance. In further iterations designers should focus their efforts on these levels to consider whether students need more scaffolding.

5. DISCUSSION

In this paper we have described a process for conceptual feature extraction from logs of gameplay in an educational game. The process follows four steps starting with the raw student log files. The files are discretized and then used to generate a two-dimensional context-free grammar that can be used to parse the towers and yield a vector of features present in the tower. We demonstrated how conceptual features could be used to perform a clustering analysis of common student solutions.

While the results we discussed are specific to *RumbleBlocks* aspects of our approach could be generalized to other games or educational technology environments by altering some of the steps in the process. One example of another game this approach would work for is *Refraction,* which has players redirecting laser beams around a grid based board by placing laser splitters to make proper fractions [1,18]. This game already takes place on a grid and so would not require a discretization step, but the other steps would be applicable. In this game, our approach would learn features corresponding to patterns of laser splitters on the grid, which could be used to generate feature vectors for each student solution and to cluster these feature vectors. These clusters would be similar to those generated by Liu *et al.* [18] but the features would be automatically generated rather than human tagged.

When applying our approach more generally, the discretization step will always be specific to a particular game or interface, as it requires an intimate knowledge of the context. Employing a replay analysis engine can assist with discretization by providing a standard format [9]. The ERG algorithm is applicable to any discrete two-dimensional representation of structure in which adjacency relations are meaningful. Converting parses into feature vectors for analysis is a technique that should be applicable to most situations.

The features generated with this method can be used by many different kinds of analyses beyond what we present here. For instance, the feature vectors could be used as a way to represent game data in a format suitable for DataShop [13], a large open repository of educational technology interaction data. A feature vector is analogous to the state of a tutoring system interface and the changes in the feature vector from step to step correspond to the student actions. Additionally, virtual agents, such as SimStudent [20], could use this data representation as a way of understanding and interacting with educational games, enabling us to model student learning in these contexts.

While the grammars extracted by our method have proven to be useful, they still have some limitations, such as an inability to represent towers that cannot be cleanly mapped to a grid or which contain overlapping or angled substructures. Making the grammar more descriptive would require the relaxing of constraints concerning how nonterminals can be parsed, e.g., not requiring strict alignment. Another issue has to do with how many different nonterminals map to nearly equivalent structures. Even though we attempt to minimize this by introducing the alignment and space rules, there are still cases where further reductions could be implemented. One potential solution, to address this problem in general, is to implement model merging to condense pairs of nonterminals that represent similar concepts into single nonterminals [15]. The ability to merge similar nonterminals is a promising direction for future work.

In addition to being able to describe more towers, model merging would also allow the generalization of grammars to cases we have not seen. Because context-free grammars can be used generatively, the generalized grammar could be used to produce novel towers, similar to the work of Talton *et al.* [26]. In our case, these novel towers would give insight into the as-yet-unseen portions of the solution space. Furthermore, the novel towers could be used as templates in creating new levels. In future work we will be exploring ways to feed this information, and information from clustering, directly back into the game development environment.

The clustering results not only provide the designers of *RumbleBlocks* with a picture of how students are playing their game, they also possess further uses beyond assisting design iteration, such as exploring research questions. One potential use of the clustering is as an empirical measure of how "open" a particular level is, by counting how many different clusters, i.e. different solutions, that level affords. Using this measure allows researchers to explore the interactions of openness with learning and engagement. Exploring this interpretation of the clustering results will be a part of our ongoing analysis of *RumbleBlocks.*

Another intriguing direction for future work would be to explore the relationship between the conceptual features and the knowledge components [14] used in building towers in *RumbleBlocks*. There may exist a mapping between the substructures used in towers and the conceptual knowledge components related to stable structures. Exploring this would require measurements of how a student's use of particular structures changed over time and how it relates to task performance. If such a mapping exists, then our approach would not only be useful for automated feature extraction, but also for automatically building models of conceptual knowledge components.

6. CONCLUSION

Framing game experiences in terms of conceptual features can help both designers and researchers better understand how students interact with their games. The main contribution of this paper is an approach for extracting conceptual features from play logs within educational games and using these features to perform clustering of student solutions. Designers can use the clusterings to better understand the space of student solutions and to know where to focus their attention to improve student learning experiences. Ultimately we envision feeding back this clustering information directly into the game design platform. This information can also enable researchers to explore important questions, such as how "openness" and difficulty relate to student engagement. While our approach was created with the specific two-dimensional world of *RumbleBlocks* in mind, it should be generalizable, and we hope others will find it useful in exploring other educational games.

7. ACKNOWLEDGMENTS

We would like to thank the designers of *RumbleBlocks* and our colleagues who conducted the formative evaluation that yielded our data. This work was supported in part by a Graduate Training Grant awarded to Carnegie Mellon University by the Department of Education #R305B090023 and the DARPA ENGAGE research program under ONR Contract Number N00014-12-C-0284.

8. REFERENCES

[1] Andersen, E., Liu, Y., Apter, E., Boucher-genesse, F., and Popovi, Z. Gameplay Analysis through State Projection. *Proc. FDG '10*, (2010).

[2] Arthur, D. and Vassilvitskii, S. How slow is the k -means method? *Proc. SCG '06*, ACM Press (2006), 144.

[3] Arthur, D. and Vassilvitskii, S. K-means++: The Advantages of Careful Seeding. *Proc. ACM-SIAM*, (2007), 1027–1035.

[4] Cherubini, A. and Pradella, M. Picture Languages: From Wang Tiles to 2D Grammars. In S. Bozapalidis and G. Rahonis, eds., *Algebraic Informatics*. Springer, Berlin, Germany, 2009, 13–46.

[5] Christel, M.G., Stevens, S.M., Maher, B.S., et al. RumbleBlocks: Teaching Science Concepts to Young Children through a Unity Game. *Proc. CGames 2012*, (2012), 162–166.

[6] Cocke, J. *Programming Languages and their Compilers: Preliminary Notes*. New York University, 1969.

[7] Gee, J.P. *What video games have to teach us about learning and literacy*. Palgrave Macmillan, New York, 2003.

[8] Hamerly, G. and Elkan, C. Learning the k in k-means. *Proc. NIPS '03*, (2003).

[9] Harpstead, E., Myers, B.A., and Aleven, V. In Search of Learning : Facilitating Data Analysis in Educational Games. *Proc. CHI '13*, (2013), 79–88.

[10] Hunicke, R., Leblanc, M., and Zubek, R. MDA : A Formal Approach to Game Design and Game Research. *Proc. of the AAAI Workshop on Challenges in Game AI*, (2004), 1–5.

[11] De Jong, T. and Van Joolingen, W.R. Scientific Discovery Learning with Computer Simulations of Conceptual Domains. *Review of Educational Research 68*, 2 (1998), 179–201.

[12] Ketelhut, D.J. The Impact of Student Self-efficacy on Scientific Inquiry Skills: An Exploratory Investigation in River City, a Multi-user Virtual Environment. *Journal of Science Education and Technology 16*, 1 (2006), 99–111.

[13] Koedinger, K.R., Baker, R.S.J. d, Cunningham, K., Skogsholm, A., Leber, B., and Stamper, J. A Data Repository for the EDM community: The PSLC DataShop. In C. Romero, S. Ventura, M. Pechenizkiy and R.S.J. d. Baker, eds., *Handbook of Educational Data Mining*. 2010, 43–55.

[14] Koedinger, K.R., Corbett, A.T., and Perfetti, C. The Knowledge-Learning-Instruction Framework: Bridging the Science-Practice Chasm to Enhance Robust Student Learning. *Cognitive Science 36*, 5 (2012), 757–98.

[15] Langley, P. *Simplicity and Representation Change in Grammar Induction*. 1995.

[16] Li, N., Cohen, W.W., and Koedinger, K.R. Learning to Perceive Two-Dimensional Displays Using Probabilistic Grammars. *LNCS 7524*, (2012), 773–788.

[17] Li, N., Schreiber, A., Cohen, W.W., and Koedinger, K.R. Creating Features from a Learned Grammar in a Simulated Student. *Proc. ECAI '12*, (2012).

[18] Liu, Y., Andersen, E., Snider, R., Cooper, S., and Popovi, Z. Feature-Based Projections for Effective Playtrace Analysis. *Proc. FDG '11*, (2011), 69–76.

[19] Lynch, C., Ashley, K.D., Pinkwart, N., and Aleven, V. Concepts , Structures , and Goals : Redefining Ill-Definedness. *International Journal of Artificial Intelligence in Education 19*, (2009), 253–266.

[20] Matsuda, N., Cohen, W.W., Sewall, J., Lacerda, G., and Koedinger, K.R. Evaluating a Simulated Student Using a Real Student's Data for Training and Testing. *LNAI 4511*, (2007), 107–116.

[21] National Research Council. *A Framework for K-12 Science Education: Practices, Crosscutting Concepts, and Core Ideas*. The National Academies Press, 2012.

[22] Pedregosa, F., Varoquaux, G., Gramfort, A., et al. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research 12*, (2011), 2825–2830.

[23] Rand, W. Objective Criteria for the Evaluation of Clustering Methods. *Journal of American Statistical Association 66*, 336 (1971), 846–850.

[24] Rosenburg, A. and Hirschber, J. V-Measure: A Conditional Entropy-based External Cluster Evaluation Measure. *Proc. EMNLP-CoNLL '07*, (2007), 410–420.

[25] Spring, F. and Pellegrino, J.W. The Challenge of Assessing Learning in Open Games : HORTUS as a Case Study. *Proc. GLS 8.0*, (2011), 200–208.

[26] Talton, J., Yang, L., Kumar, R., Lim, M., Goodman, N., and Měch, R. Learning design patterns with bayesian grammar induction. *Proc. UIST '12*, (2012), 63–74.

Extending the Assistance Model: Analyzing the Use of Assistance over Time

William Hawkins, Neil Heffernan, Yutao Wang
Department of Computer Science
Worcester Polytechnic Institute
100 Institute Road, Worcester, MA
{bhawk90, nth, yutaowang}@wpi.edu

Ryan S.J.d. Baker
Department of Human Development
Teachers College Columbia University
New York, NY
baker2@exchange.tc.columbia.edu

ABSTRACT
In the field of educational data mining, there are competing methods for predicting student performance. One involves building complex models, such as Bayesian networks with Knowledge Tracing (KT), or using logistic regression with Performance Factors Analysis (PFA). However, Wang and Heffernan showed that a raw data approach can be applied successfully to educational data mining with their results from what they called the Assistance Model (AM), which takes the number of attempts and hints required to answer the previous question correctly into account, which KT and PFA ignore. We extend their work by introducing a general framework for using raw data to predict student performance, and explore a new way of making predictions within this framework, called the Assistance Progress Model (APM). APM makes predictions based on the relationship between the assistance used on the two previous problems. KT, AM and APM are evaluated and compared to one another, as are multiple methods of ensembling them together. Finally, we discuss the importance of reporting multiple accuracy measures when evaluating student models.

Keywords
Student Modeling, Knowledge Tracing, Educational Data Mining, Assistance Model, Assistance Progress Model

1. INTRODUCTION
Understanding and modeling student behavior is important for intelligent tutoring systems (ITS) to provide assistance to students and help them learn. For nearly two decades, Knowledge Tracing (KT) [5] and various extensions to it [12, 16, 18] have been used to model student knowledge as a latent using Bayesian networks, as well as to predict student performance. Other models used to predict student performance include Performance Factors Analysis (PFA) [14] and Item Response Theory [8]. However, these models do not take assistance information into account. In most systems, questions in which hints are requested are marked as wrong, and students are usually required to answer a question correctly before moving on to the next one. Therefore, the number of hints and attempts used by a student to answer a question correctly is likely valuable information.

Previous work has shown that using assistance information helps predict scores on the Massachusetts Comprehensive Assessment Systems math test [6], can help predict learning gains [1], and can be more predictive than binary performance [17]. Recently, it has been shown that using simple probabilities derived from the data based on the amount of assistance used, an approach called the Assistance Model (AM), can improve predictions of performance when ensembled with KT [15].

This work continues research in the area of using assistance information to help predict performance in three ways:

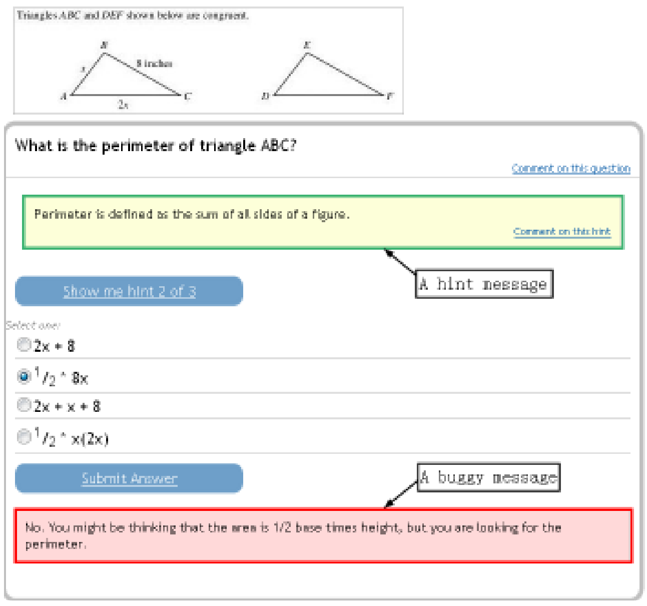
1. Specifying a framework for building "tabling methods" from the data, a generalization of AM
2. Experimenting with a new model within this framework called the Assistance Progress Model (APM), which makes predictions based on the relationship between the assistance used on the previous two problems
3. Experimenting with new ways of ensembling these models to achieve better predictions

Additionally, the importance of reporting multiple accuracy measures when evaluating student models is discussed, as well as why three of the most commonly reported measures (mean absolute error (MAE), root mean squared error (RMSE) and area under the ROC curve (AUC)) do not always agree on which model makes the most accurate predictions.

Section 2 describes the tutoring system and dataset used. Section 3 describes the methodology: the models and ensembling methods used, the tabling method framework, and the procedure for evaluating the models. Section 4 presents the results, followed by discussion and possible directions for future work in Section 5.

2. DATA
The data used here was the same used in [15], which introduced AM. This dataset comes from ASSISTments, a freely available web-based tutoring system for 4th through 10th grade mathematics.

While working on a problem within ASSISTments, a student can receive assistance in two ways: by requesting a hint, or by entering an incorrect answer, as shown in Figure 1.



Triangles ABC and DEF shown below are congruent.

What is the perimeter of triangle ABC?

Perimeter is defined as the sum of all sides of a figure.

- 2x + 8
- [✓ $^1/_2$ * 8x]
- 2x + x + 8
- $^1/_2$ * x(2x)

No. You might be thinking that the area is 1/2 base times height, but you are looking for the perimeter.

Figure 1. Examples of assistance within ASSISTments (from Wang and Heffernan, 2011)

The dataset comes from four Mastery Learning classes conducted in 2009, where students worked on problem sets until achieving some criterion, usually specified as answering three questions in a row correctly. The questions in these problem sets were generated randomly from templates, with the difficulty of each question assumed to be the same as all other questions generated from the same template. No problem selection algorithm was used to select the next question.

Two hundred 12-14 year old 8th grade students participated in these classes, generating 17,776 problem logs from 93 problem sets. However, due to the nature of the models studied in this paper, data from two of these students could not be used since these two students never answered more than one question within the same problem set.

Since two of the models cannot be used to predict performance on the first question of a problem set, as they rely on assistance usage on previous problems, these models were not trained or evaluated on the first question answered by a student on a given problem set. This reduced the dataset for these models to 12,099 problem logs. KT models were still trained using the entire dataset, but only evaluated on the 12,099 logs they had in common with the other models.

3. METHODS

This section begins by giving an overview of KT, then introduces a framework for building data-driven student models called "tabling methods," and describes two such methods: AM and APM. Next, the approaches used to ensemble these individual models together are briefly discussed. Finally, the procedure and measures used to evaluate all models are discussed.

3.1 Knowledge Tracing

KT is a well-studied student model introduced in [5] that keeps track over time of the probability that a student has mastered a given skill, given their past performance as evidence. The probability that a skill for a given student is in the "known" (vs. the "unknown") state can then be used to predict future performance.

Constructing KT models involves learning four parameters:

1. Initial Knowledge (L_0) – the probability the student has mastered the skill before attempting the first question
2. Learn Rate (T) – the probability the student will have mastered the skill after attempting a given question if they have not mastered the skill already, independent of performance
3. Guess Rate (G) – the probability the student will answer correctly despite not having mastered the skill
4. Slip Rate (S) – the probability the student will answer incorrectly despite having mastered the skill

KT models can be represented as static, "unrolled" Bayesian networks, as shown in Figure 2. The level of knowledge K_m at time step m influences performance on question Q_m. Initial knowledge influences K_0, while knowledge at time step m-1 influences knowledge at time step m for $m > 0$. The learned T, G and S parameters are the same across all practice opportunities, meaning that the conditional probability tables (CPTs) for all nodes K_m where $m > 0$ have the same values, and the CPTs for all Q nodes have the same values.

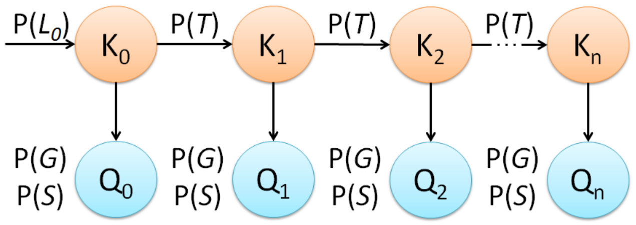


Figure 2. Static Bayesian network representation of Knowledge Tracing

In this work, the Bayes Net Toolbox for Matlab [9] is used to create separate KT models for each problem set. The parameters for each model are learned using Expectation-Maximization, with initial values of 0.3 for L_0, 0.09 for T, 0.1 for G and 0.09 for S.

3.2 Tabling Methods

In previous work [15], a data-driven approach called AM was used to predict performance based on the number of attempts and hints used on the previous problem. This was done by creating a table of probabilities of the student answering the next question correctly on the first attempt without any hints, indexed by the number of attempts and hints used on the previous problem. These probabilities were computed simply by computing the percentage of questions answered correctly on the first attempt with no hints, parameterized by the number of attempts and hints used on the previous problem.

Then, unseen test data was predicted by using the number of attempts and hints used on the previous problem to do a table lookup. The corresponding probability of getting the next question correct in the table was assigned as the prediction.

In this work, we present a generalization of this approach that serves as a framework for data-driven approaches for student modeling. The general procedure is as follows:

1. Create a table based on one or more attributes of the training data.

2. Compute the probability of answering a question correctly for each combination of values of the attributes selected in Step 1, and insert these probabilities into the proper cells in the table.
3. For each previously unseen test case, do a table lookup based on the attributes of the test case to obtain the probability (over the training data) of the student answering the question correctly.
4. Assign the retrieved probability as the prediction for the test case.

The attributes selected in Step 1 can be anything available or computable from the data, such as the number of hints and attempts used on the previous problem as AM does, or the correctness of the previous problem, the time taken, the type of skill, etc. These attributes could also represent which bin an instance falls into, where bins are constructed by splitting up students and/or problems based on some criteria.

Cells may need to be added to the table when values for one or more of the attributes are not available, depending on the nature of the attributes. If there are not enough data points for certain cells, it may help to simply combine them with others. Finally, depending on the nature of the selected attributes and the data, it may be useful to split certain cells based on some criterion.

In this work, two data-driven approaches that follow this framework are explored: the Assistance Model, as described by Heffernan and Wang and further described below, and the Assistance Progress Model (APM), which constructs a table based on the relationships between hints and attempts used on the previous two problems.

3.2.1 Assistance Model

As described previously, AM consists of a table of probabilities of a student answering a question correctly based on the number of attempts and the percentage of available hints used on the previous problem of the same skill. Attempts are broken into three bins: 1, (1, 6] and (6, ∞), while the percentage of hints is broken into four: 0, (0, 50], (50, 100) and 100. The AM table constructed from the entire dataset is shown in Table 1.

Table 1. AM table for entire dataset

		Attempts		
Hint %		**1**	**(1, 6]**	**(6, ∞)**
	0	0.778	0.594	0.480
	(0, 50]	0.560	0.623	0.444
	(50, 100)	0.328	0.461	0.444
	100	0.264	0.348	0.374

For instance, according to Table 1, when students answered correctly on the first attempt with no hints, they answered the next question correctly 77.8% of the time. On the other hand, if they required over six attempts and used all of the hints available, they answered the next question correctly only 37.4% of the time.

According to Table 1, when attempts are held constant, the general trend is that as hint usage increases, the probability that the student will answer the next question correctly decreases. This makes sense since hints are more likely to be used by students with lower knowledge of the skill.

When hints are held constant, different patterns occur with respect to the number of attempts used. When no hints are used, the probability of answering the next question correctly decreases as the number of attempts increases. This relationship is reversed when all hints are used. Finally, if just some of the hints are used, making a few attempts (between 2 and 6, inclusive) helps more than making one attempt, but making many attempts (> 6) decreases the probability of answering the next question correctly.

The pattern for no hints can be explained as more attempts required being indicative of lower student knowledge. For all hints being used, more attempts may indicate the student is attempting to learn rather than just requesting hints until the answer is given to them. Using some of the hints suggests the student has not mastered the skill, but has some knowledge of it and is attempting to learn. The relationship between making one attempt and making a few attempts can be explained by the more attempts the student makes, the more they learn, to a point. The use of excessive amounts of attempts probably indicates the student is not learning, despite using some of the hints.

The highest probability in the table, 0.778, corresponds to the case where the previous question was answered correctly. This is unsurprising since in this case, the student likely has mastered the skill. The lowest probability, 0.264, corresponds to making only one attempt while requesting all of the hints. This corresponds to the case where the student requests hints until the answer is given to them. This could be caused by the student simply not understanding the skill, or by the student "gaming the system," or "attempting to succeed in an interactive learning environment by exploiting properties of the system rather than by learning the material" [2]. In either case, not much learning takes place.

In [15], the AM table was constructed using 80% of the data and used to predict the remaining 20%. In this work, all models were evaluated using five-fold cross-validation.

3.2.2 Assistance Progress Model

AM only takes into account the number of attempts and percentage of hints required on the previous question to predict the student's performance on the following question, without considering the progress the student is making over time in terms of attempts and hints used. APM, on the other hand, takes into account the relationships between the attempts and percentage of hints used on the previous two problems to predict performance on the next question.

The initial model looked like Table 2, each entry corresponding to a case where the second of the two previous problems requires a lower, equal or higher number of attempts or percentage of hints than the one before it. The number of data points for each cell appears in parentheses.

Table 2. Initial APM table for the entire dataset

		Hint % Relationship		
Attempts Relationship		<	=	>
	<	0.672 (586)	0.611 (1410)	0.567 (60)
	=	0.649 (248)	0.734 (8309)	0.590 (83)
	>	0.541 (85)	0.552 (1019)	0.512 (299)

However, it was necessary to extend the model to handle the case where there were fewer than two previous questions, so a separate cell was added for this situation (it had been treated as (equal attempts, equal hint %)). Next, it was observed that certain cells had few observations, so these cells were combined. Finally, it was realized that the (equal attempts, equal hint %) cell combined data of two very different situations: the case where both questions being compared were answered correctly, and the case where they were both answered incorrectly. Therefore, this cell was split into two cells according to correctness. The final APM table, with probabilities taken over the entire dataset, is shown in Table 3. Cells without enough data on their own have been merged, and the (equal attempts, equal hint %) cell has been split in two: the top cell corresponds to the case when both questions are answered correctly, and the bottom to when both are answered incorrectly. The top-left cell contains the probability that questions with fewer than two predecessors will be answered correctly. The number of data points per cell are in parentheses.

Table 3. APM table for the entire dataset

	Hint % Relationship			
Attempt Relationship	0.708 (2722)	<	=	>
	<	0.672 (586)	0.611 (1410)	0.580 (143)
	=	0.649 (248)	0.791 (5028)	
			0.352 (559)	
	>	0.551 (1104)		0.512 (299)

According to Table 3, when the relationship between attempts is held constant, the general pattern is that the probability of correctness decreases as the relationship between the percentage of hints used worsens. The (equal attempts, equal hint %) cells do not fit this pattern, though this could be because they are split based on correctness. However, the same cell from Table 2 also does not fit the pattern. The same relationship exists between the attempt relationship and probability of answering correctly when the hint % relationship is held constant, again with the exception of the (equal attempts, equal hint %) cells. These patterns are intuitive, as students who are learning the material should require less assistance from one problem to the next and are likelier to answer the next question correctly, whereas those who are not learning will generally require the same amount of assistance or more to proceed, and are less likely to answer the next question correctly without assistance.

The highest probability in the table corresponds to the case where the hints and attempts used are the same for the previous two questions, and both are answered correctly (0.791). The lowest is when they are the same and are both answered incorrectly (0.352). The former result is intuitive since it corresponds to the case where the student answers two questions in a row correctly, the best situation represented in the table. The latter corresponds to no progress in terms of assistance over the previous two questions, indicating that little if any learning has taken place.

3.3 Ensembling Models

As shown in [15], ensembling models can give better results than any individual model on its own. There are two goals in this work regarding ensembled models: improving the predictive power of AM by ensembling it with APM, and improving the predictive power of KT using both AM and APM. Wang and Heffernan already showed that ensembling KT with AM gives better results than KT on its own. It remains to be seen whether including APM will result in further improvements.

In addition to using means and linear regression models, as done in [15], this work also uses decision trees and random forests.

3.4 Evaluation

To evaluate the models, three metrics are computed: MAE, RMSE, and AUC. These metrics are computed by obtaining predictions using five-fold cross-validation (using the same partition for each model), then computing each metric per student. Finally, the individual student metrics are averaged across students to obtain the final overall metrics. Computing the average across students for each metric in this way avoids favoring students with more data than others, and avoids statistical independence issues when it comes to computing AUC. For these reasons, Pardos et al used average AUC per student as their accuracy measure in their work in evaluating several student models and various ways of ensembling them [11].

All three of these metrics are reported because they are concerned with different properties of the set of predictions and therefore do not always agree on which model is best. MAE and RMSE are concerned with how close the real-valued predictions are, on average, to their actual binary values. On the other hand, AUC is concerned with how separable the predictions for positive and negative examples are, or how well the model is at predicting binary classes rather than real-valued estimates.

For example, in Table 4, the first two sets of predictions (P1 and P2) achieve AUCs of 1 since both perfectly separate the two classes (0 and 1). However, P2 achieves much better MAE (0.3960) and RMSE (0.6261) values than P1 (0.5940 and 0.7669, respectively). What's more, P3 achieves an AUC of only 0.5, but outperforms both P1 and P2 in terms of RMSE (0.5292) and P1 in terms of MAE (0.4400).

Table 4. Example dataset

Actual Value	P1	P2	P3
0	0	0.99	0.8
1	0.01	1	0.8
1	0.01	1	0.8
0	0	0.99	0.8
1	0.01	1	0.8

Therefore, it is important to report all of these metrics. As shown above, they do not necessarily agree with each other. Additionally, although MAE and RMSE are similar, not even they always agree on the best model, as RMSE punishes larger errors more than MAE does.

4. RESULTS

In this section, the results for both the individual models and the ensemble models are reported. Given the importance of reporting multiple accuracy measures as discussed in the preceding section, three measures are reported for each model: MAE, RMSE and AUC. Each measure is computed by first computing the measure for each individual student, then averaging across students. The individual student measures are obtained by using predictions made using five-fold cross-validation, where the folds used are identical for every model.

4.1 Individual Models

The results of the three individual models are shown in Table 5. As described before, each of the three metrics are measured for each individual student, and then averaged across students.

In addition to the individual models discussed in Section 3, the results for a baseline model (always predicts 1, the majority class) are reported to serve as a baseline for the other models.

Table 5. Results for the individual models

	MAE	RMSE	AUC
Baseline	0.2510	0.4642	0.5000
AM	0.3657	0.4129	0.5789
APM	0.3844	0.4221	0.5618
KT	0.3358	0.4071	0.6466

Unsurprisingly, KT performs reliably better than AM and APM in MAE (*t*(197) = -8.45, p < .0001; *t*(197) = -13.55, p < .0001) and AUC (*t*(187) = 6.35, p < .0001; *t*(187) = 5.97, p < .0001), and at least marginally better in RMSE (*t*(197) = -1.75, p = .0824; *t*(197) = -4.44, p < .0001), as KT is a full student model, whereas AM and APM do not attempt to model student knowledge and make predictions solely on the basis of table lookups. Additionally, AM outperforms APM in MAE and RMSE (*t*(197) = -12.88, p < .0001; *t*(197) = -5.61, p < .0001), which is also not surprising considering that APM does not consider the actual number of attempts or percentage of hints used, only the relationships between them for the previous two questions. APM also has fewer parameters (9) than AM (12). The difference in AUC was not reliable (*t*(187) = 1.62, p = .1063).

The baseline model reliably outperforms all other models in terms of MAE (*t*(197) = -15.30, p < .0001; *t*(197) = -18.36, p < .0001; *t*(197) = -10.62, p < .0001), and reliably underperforms all other models in terms of RMSE (*t*(197) = 5.87, p < .0001; *t*(197) = 4.92, p < .0001; *t*(197) = 6.01, p < .0001) and AUC (*t*(187) = -9.72, p < .0001; *t*(187) = -6.34, p < .0001; *t*(187) = -12.80, p < .0001). It makes sense that the baseline performs well in terms of MAE, given that the mean value of the target attribute, the correctness of a question, is 0.6910. RMSE punishes larger differences more than MAE, making the baseline pay more for its wrong predictions of all cases where the student got the question wrong. Finally, since all predictions share the same value, the baseline cannot do any better than chance at separating the data. Therefore, it earns an AUC value of 0.5000.

These drastic differences in performance for the baseline alone across measures highlight the need for reporting multiple accuracy measures when evaluating student models.

4.2 Ensembled Models

In this subsection, various ways of ensembling the individual models are evaluated. Since KT was the best performer of the individual models in all three measures by at least marginally reliable margins, the ensembled models here are compared to KT. In the results for each ensemble method, <u>underlined</u> type indicates measures that are reliably worse than those for KT, **boldface** type indicates measures that are reliably better than those for KT, and regular type indicates there is no reliable difference between the measures for KT and the model in question. Statistical significance was determined using two-tailed pairwise t-tests and Benjamini and Hochberg's false discovery rate procedure [4].

4.2.1 Mean

The first ensembling method involved taking the simple mean of the predictions given by the various models. This was done in five ways: 1) with AM and APM to determine if it outperformed AM and APM on their own; combining KT with 2) AM and 3) APM to determine if either AM or APM improved predictions over using KT on its own; 4) with all three models to determine if it outperformed any of the individual models, and 5) taking the mean of AM and APM first, then taking the mean of those results with KT. The intuition for the last method is that KT performs better than AM, and most likely APM as well. Therefore, taking the mean of AM and APM first gives KT more influence in the final result while still incorporating both AM and APM. The results for these models are shown in Table 6.

Table 6. Results for the mean models

	MAE	RMSE	AUC
AM, APM	<u>0.3751</u>	0.4137	<u>0.5917</u>
KT, AM	<u>0.3508</u>	**0.4006**	0.6472
KT, APM	<u>0.3601</u>	0.4033	0.6409
KT, AM, APM	<u>0.3620</u>	0.4032	0.6433
KT, (AM, APM)	<u>0.3554</u>	**0.4010**	0.6469

According to the table above, taking the mean of KT and any combination of AM and APM predictions produces results that do as well as or reliably outperform KT in RMSE and AUC but reliably underperform in MAE. There is no reliable difference between the top two performing models, "KT, AM" and "KT, (AM, APM)" except in MAE, where "KT, AM" performs reliably better (*t*(197) = -12.88, p < .0001). Therefore, at least when taking means, adding APM to a model that already includes AM and KT does not reliably improve accuracy in any measure.

Additionally, taking the mean of the AM and APM models yields predictions that are comparable in RMSE and AUC, while reliably worse in MAE (*t*(197) = 12.88, p < .0001). Therefore, including APM predictions in mean models does not appear to improve predictive accuracy.

4.2.2 Linear Regression

The second ensembling method is linear regression. In this method, the training data for each fold was used to construct AM, APM and KT models. Predictions were then made for each training instance using these models, and then a linear regression model was built using the three individual predictions as predictors, along with the number of attempts and percentage of hints used, and nominal attributes describing the relationship between the attempts and hints used on the previous two problems. This model was then applied to the fold's test data, whose instances were augmented with predictions from the AM, APM and KT models built from the fold's training data.

Linear regression models were built with six different subsets of the aforementioned features:

1. AM – includes the AM prediction as well as the number of attempts and percentage of hints used on the previous problem
2. AM, KT – the AM set, along with the KT prediction
3. AM, APM* – the AM set, along with the two nominal attributes indicating the relationships between the attempts and hints used for the previous two problems
4. AM, APM*, KT – the AM, APM* set, along with the KT prediction
5. AM, APM – the AM, APM* set along with the APM prediction
6. AM, APM, KT – the AM, APM*, KT set along with the APM prediction

The motivation for testing these subsets of attributes is to determine the relative improvements attained by progressively adding more assistance relationship information to the model, both with and without KT. These models are built in Matlab using the LinearModel class. The results for the linear regression models are shown in Table 7.

Table 7. Results for the linear regression models

	MAE	RMSE	AUC
AM	0.3701	0.4148	0.5770
AM, KT	**0.3338**	**0.4024**	0.6500
AM, APM*	0.3671	0.4127	0.5753
AM, APM*, KT	**0.3319**	**0.4005**	0.6341
AM, APM	0.3647	0.4112	0.5874
AM, APM, KT	**0.3316**	**0.4000**	0.6379

Not surprisingly, models that incorporate KT predictions all outperform their counterparts that lack KT predictions across all three measures. AM and APM together do better than AM, but not when KT is included. The best combination of models for linear regression is AM and KT, as it was for the mean models.

Unlike its corresponding mean model, the linear regression model that combines AM and KT reliably outperforms KT in MAE and RMSE, and is comparable in terms of AUC. This is consistent with the previous finding that combining AM and KT using linear regression outperforms KT [15], though their model did reliably better than KT for all three measures, which were taken over the entire dataset rather than averaged across students.

4.2.3 *Decision Trees*

Next, decision tree models were built from the results of the three individual models in the same way that the linear regression models were built, with the exception that the minimum number of data points per leaf and the level of pruning were optimized using brute force search per fold by using sub-fold cross-validation. The search varied the pruning level from 0 to 100% of the model in steps of 5%, and varied the minimum data points per leaf from 5 to 50 in steps of 5.

The decision trees were given the same set of attributes as the linear regression models, and were tested using the same six subsets of those attributes as described above for the linear regression models. The decision trees were built in Matlab using classregtree, specifying the method as 'regression'.

The same sub-folds were used for each fold for all decision tree models. The results for these models are reported in Table 8. The model names correspond to the same subsets of attributes used for the linear regression models.

Table 8. Results for the decision tree models

	MAE	RMSE	AUC
AM	0.3637	0.4119	0.5793
AM, KT	**0.3293**	**0.4009**	0.6385
AM, APM*	0.3586	0.4087	0.5847
AM, APM*, KT	**0.3286**	**0.4008**	0.6358
AM, APM	0.3586	0.4090	0.5860
AM, APM, KT	**0.3290**	**0.4012**	0.6351

As for the linear regression models, the models that include KT predictions perform better than those that did not, across all three accuracy measures. Adding APM* to AM reliably improves accuracy, but there is no difference between this and combining AM and APM. Adding APM features of any kind do not improve models that include KT predictions. As for the linear regression models, the decision tree that performs the best is the one that only includes KT and AM, which reliably outperforms KT in both MAE and RMSE, with no reliable difference in AUC.

4.2.4 *Random Forest*

The final ensembling method used in this work was Random Forest, which is a collection of decision trees where each individual decision tree was built from a random subset of the attributes and a random subset of the data. In this work, random forests consisted of 1,000 such trees, which were each built randomly from any subset of the attributes and between 10% and 90% of the data. The prediction of the random forest as a whole for a given test instance was the simple mean of the predictions given by each individual tree within the forest. The trees were regression trees and required a minimum of five data points per leaf node. No pruning was done, as varying the pruning levels did not appear to significantly affect the predictive accuracy of the forests for this dataset.

The same set of attributes used for linear regression and decision trees were used in the random forest models, and the same six attribute subsets were tested separately as for the other methods.

With the exception of MAE (many of the predictions were 1, which happens to be the majority class), these models performed worse than the other ensembling methods. This could be due to most of the trees being overfit to the training data, as sub-fold cross-validation with brute force search of optimal pruning parameters was not performed for these trees as it was for the individual decision trees reported on in the previous section.

However, averaging these models with KT produced better results, as shown in Table 9.

Table 9. Results for averaging the KT and random forest models

	MAE	RMSE	AUC
AM	0.3505	**0.4002**	0.6461
AM, KT	**0.3054**	0.4117	0.6313
AM, APM*	0.3479	**0.3985**	0.6477

AM, APM*, KT	**0.3005**	0.4109	0.6358
AM, APM	<u>0.3485</u>	**0.3990**	0.6468
AM, APM, KT	**0.2997**	0.4090	0.6375

Unlike other ensembling methods, when random forest predictions are averaged with those of KT, progressively more APM data improves accuracy, though not always significantly. Otherwise, adding APM predictions appears to worsen results.

4.2.5 Overall

For the first three ensembling methods, those that included only AM and KT performed the best. However, for random forests, it was the average of KT with the random forest consisting of predictions from all three individual models. Table 10 reproduces these results, with **bold-faced** type indicating values that are reliably better than KT, and <u>underlined</u> type indicating values that are reliably worse. Table 11 reports the p-values of the differences between these models for each accuracy measure, with values indicating reliable differences in **bold-faced** type.

Table 10. Results for the best of each ensembling method

	MAE	RMSE	AUC
MEAN	<u>0.3508</u>	**0.4006**	0.6472
LR	**0.3338**	**0.4024**	0.6500
TREE	**0.3293**	**0.4009**	0.6385
RF	**0.2997**	0.4090	0.6375

Table 11. Significance tests for the best ensembling methods

	MAE	RMSE	AUC
MEAN, LR	**0.0000**	0.1659	0.4274
MEAN, TREE	**0.0000**	0.8803	0.1116
MEAN, RF	**0.0000**	**0.0022**	0.1400
LR, TREE	**0.0000**	0.1669	0.0223
LR, RF	**0.0000**	**0.0026**	0.0406
TREE, RF	**0.0000**	**0.0001**	0.8476

From Tables 10 and 11, it appears that either the decision tree or random forest (averaged with KT) models could be considered the best model, depending on which measure is considered the most important. The random forest model is reliably better than the decision tree in terms of MAE, but reliably worse in terms of RMSE.

In general, it appears there is some value in comparing the usage of assistance over the previous two problems, as ensembling APM with AM consistently gives better results than using AM on its own, except when taking means. Despite this, ensemble methods that use only KT and AM perform better than any other model studied in this work, including all of those using APM. One explanation could be that one important thing that APM captures is learning over the previous two questions, which is already modeled in KT. The one exception is when a random forest of all individual models is averaged with KT, which indicates that there is information that APM takes into account that neither AM nor KT considers. Right now, it is not clear which of these ensemble models is best given the disagreement among the metrics. It depends on the relative importance placed on each metric.

5. DISCUSSION AND FUTURE WORK

In this work, we generalized an existing raw data model, AM, into a framework for predicting student performance by tabling raw data. This framework provides an efficient way for adding new sources of information into existing student models. From there, we developed a new model, APM, which makes predictions based on the relationship between the assistance used on the previous two problems. Finally, we evaluated these models and KT, and then explored several ways of ensembling these models together.

We found that although APM is not as predictive as AM, combining the two with various ensembling methods produces models that reliably outperform AM on its own. This shows that prediction accuracy can be strengthened by recognizing the progress a student makes in terms of the assistance they use. However, for the most part, the best models studied in this paper were those that only ensembled KT and AM. Adding APM to such models did not improve accuracy, except in the case of random forests averaged with KT. Despite this, it is still evident that there is value in considering student progress in terms of assistance. Perhaps there are better methods of incorporating that information into predictive models that will yield better results.

We also confirmed that ensembles of AM and KT reliably outperform KT, in line with previous work [15]. Whereas previous work showed this was the case when computing the measures across all problem logs, this work shows it also holds when the measures are computed as averages across students.

We reported three different accuracy measures to fairly compare models against each other, and argued that reporting multiple measures is necessary since they measure different properties of the predictions and therefore do not always agree on which model is best. We also argued that computing these measures per student, then averaging across students is more reliable than treating all problems as equal since the latter approach favors models that are biased towards students with more data.

Although we found that the ensemble methods perform better than KT at predicting performance, such models are difficult to interpret and therefore may be limited in usefulness. Fitting a KT model for a given skill yields four meaningful parameters that describe the nature of that skill, whereas ensemble methods in this work give models of how to computationally combine predictions from KT and AM to maximize predictive accuracy. Since KT models student knowledge, it can be used to guide an ITS session. KT can also be extended to quantify the effects of help [3], gaming [7], and individual items [10], among other factors, on learning and performance. It appears the usefulness of the ensemble methods is limited to prediction of the next question, a task that serves as a good measure of the validity of a student model but does not appear to be useful in guiding ITS interaction.

On the other hand, AM and APM are simple to compute and do not suffer from the identifiability problem that KT does [13]. AM and APM consist of summaries of the raw data rather than inferred parameters. Although not as predictive as KT, AM and APM give interpretable statistics with little chance of overfitting. Additionally, they consider the assistance used, which could indicate the usefulness of a system's help features. Other tabling methods could be used to study the effects of other aspects of ITS, though likely with lower predictive accuracy than KT due to the limited set of values such methods can use as predictions.

Since the ensemble models outperformed KT but appear to be limited to predicting a student's performance on the next question, finding a way to use such predictions within ITS would be a useful contribution. Question selection could be a possible application (i.e. selecting an easier question if the model predicts the student will answer the next question incorrectly).

Another direction for future work could be determining other useful specializations of the framework we presented for building models from raw data. AM and APM focus on assistance, but other attributes could prove useful. Additionally, this work did not investigate grouping students or problems.

Another future direction could be determining why some models, like APM, can reliably improve a model such as AM when ensembled with it, but not improve results when a third model such as KT is involved. It appears that the information that APM uses is important, but may not be used by APM in the best way possible. Examining the use of assistance over the course of more than just the previous two problems may also prove useful.

Finally, experimenting with other methods of ensembling the models described here and other raw data models within this framework is also worth looking into. Previous work experimented with means and linear regression [15], and this work expanded upon those methods by including decision trees and random forests. However, other work in ensembling student models suggests that neural networks may perform better [11].

6. ACKNOWLEDGMENTS

This research was supported by a Graduate Assistance in Areas of National Need (GAANN) fellowship and Neil Heffernan's CAREER grant. We also acknowledge the many additional funders of the ASSISTments Platform found here: http://www.webcitation.org/5ym157Yfr

All of the opinions expressed in this paper are those solely of the authors and not those of our funding organizations.

7. REFERENCES

[1] Arroyo, I., Cooper, D.G., Burleson, W., and Woolf, B.P. Bayesian Networks and Linear Regression Models of Students' Goals, Moods, and Emotions. in *Handbook of educational data mining*, CRC Press, Boca Raton, FL, 2010, 323-338.

[2] Baker, R.S.J.d., Is Gaming the System State-or-Trait? Educational Data Mining Through the Multi-Contextual Application of a Validated Behavioral Model. in *Complete On-Line Proceedings of the Workshop on Data Mining for User Modeling at the 11th International Conference on User Modeling*, (Corfu, Greece, 2007), 76-80.

[3] Beck, J.E., Chang, K., Mostow, J., Corbett, A. Does help help? Introducing the Bayesian Evaluation and Assessment methodology. *Intelligent Tutoring Systems*, Springer Berlin Heidelberg, 2008, 383-394.

[4] Benjamini, Y., Hochberg, Y. Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. *Journal of the Royal Statistical Society*, *Series B*, *57*(1), 289–300.

[5] Corbett, A. and Anderson, J. Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction*, *4*(4), 253-278.

[6] Feng, M., and Heffernan, N.T., Can We Get Better Assessment From a Tutoring System Compared to Traditional Paper Testing? Can We Have Our Cake (Better Assessment) and Eat It Too (Student Learning During the Test)? in *Proceedings of the 3rd International Conference on Educational Data Mining*, (Pittsburgh, PA, 2010), Springer Berlin Heidelberg, 309-311.

[7] Gong, Y., Beck, J., Heffernan, N., Forbes-Summers, E, The impact of gaming (?) on learning at the fine-grained level. in *Proceedings of the 10th International Conference on Intelligent Tutoring Systems*, (Pittsburgh, PA, 2010), Springer, 194-203.

[8] Johns, J., Mahadevan, S., Woolf, B. Estimating student proficiency using an item response theory model. *Intelligent Tutoring Systems*, Springer Berlin Heidelberg, 2006, 473-480.

[9] Murphy, K. The bayes net toolbox for matlab. *Computing science and statistics*, *33*(2), 1024-1034.

[10] Pardos, Z.A., Dailey, M.D., Heffernan, N.T. Learning what works in ITS from non-traditional randomized controlled trial data. *International Journal of Artificial Intelligence in Education*, *21*(1), 47-63.

[11] Pardos, Z.A., Gowda, S. M., Baker, R.S.J.d. and Heffernan, N. T. The Sum is Greater than the Parts: Ensembling Models of Student Knowledge in Educational Software. *ACM SIGKDD Explorations*, *13*(2), 37-44.

[12] Pardos, Z. A., Heffernan, N. T., Modeling Individualization in a Bayesian Networks Implementation of Knowledge Tracing. in *Proceedings of the 18th International Conference on User Modeling, Adaptation and Personalization*, (Big Island, Hawaii, 2010), 255-266.

[13] Pardos, Z. A., Heffernan, N. T., Navigating the parameter space of Bayesian Knowledge Tracing models: Visualizations of the convergence of the Expectation Maximization algorithm. in *Proceedings of the 3rd International Conference on Educational Data Mining*. (Pittsburg, PA, 2010), Springer Berlin Heidelberg, 161-170.

[14] Pavlik, P.I., Cen, H., Koedinger, K., Performance Factors Analysis – A New Alternative to Knowledge. in *Proceedings of the 14th International Conference on Artificial Intelligence in Education*, (Brighton, U.K., 2009), 531-538.

[15] Wang Y., Heffernan N. T., The "Assistance" Model: Leveraging How Many Hints and Attempts a Student Needs. in *Proceedings of the 24th International FLAIRS Conference*, (Palm Beach, FL, 2011)

[16] Wang, Y., Heffernan, N.T., The Student Skill Model. in *Proceedings of the 11th International Conference on Intelligent Tutoring Systems*, (Chania, Greece, 2012), 399-404.

[17] Wang, Y., Heffernan, N.T. and Beck, J.E., Representing Student Performance with Partial Credit. in *Proceedings of the 3rd International Conference on Educational Data Mining*, (Pittsburgh, PA, 2010), Springer Berlin Heidelberg, 335-336

[18] Xu, Y., Mostow, J., Comparison of methods to trace multiple subskills: Is LR-DBN best? in *Proceedings of the Fifth International Conference on Educational Data Mining*. (Chania, Greece, 2012), 41-48.

Mining Meaningful Patterns from Students' Handwritten Coursework

James Herold
Department of Computer Science
University of California, Riverside
900 University Ave
Riverside, CA 92521
jhero001@ucr.edu

Alex Zundel
Department of Mechanical Engineering
University of California, Riverside
900 University Ave
Riverside, CA 92521

Thomas F. Stahovich
Department of Mechanical Engineering
University of California, Riverside
900 University Ave
Riverside, CA 92521
stahov@engr.ucr.edu

ABSTRACT

A key challenge in educational data mining research is capturing student work in a form suitable for computational analysis. Online learning environments, such as intelligent tutoring systems, have proven to be one effective means for accomplishing this. Here, we investigate a method for capturing students' ordinary handwritten coursework in digital form. We provided students with Livescribe^{TM}digital pens which they used to complete all of their homework and exams. These pens work as traditional pens but additionally digitize students' handwriting into time-stamped pen strokes enabling us to analyze not only the final image, but also the sequence in which it was written. By applying data mining techniques to digital copies of students' handwritten work, we seek to gain insights into the cognitive processes employed by students in an ordinary work environment.

We present a novel transformation of the pen stroke data, which represents each student's homework solution as a sequence of discrete actions. We apply differential data mining techniques to these sequences to identify those patterns of actions that are more frequently exhibited by either good- or poor-performing students. We compute numerical features from those patterns which we use to predict performance in the course. The resulting model explains up to 34.4% of the variance in students' final course grade. Furthermore the underlying parameters of the model indicate which patterns best correlate with positive performance. These patterns in turn provide valuable insight into the cognitive processes employed by students, which can be directly used by the instructor to identify and address deficiencies in students' understanding.

1. INTRODUCTION

Educational data mining has typically been applied to data extracted from students' interactions with Intelligent Tutoring Systems (ITS) and Course Management Systems (CMS). This research has been used to improve the way students interact with these interfaces and has led to a better understanding of the ways students learn when using these systems.

In this study, we investigate a method for capturing students' ordinary, handwritten coursework in digital form. In the winter quarter of 2012, undergraduate Mechanical Engineering students were provided Livescribe^{TM}digital pens with which they completed all their coursework. These pens record students' handwriting as time-stamped pen strokes, enabling us to analyze not only the final image, but also the sequence in which it was written.

We have developed a novel representation of a student's handwritten assignment which characterizes the sequence of actions the student took to solve each problem. This representation comprises an alphabet of canonical actions that a student may perform when solving a homework assignment. Each action is characterized by its duration, problem number, and semantic content. This representation allows us to apply traditional data mining techniques to our database of students' handwritten homework solutions.

Our analysis focus on two separate groups of students: those who scored in the top third of the class on exams, and those who scored in the bottom third. We applied a differential data mining technique to the sequences of each of these groups and identified behaviors that are more frequently exhibited by one group than the other.

These patterns serve as the basis for a number of features used to train a linear regression model to predict students' performance in the course. This model achieves an R^2 of 0.34. More importantly, the underlying parameters of this model provide valuable insights as to which of the patterns most correlate with performance. Using these most-predictive patterns, we are able to identify high-level, cognitive behaviors exhibited by the students.

2. RELATED WORK

Data-driven educational research has traditionally been limited by the time-consuming process of monitoring students' learning. For example, substantial research has been performed which investigated the correlation between performance and the amount of time and effort spent on homework assignments [2, 5, 8, 16, 18]. Manually watching each student solve each homework assignment would require an intractable amount of time and, additionally, may skew the results of the study. Instead, each of these researchers relied on students or their parents to self-report the amount of time spent on each homework assignment.

Cooper et al. [6] compared the results of each of these studies and found an average correlation of $r = 0.14$ with a range from -0.25 to 0.65. Cooper et al. summarize this inconsistency in findings when they state that, "to date, the role of research in forming homework policies and practices has been minimal. This is because the influences on homework are complex, and no simple, general finding applicable to all students is possible." This underlies the impact that Educational Data Mining can have on the educational research community. By instrumenting students' natural problem-solving processes, we are able to capture a precise measurement of the actions students perform when solving their homework assignments.

More recently, researchers have applied data mining techniques to ITS and CMS data. For example, Romero et al. [17] applied data mining techniques to data collected with the Moodle CMS. This system allows students to both view and submit various assignments, e.g., homework and exams, and records detailed logs of students' interactions. These interaction logs were mined for rare association rules, that is, patterns which appear infrequently in the data. The resulting rules were then manually inspected to identify fringe behaviors exhibited by students.

Similarly Mostow et al. [14] applied data mining techniques to interaction logs taken from Project LISTEN's Reading Tutor, an ITS. This system tutors young students as they learn to read by listening to them read stories aloud and providing feedback. The authors developed a system which automatically identified meaningful features from these logs which were then used to train classifiers to predict students' future behavior with the system.

Sequential pattern mining [1] is a technique used to identify significant patterns in sequences of discrete items, e.g., consumer transaction records [1] or DNA transcripts [4]. These techniques have typically been used to mine patterns from a single database of sequences. In Educational Data Mining, it is often the case that researchers seek to find patterns that best distinguish students who do and do not perform well in the course. Thus there is a need for novel pattern mining techniques aimed at differentiating between two databases of sequences.

More recently, Ye and Keogh [20] developed a novel technique which identifies patterns which best separate two time-series databases. This technique identifies frequently occurring patterns within each database, as traditional pattern mining techniques have, but furthermore, evaluates each pattern by using it to separate sequences from the two databases. If a sequence contains the pattern, that sequence is identified as being part of the same database that the pattern came from. The pattern which provides the greatest information gain is kept as the "shapelet" that best separates the two databases.

Similarly, Kinnebrew and Biswas [13] have developed a novel differential pattern mining technique used to identify patterns that differentiate between the interactions of different groups of students with the Betty's Brain ITS. This technique begins by using SPAM [3] to identify patterns that occur in a significant number of sequences in either database. A t-test for each pattern is then performed to determine if there is a significant difference in the frequency of that pattern in each sequence of each of the two databases. This algorithm can identify patterns that occur significantly frequently in one database and not the other.

The work of Oviatt et al. [15] suggests that natural work environments are critical to student performance. Their examination of computer interfaces for completing geometry problems suggests that, "as the interfaces departed more from familiar work practice..., students would experience greater cognitive load such that performance would deteriorate in speed, attentional focus, meta-cognitive control, correctness of problem solutions, and memory." Thus, our goal is to apply Educational Data Mining techniques to data collected in natural work environments.

To that end, recent research has focused on mining ordinary, handwritten coursework data. For example, Van Arsdale and Stahovich [19] demonstrated that a correlation exists between the temporal and spatial organization of students' handwritten problem solutions and the correctness of the work. The organization of exam solutions was characterized by a set of quantitative features, which were then used to predict performance on those problems. On average these features accounted for 40.0% of the variance in students' performance on exam problems.

Similarly, Herold and Stahovich [12] presented a study in which data mining techniques were applied to students' handwritten coursework to identify how self-explanation affected students' solution processes. In this study, students from a Mechanical Engineering course were split into two groups, one which provided handwritten self-explanation along with their homework assignments and one which did not. Digital copies of the students' handwritten homework were mined for commonly occurring n-grams, revealing that students who generated self-explanation solved problems more like an expert than did those who did not generate self-explanation.

In this work, we build upon these prior efforts by applying differential pattern mining techniques to students' ordinary, handwritten problem-solving processes. In so doing, we aim to identify successful and unsuccessful solution habits and infer the higher-level cognitive processes they indicate.

3. DATA COLLECTION

In the winter quarter of 2012, students enrolled in an undergraduate Mechanical Engineering Statics course were given Livescribe$^{\text{TM}}$digital pens. Students completed all their coursework with these pens, creating a digital record of their handwritten homework, quiz, and exam solutions. A typical exam problem is shown in Figure 1. Each problem includes a figure describing a system subject to external forces. The

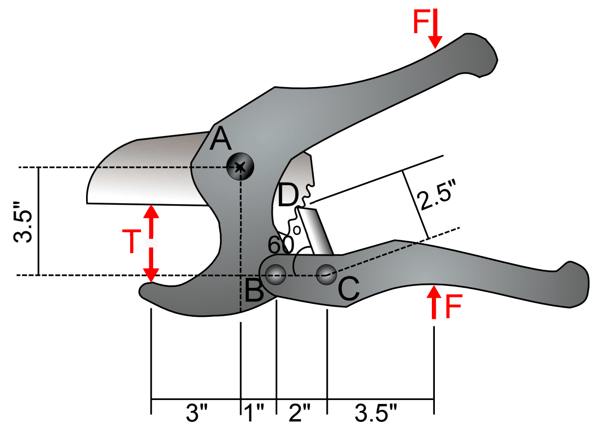


Figure 1: A typical Statics problem. The problem statement reads, "The device shown is used for cutting PVC pipe. If a force, $F = 15lb$, is applied to each handle as shown, determine the cutting force T. Also, determine the magnitude and the direction of the force that the pivot at A applies to the blade."

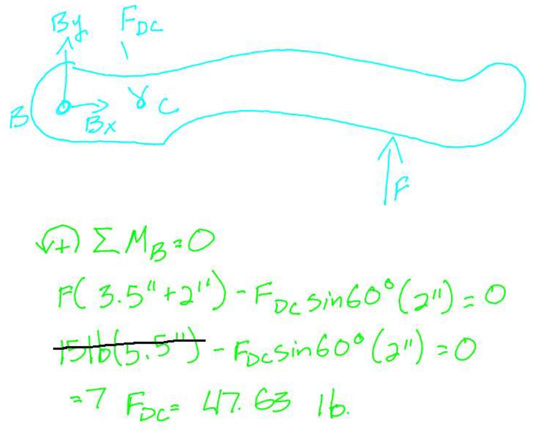

By
FDC
B
Bx
γ C
F
↺+ ΣMB=0
F(3.5"+2") - FDC sin60°(2") = 0
~~15lb(5.5")~~ - FDC sin60°(2") = 0
=> FDC = 47.63 lb.

Figure 2: A hypothetical solution to a Statics problem. The color of each pen-stroke identifies the component to which it refers: cyan = FBD, green = equation, and cross-out = black.

student must apply the principles of equilibrium to compute the resulting reaction forces.

Students typically begin solving problems by drawing a free body diagram (FBD) representing the boundary of the system and the forces acting on it. The free body diagram is then used as a guide for constructing force and moment equilibrium equations. Most pen strokes in a solution correspond to either a free body diagram, an equation, or a cross-out. (Because the Livescribe pens use ink, students cannot erase errors and must instead cross them out.) Figure 2 shows a hypothetical solution to a Statics problem.

In this study, we focus on the data for homework assignments three, four, five, six, and eight, as these are the ones focused on equilibrium analysis. Assignments one and two, by contrast, focused on basic vector math, while assignment seven was a review of centroids.

The resulting data set comprises 556 sketches from 132 students. Each sketch corresponds to a single page of work from a student. Each sketch, $K = \{s_1, ..., s_m\}$, comprises a series of pen strokes. Each pen stroke, $s_i = \{p_1, ..., p_n\}$, comprises a series of points. Each point $p_j = \{x, y, t\}$ is a triple where x and y are two-dimensional Cartesian coordinates, and t is the time-stamp of that point. All points within a pen stroke, and all pen strokes within a sketch, are ordered by increasing time-stamp. The time-stamp of the first point in a pen stroke signifies the start time of that pen stroke and the last point is used to signify its end time. A sequence of labels also exists for each sketch, $L = \{l_1, ..., l_m\} | l \in \{FBD, EQN, CRO\}$. Each label, l_i, identifies stroke, s_i, by its semantic content: free body diagram (FBD), equation (EQN), or cross-out (CRO). We manually labeled the pen strokes of each sketch, but it has been shown in recent work that this process may be automated reliably [11].

While these labels account for virtually all the ink written by students, more fine-grained labeling schemes could have been developed by subdividing each label. For example, instead of labeling a pen stroke as being part of an equation, it could be labeled according to the type of equation to which it corresponds, namely, sum of forces in the X or Y direction or the sum of moments. We chose the labeling scheme presented for two major reasons.

First, this labeling scheme is sufficient for investigating hitherto unverifiable intuitions about the ways students solve Statics problems, such as the intuition that students who possess a strong understanding of the material will complete their FBD entirely before beginning their equation work. Similarly, we may corroborate the intuition that students who possess a strong understanding of the material will complete their problems in problem-number order, that is, they complete problem one entirely before completing problem two and so on.

Second, by subdividing each of the labels, we risk increasing the granularity of the resulting action sequences too far, increasing the number of total discrete actions to a point that prevents patterns from being identified.

4. ACTION SEQUENCES

In this section, we describe how each sketch may be transformed into an *action sequence*, comprising discrete actions, that is suitable for differential pattern mining. Each action is an element of a predefined alphabet of canonical actions. Each element in the alphabet represents an uninterrupted period of problem-solving performed by a student as he or she solves a homework assignment. We seek to characterize the duration, semantic content, and homework problem number for each action.

We begin by segmenting the pen strokes of each sketch by semantic type. To do so, we simply identify each index, i, in L such that $l_i \neq l_{i+1}$, and segment the series of pen strokes at each identified index. Each resulting segment contains

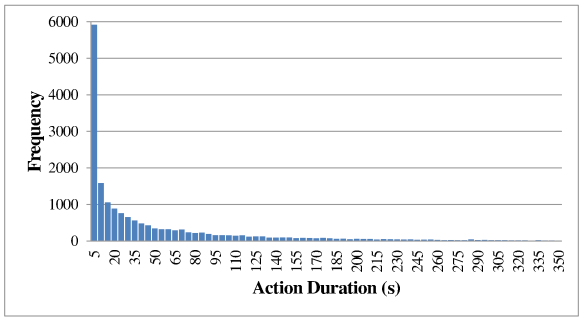


Figure 3: A histogram of the durations of FBD actions across all homework assignments. For example, the first(leftmost) bar indicates that approximately 6,000 FBD actions were between zero and five seconds long.

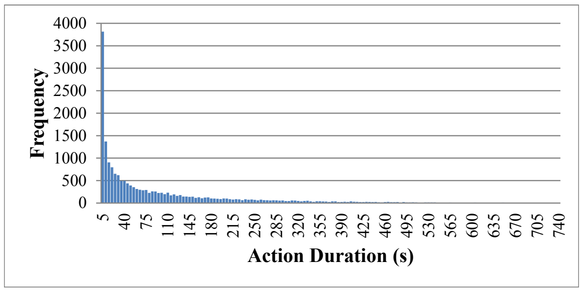


Figure 4: A histogram of the durations of EQN actions across all homework assignments. For example, the first(leftmost) bar indicates that approximately 3,500 EQN actions were between zero and five seconds long.

a sequence of actions corresponding to the same semantic type. The resulting segments do not yet satisfy the above definition of an action, as they do not necessarily contain uninterrupted work. Thus, we further segment the sketch at each index, j, such that the difference between the start time of s_j and the end time of s_{j-1} is greater than a specified threshold. In this paper, we use a threshold of five minutes, which was determined *a priori*; five minutes is a sufficiently large gap to be considered an intended break in the problem-solving process.

Each segment is then labeled with an element from the alphabet of canonical actions. If the segment comprises cross-out pen strokes, then it is given the cross-out label, C, regardless of its length or problem number. The remaining groups are labeled with a triple, $\{P, T, D\}$, where P represents the problem number, T represents the semantic type, and D, represents the duration of the action. $P \in \{1, ..., 8\}$ as there are never more than eight problems on a given homework assignment. $T \in \{F, E\}$ where F represents a FBD action and E represents an equation action. Lastly, $D \in \{S, M, L\}$, where S, M, and L indicate an action of small, medium, or large duration respectively. Take for example, the label <1-E-S>. This indicates a small action on the equations from problem one of an assignment.

The cut-off points for each duration category were determined by studying the distribution of lengths of all the FBD and equation actions. Figures 3 and 4 show a histogram for the duration of FBD and equation actions respectively. We partition each distribution into three segments such that the area under the curve for each segment is equal. The resulting thresholds are 11.26 and 80.1 seconds for FBDs and 29.59 and 147.82 seconds for equations. There are 49 unique labels in the canonical action alphabet, comprising the 48 possible combinations for a given triple and the additional cross-out label.

We seek to assign the action sequences of a student to a *performance group* based on that student's performance. In particular, we group a student's action sequence for an assignment by that student's performance on the most relevant exam, which we defined as the one that occurred most recently after that assignment was due.

Students completed homework assignments three and four prior to the first midterm exam. Students completed homework assignments five and six after the first midterm exam and before the second. Students completed homework assignment eight after the second midterm exam and before the final exam. Each midterm exam only comprised problems similar to those encountered on the homework assignments leading up to it. Thus the first midterm exam required that students solve problems similar to those found on homework assignment three and four and the second midterm exam required students to solve problems similar to those found on homework assignments five and six. The final exam comprised problems similar to all those encountered on all homework assignments.

Using this schedule of exams and homework assignments, we assign each action sequence to a group based on performance. An action sequence is assigned to the top-performing group if the student who performed those actions scored in the top third on the relevant exam. Similarly, an action sequence is assigned to the bottom-performing group if the student scored in the the bottom third of the class. The differential mining technique employed in this paper requires exactly two databases as input, thus the remaining middle-performing students are excluded from our analysis to help accentuate the differences in problem-solving behaviors of top- and bottom-performing students.

Descriptive statistics of the lengths of the action sequences for the two performance group for each assignment are shown in Table 1. It is interesting to note that the average action sequences of the bottom-performing group are always longer than those of the top-performing group, and in two cases this difference is significant ($p < 0.01$).

5. DIFFERENTIAL MINING

To identify patterns that distinguish good performance from poor performance we employ the differential pattern mining technique developed by Kinnebrew and Biswas [13]. This algorithm identifies patterns that are differentially frequent with respect to two databases of sequences, called the left and right databases.

This algorithm uses two metrics to measure the frequency of a pattern, s-frequency and i-frequency. s-frequency is de-

Table 1: Average, median, and standard deviation of the sequences for each grouping of sequences on each assignment. The fourth column contains the p-value of a t-test comparing the bottom-performing and top-performing groups on each assignment.

Group	Average	Median	Std. Dev.	t-test
HW3 Bot.	89.52	88	43.30	0.21
HW3 Top	75.38	54.5	56.12	–
HW4 Bot.	130.25	128	63.97	0.00
HW4 Top	83.28	78	46.67	–
HW5 Bot.	127.88	119.5	70.89	0.01
HW5 Top	87.14	72	54.42	–
HW6 Bot.	144.73	140	52.55	0.171
HW6 Top	126.52	122	66.05	–
HW8 Bot.	82.45	72	73.94	0.17
HW8 Top	62.28	53.5	39.64	–

fined as the number of sequences in a database that contains a specific pattern. i-frequency is defined as the number of times a pattern appears within a single sequence. Take for example, a database of ten sequences in which the first seven sequences contain one instance of a particular pattern and the last three sequences contain two instances of that same pattern. This pattern would then have a s-frequency of 10. This pattern would have an i-frequency of one in the first pattern and an i-frequency of two in the last pattern.

This algorithm begins by finding all patterns that meet a specified s-frequency threshold in the left and right database separately. Each such pattern is called an s-frequent pattern. A modified implementation of the SPAM algorithm [3] is used to identify the initial set of s-frequent patterns constrained by the a maximum gap between subsequent elements within a pattern. We use a maximum gap constraint of two in our study.

The i-frequency of each s-frequent pattern is computed for each sequence in each database. A separate t-test is computed for each s-frequent pattern to determine if the i-frequency values computed using the left database are significantly different from those computed for the right database. If the resulting p-value of the t-test is below a certain threshold, called the p-value threshold, it is considered to be *differentially frequent.* This algorithm identifies four types of differentially frequent patterns: those that are s-frequent in both sets but whose average i-frequency is higher in the left database; those that are s-frequent in both sets but whose average i-frequency is higher in the right database; those that are are only s-frequent in the left database; and those that are only s-frequent in the right database. In this study, we consider only the sequences from the last two cases as they are the most most useful for distinguishing between good- and poor-performing students.

In our implementation, we use the set of sequences from the bottom-performing group as the left database and those from the top-performing group as the right database. We use a s-frequency threshold of 0.6, meaning that a pattern must appear in at least 60% of the sequences in a database in order to be considered s-frequent. We use a p-value threshold of 0.1.

6. PERFORMANCE PREDICTION

The differential pattern mining technique identified 98 patterns in total: 6 that were s-frequent in the top-performing group but not in the bottom-performing group, and 92 that were s-frequent in the bottom-performing group but not in the top-performing group.

Our goal is to use these 98 patterns to construct a model to distinguish between good- and poor-performing students. We represent each student with 98 binary features. Each feature indicates whether a particular differential pattern from a particular assignment is contained within a student's action sequence for that assignment. To avoid computing a model that over-fits the data, we used the Correlation-based Feature Selection (CFS) algorithm with 10-fold cross-validation to identify the subset of the 98 features with the most predictive power. Those features that were selected in more than six of the ten folds by the CFS algorithm were included in the final feature subset. Table 2 shows the 20 features that were ultimately selected in this way.

We then used these 20 features to construct a linear regression model which predicts students' overall performance in the course. While more robust, non-linear classifiers could have been used, e.g., AdaBoost [7] or Support Vector Machines [9], we use a linear regression model because of the ease of interpretation; the coefficients that comprise the model give insight into the predictive power of the features used to train it. We used the linear regression package available in the WEKA machine learning software suite [10] to train the model. Our predictive model achieves an R^2 of 0.343 and includes seven features with non-zero coefficients. Table 3 lists these seven features.

7. DISCUSSION

We manually inspected each of the 98 patterns identified by the differential pattern mining algorithm and categorized the different types of cognitive processes they demonstrate. We identified seven distinct categories. *Difficulty* is the category in which students seem to encounter difficulties with a particular problem, evidenced by either repeated cross-outs or repeated attempts at the same component of the same problem. For example, the pattern <C, 1-E-S, C> describes a scenario in which the student crossed out work, worked on equations for problem one for a short time, and then again crossed out work.

Three categories describe patterns in which actions are repeated: *Repeated Equation*, *Repeated FBD*, and *Repeated Cross-out.* For instance, <2-E-S 2-E-S> is an example of a *Repeated Equation* action. Such sequences may be an indication that a student is taking a break in the middle of a particular activity to think more carefully before continuing with that activity.

Two categories describe patterns suggesting that a student may be revising either a FBD (*FBD Revision*) or an equation (*Equation revision*). These patterns comprise a cross-out followed by either the FBD or equation they are most likely revising. Also, when a student moves from working on an equation back to a FBD, this is likely an indication that the FBD is being revised; students typically attempt to complete their FBD before moving on to equations.

Table 2: Features selected using the CFS algorithm. Each feature corresponds to a pattern identified by the differential pattern mining algorithm. Each line shows the homework number and group (top or bottom) from which the pattern was identified. The final column shows the pattern that was used to compute the feature.

HW No.	Perf. Group	Sequence
3	Top	1-E-M 1-F-S
3	Top	1-F-M 1-E-M
3	Bot	2-F-M 2-E-S
3	Bot	C 5-E-S
3	Bot	5-E-M 5-F-S
3	Bot	5-E-S 5-F-M
3	Bot	C 4-E-L
4	Bot	C 1-E-M
4	Bot	1-E-L C
4	Bot	C 5-E-L
4	Bot	1-E-M 1-E-S
4	Bot	C C
5	Bot	4-E-S 4-E-S
6	Bot	1-F-M 1-E-S 1-E-M
6	Bot	1-E-S 1-E-M 1-F-M
6	Bot	1-F-M 1-F-S
6	Bot	1-F-S 1-F-M
8	Bot	5-F-M 5-F-S
8	Bot	5-F-S 5-E-M
8	Bot	5-F-M 5-E-S

Table 3: Non-zero feature coefficients for the linear regression model trained to predict student performance.

Sequence	HW	Weight	Category
1-F-S 1-F-M	3	48.8	Repeated FBD
C 5-E-L	3	51.0	EQN Revision
C 5-E-S	4	51.2	EQN Revision
1-E-M 1-F-S	4	55.5	FBD Revision
C 1-E-M	6	62.7	EQN Revision
1-F-M 1-E-S 1-E-M	6	63.1	Difficulty
5-F-M 5-F-S	8	73.1	Repeated FBD

Lastly, is the *Normal* category. This is the category for all patterns in which a FBD is followed by an equation of the same problem number. A differential pattern belonging to the *Normal* category is particularly informative when one group exhibits significantly more normal sequences – it is an indication that the other group is solving their homework assignment out-of-order more often.

The non-zero weighted features of the linear regression model (Table 3) help identify the patterns which are most predictive of students' grades, and thus provide insight into the behaviors which best correlate with students' performance. In Table 3, Patterns 1, 4, 5, and 6 are all similar in that they comprise actions pertaining to the first problem on a homework assignment, and suggest that a student may be having difficulty or is frequently revising his or her work. This is an indication that when students encounter difficulty on the first problem, which is typically the easiest problem of the homework assignment, that they may continue to encounter those difficulties throughout the quarter.

Patterns 2, 3, and 7 in Table 3 are all similar in that they pertain to problems that are very similar to problems that appear on either a later midterm, the final exam, or both. (These problems differ only superficially from exam problems. For example, the geometry may be rotated.) These patterns all describe situations in which the student is revising his or her equations or FBDs. The features suggest that students who frequently revise problems which are similar to an exam problem are likely to have difficulty with those problems later on during an exam.

It would be difficult to use the linear regression model to predict performance for students of a future section in Statics. To do so would require that the instruction, assignments, and exams, be identical. This is not likely to be the case, as some of the homework problems are modified each year to prevent copying solutions from the previous offering.

Instead, the patterns and correlations discovered in this paper may be used to guide future offerings of this course. For example, if a student's work contains patterns which indicate difficulty, similar to those found in this study, on the first problem of an assignment or on a problem which is similar to one that will appear in a future exam, the instructor can provide targeted materials for that student to address that difficulty. Furthermore, the results here indicate which problems have a strong bearing on students' performance. For example, students who seemed to have difficulty constructing a FBD on problem five of homework eight typically did not perform well in the course. This indicates to the instructors of future offerings this course, that more time should be spent in class reviewing how the FBD for this problem should be constructed.

8. CONCLUSION

We have presented an application of data mining techniques to educational data extracted from a novel environment. We have given undergraduate Mechanical Engineering students Livescribe^{TM}digital pens with which they completed all their coursework. These pens record students' handwriting as time-stamped pen strokes enabling us to not only analyze the final image, but also the sequence in which it was written.

We developed a novel representation of students' handwritten work on an assignment which characterizes the sequence of actions the student took to solve that problem. This representation comprises an alphabet of 49 canonical actions that a student may make when solving his or her homework assignment. Each action is characterized by its duration, problem number, and semantic content. This representation allows us for the first time, to apply traditional data mining techniques to sequences of students' handwritten problem solutions.

We assigned these sequences into top- and bottom-performing groups according to performance on each sequence's most relevant exam. The most relevant exam for a sequence from a particular homework assignment is the exam which occurs most recently after that homework assignment was due. Sequences from students who performed in the top third of the class on that assignment comprise the top-performing group and sequences from students who performed in the bottom third comprise the bottom-performing group. We applied a differential data mining technique to the sequences from the students in each of these groups and identified patterns that are more frequently exhibited by one group than the other.

These patterns serve as the basis for features used to train a linear regression model to predict students' performance in the course. This model achieves an R^2 of 0.34. Furthermore, the underlying parameters of this model provide valuable insights as to which of the patterns best correlate with performance. From these best-correlating patterns, we have manually identified high-level cognitive behaviors exhibited by the students. These behaviors provide insight as to when students may be experiencing difficulty in the course. These techniques may be applied in future sections of this course to identify when students are having difficulty in class, enabling the instructor to rapidly address those difficulties.

9. REFERENCES

[1] R. Agrawal and R. Srikant. Mining sequential patterns. In *Data Engineering, 1995. Proceedings of the Eleventh International Conference on*, pages 3–14. IEEE, 1995.

[2] J. L. Antonek. Interactive homework: Linking parents to the foreign language learning of their children. In *Dissertation Abstracts International*, volume 57, 1996.

[3] J. Ayres, J. Flannick, J. Gehrke, T. Yiu, et al. Sequential pattern mining using a bitmap representation. In *Conference on Knowledge Discovery in Data: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, volume 23, pages 429–435, 2002.

[4] S. Bai and S. Bai. The maximal frequent pattern mining of dna sequence. In *Granular Computing, 2009, GRC '09. IEEE International Conference on*, pages 23 –26, 2009.

[5] N. K. Bowen and G. L. Bowen. The mediating role of educational meaning in the relationship between home academic culture and academic performance. In *Family Relations*, volume 47, pages 45–51, 1998.

[6] H. Cooper, J. Robinson, and P. Erika. Does homework improve academic achievement? *Review of Educational Research*, 76, 2006.

[7] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[8] R. Deslandes, E. Royer, P. Potvin, and D. Leclerc. Patterns of home and school partnership for general and special education students at the secondary level. In *Exceptional Children*, volume 65, pages 496–506, 1999.

[9] Y. Freund and R. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37. Springer, 1995.

[10] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. In *SIGKDD Explorations*, 2009.

[11] L. Han-lung, T. Stahovich, and J. Herold. Automatic handwritten statics solution classification and its applications in predicting student performance. In *Proceedings of the American Society for Engineering Education Annual Conference*, 2012.

[12] J. Herold and T. Stahovich. Characterizing students' handwritten self-explanations. In *Proceedings of the American Society for Engineering Education Annual Conference*, 2012.

[13] J. Kinnebrew and G. Biswas. Identifying learning behaviors by contextualizing differential sequence mining with action features and performance evolution. In *Proceedings of the Fifth International Conference on Educational Data Mining*, 2012.

[14] J. Mostow, J. Gonzalez-Brenes, and B. H. Tan. Learning classifiers from a relational database of tutor logs. In *Proceedings of the Fourth International Conference on Educational Data Mining*, 2011.

[15] S. Oviatt, A. Arthur, and J. Cohen. Quiet interfaces that help students think. In *Proceedings of the 19th annual ACM symposium on User interface software and technology*, pages 191–200, 2006.

[16] K. Pezdek, T. Berry, and P. A. Renno. Children's mathematics achievement: The role of parents' perceptions and their involvement in homework. In *Journal of Educational Psychology*, volume 94, pages 771–777, 2002.

[17] C. Romero, J. Romero, J. Luna, and V. S. Mining rare association rules from e-learning data. In *Proceedings of the Third International Conference on Educational Data Mining*, 2010.

[18] K. Singh, M. Granville, and S. Dika. Mathematics and science achievement: Effects of motivation, interest, and academeic engagement. In *Journal of Educational Research*, volume 95, pages 323–332, 2002.

[19] T. Van Arsdale and T. Stahovich. Does neatness count? what the organization of student work says about understanding. In *Proceedings of the American Society for Engineering Education Annual Conference*, 2012.

[20] L. Ye and E. Keogh. Time series shapelets: a new primitive for data mining. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 947–956. ACM, 2009.

Predicting Future Learning Better Using Quantitative Analysis of Moment-by-Moment Learning

Arnon Hershkovitz
Teachers College, Columbia University
ah3096@columbia.edu

Ryan S.J.d. Baker
Teachers College, Columbia University
baker2@exchange.tc.columbia.edu

Sujith M. Gowda
Worcester Polytechnic Institute
sujithmg@wpi.edu

Albert T. Corbett
Carnegie Mellon University
corbett@cmu.edu

ABSTRACT

In recent years, student modeling has been extended from predicting future student performance on the skills being learned in a tutor to predicting a student's preparation for future learning (PFL). These methods have predicted PFL from a combination of features of students' behaviors related to meta-cognition. However, these models have achieved only moderately better performance at predicting PFL than traditional methods for latent knowledge estimation, such as Bayesian Knowledge Tracing. We propose an alternate paradigm for predicting PFL, using quantitative aspects of the moment-by-moment learning graph. This graph represents individual students' learning over time and is developed using a knowledge-estimation model which infers the degree of learning that occurs at specific moments rather than the student's knowledge state at those moments. As such, we analyze learning trajectories in a fine-grained fashion. This new paradigm achieves substantially better student-level cross-validated prediction of student's PFL than previous approaches. Particularly, we find that learning which is spread out over time, with multiple instances of significant improvement occurring with substantial gaps between them, is associated with more robust learning than either very steady learning or learning characterized by a single "eureka" moment or a single period of rapid improvement.

Keywords

Moment-by-moment learning graph, preparation for future learning, student modeling.

1. INTRODUCTION

In recent years, there has been increasing emphasis in learning sciences research on helping students develop robust understanding that supports a student in achieving preparation for future learning (PFL) (cf. [9,15,17,26]), with evidence suggesting that differences in the design of educational experiences can substantially impact PFL [11,28]. Multiple approaches have now been found to be successful at supporting PFL. For example, learning-by-teaching when implemented with the use of "teachable agents", computer characters that the student have to teach during the learning process, has been shown to support PFL [11,26,28]. Another approach shown to support PFL is the use of invention activities, during which students are asked to "invent" representation of a given problem (e.g., variance of a data set) [9,25-26].

Given the existence of methods that can support PFL, there is increasing potential to enhance individualization within computer-based learning environments to optimize not just learning of the material being taught (cf. [10,12,24]), but PFL as well. However, individualization of this nature depends on student models that can effectively infer PFL.

In the last two years, approaches that can infer PFL and other forms of robust learning have begun to emerge, but these approaches are still in their early stages, and are only modestly better than simply assessing student knowledge. In specific, models that leverage data on metacognitive and motivational aspects of student behavior (e.g., off-task, help-avoidance) have achieved cross-validated correlations about 0.05-0.1 higher than classical knowledge models (e.g. Bayesian Knowledge Tracing) to both PFL and transfer tests [4-5]. In addition, retention (another aspect of robust learning) has been effectively predicted using inferences of memory decay during periods of non-practice (e.g. forgetting; [16,29]).

In this paper, we propose an alternate method for predicting PFL more precisely than the meta-cognitive/motivational behavior approach proposed in [4]: using quantitative aspects of the Moment-by-Moment Learning Graph. Similar to the classic learning curve (cf. [19,22]), a Moment-by-Moment Learning Graph (MBMLG) represents the probability that learning has occurred at a specific moment [3], for a given student and a given Knowledge Component (KC)/skill, at each step of the learning process. These probabilities are calculated based on a machine learned model that smoothes probabilities calculated using the probability that the student has learned the skill up to the point of a specific step, and the probability of their future actions given the probability that they learned the skill at that problem step.

Earlier work (discussed in greater detail in Section 3.2) suggests that visual interpretations of the patterns of the MBMLG correlate to PFL [6]. This earlier work used human coders to interpret the visual characteristics of the MBMLG. In this work, we study whether an automated approach – based on quantitative analysis of features of the MBMLG – inspired by this earlier work can improve the prediction of PFL.

2. DATASET

We use attributes of the form of individual student's MBMLG to predict student preparation for future learning. We do so in a combined data set from three studies, in total comprising 181 undergraduate and high-school students who used an intelligent tutoring system to learn Genetics. The students enrolled in Genetics courses at Carnegie Mellon University, or in high school biology courses in Southwestern Pennsylvania.

Study 1 (College Undergraduates, Three-Factor Cross). 72 undergraduates enrolled in a Genetics course or in an Introductory Biology course at Carnegie Mellon University were recruited to participate in the study for pay, at a point in the semester where the tutor software was relevant to their classroom learning. The 72 students completed a total of 22,885 problem solving attempts across a total of 10,966 problem steps in the tutor.

Study 2 (College Undergraduates, Gene Interaction). 53 undergraduates enrolled in a Genetics course or in an Introductory Biology course at Carnegie Mellon University were recruited to participate in the study for pay, at a point in the semester where the tutor software was relevant to their classroom learning. The 53 students completed a total of 33,643 problem solving attempts across a total of 22,126 problem steps in the tutor.

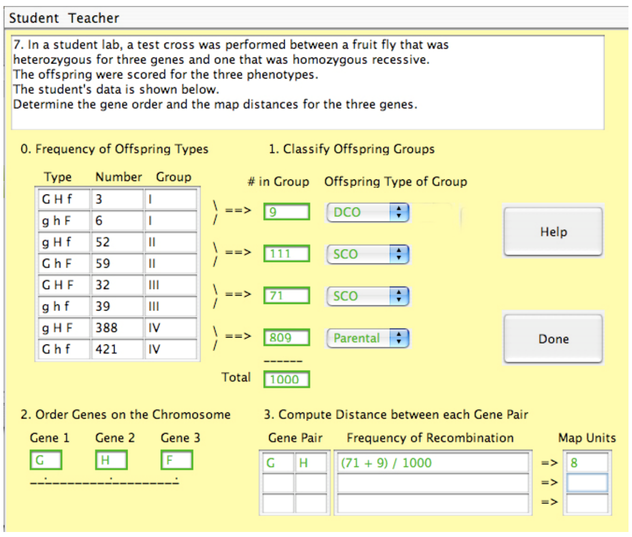
Study 3 (High school students, Three-Factor Cross). 56 high school students who were enrolled in high school biology courses used the tutor. The students were recruited to participate in the study for pay through several methods, including advertisements in a regional newspaper and recruitment handouts distributed at two area high schools. The 56 students completed a total of 21,498 problem solving attempts across a total of 9,204 problem steps in the tutor.

2.1 Learning System and Learning Activity

The data used in this paper was drawn from student use of the Genetics Cognitive Tutor [14]. This tutor consists of 19 modules that support problem solving across a wide range of topics in genetics (Mendelian transmission, pedigree analysis, gene mapping, gene regulation and population genetics). Various subsets of the 19 modules have been piloted at 15 universities in North America.

This study focuses on two of these tutor modules. One employs a gene mapping technique called a Three-Factor Cross. The tutor interface for this reasoning task is displayed in Figure 1. In this technique two organisms are crossed (two fruit flies in the example) and the resulting distribution of offspring phenotypes is analyzed to infer the order of three genes on the chromosome and the relative distances between the three pairs of genes.

The other module, Gene Interaction and Epistasis, engages students in extending basic Mendelian transmission to two genes. In this task, displayed in Figure 2, students cross three true-breeding strains, perform intercrosses, and based on offspring phenotype frequencies, infer the genotypes of the true-breeding strains and each of the offspring phenotypes.



7. In a student lab, a test cross was performed between a fruit fly that was heterozygous for three genes and one that was homozygous recessive.
The offspring were scored for the three phenotypes.
The student's data is shown below.
Determine the gene order and the map distances for the three genes.

0. Frequency of Offspring Types

Type	Number	Group
G H f	3	I
g h F	6	I
g H f	52	II
G h F	59	II
G H F	32	III
g h f	39	III
g H F	388	IV
G h f	421	IV

1. Classify Offspring Groups

# in Group	Offspring Type of Group
9	DCO
111	SCO
71	SCO
809	Parental
Total 1000	

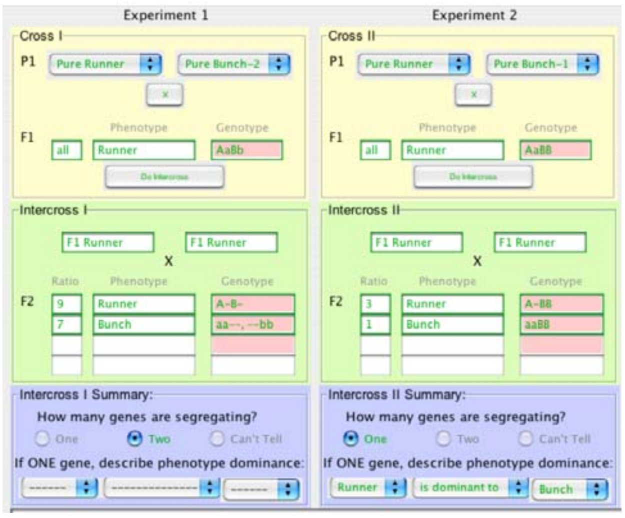
2. Order Genes on the Chromosome

Gene 1	Gene 2	Gene 3
G	H	F

3. Compute Distance between each Gene Pair

Gene Pair		Frequency of Recombination		Map Units
G	H	(71 + 9) / 1000	=>	8
			=>	
			=>	

Figure 1. Screenshot from the Three-Factor Cross lesson of the Genetics Cognitive Tutor



Experiment 1 — Cross I: P1 Pure Runner × Pure Bunch-2; F1: all, Phenotype Runner, Genotype AaBb. Intercross I: F1 Runner X F1 Runner; F2: Ratio 9, Phenotype Runner, Genotype A-B-; Ratio 7, Phenotype Bunch, Genotype aa--, --bb. Intercross I Summary: How many genes are segregating? One [✓ Two] Can't Tell. If ONE gene, describe phenotype dominance: ------ -------------- ------

Experiment 2 — Cross II: P1 Pure Runner × Pure Bunch-1; F1: all, Phenotype Runner, Genotype AaBB. Intercross II: F1 Runner X F1 Runner; F2: Ratio 3, Phenotype Runner, Genotype A-BB; Ratio 1, Phenotype Bunch, Genotype aaBB. Intercross II Summary: How many genes are segregating? [✓ One] Two Can't Tell. If ONE gene, describe phenotype dominance: Runner is dominant to Bunch

Figure 2. Screenshot from the Gene Interaction lesson of the Genetics Cognitive Tutor

2.2 Design

The studies were conducted in computer clusters at Carnegie Mellon University. All students attended study sessions on two consecutive days; in studies 1 and 2, each of these lasted 2 hours, while in study 3, each lasted 2.5 hours. All students engaged in Cognitive Tutor-supported activities for about one hour in each of the two sessions. In studies 1 and 3 all students completed standard Three-Factor Cross problems, as depicted in Figure 1, in both sessions, while in study 2 all students completed standard Gene Interaction problems, as depicted in Figure 2, in both sessions.

During the first session of each study, some students were assigned to complete other cognitive-tutor activities designed to support deeper understanding; however, no significant differences were found between conditions for PFL or any other robust learning measure (this is reported for study 1 in [14]), so in this analysis we collapse across the conditions and focus solely on student behavior and learning within the standard problem-solving Cognitive Tutor activities.

All students completed a problem-solving pre-test at the beginning of the first session, and a problem-solving post-test immediately following the Cognitive Tutor activities in the second session. Following the problem-solving post-test in the second session, students also completed a transfer test and preparation-for-future-learning (PFL) test. Finally, students in studies 1 and 2 returned a week later to complete a problem-solving retention test. Within this paper, we focus all analysis on the PFL test, as a particularly strong indicator that student learning is robust (cf. [9]). A PFL prediction model was built on a dataset combining the three studies.

2.3 PFL Test

This study examines student performance on preparation-for-future-learning problem-solving tests. By definition, the reasoning in each of these two tests is related to solving Three-Factor Cross or Gene Interaction problems, respectively, but is sufficiently more complicated that a student could not be expected invent a solution method by direct transfer, and certainly not in a short period of time. Consequently, each of the PFL tests incorporated instructional text on the required reasoning, which students read prior to problem solving. The PFL tests were designed in collaboration between biology experts and a cognitive scientist (the fourth author).

In the Three-Factor Cross studies, students were asked to solve parts of a four-factor cross problem; the PFL test presented a 2.5-page description of the reasoning in a four-factor cross experiment, then asked students to solve some elements of a four-factor cross problem: identifying the middle genes, identifying all the offspring groups with a crossover between two specific genes and finding the map distance between those two genes.

In the Gene Interaction study, students were asked to reason about gene regulation problems. In these problems, three genes, an operator, a structural gene and a regulatory gene, act together to control DNA transcription. The test presented a 1.5 page description of several gene regulatory systems, then asked student to reason about the impact of dominant and recessive alleles of the component genes on transcription.

PFL tests were completed by all students in the three studies, with an average percent correct of 0.89 (SD=0.15), 0.74 (SD=0.24), and 0.66 (SD=0.28) for Study 1, Study 2, and Study 3, respectively.

3. MOMENT-BY-MOMENT LEARNING GRAPH

3.1 Construction of the Graph

The construction of the Moment-By-Moment Learning Graph (MBMLG) is based on a three-phase process, which first infers moment-by-moment learning using data from the future, then infers the same construct without data from the future, and then integrates across inferences over time to create a graph.

The first step is to infer moment-by-moment learning using data from the future, based on an approach first proposed by [2]. To obtain this inference, a Bayesian Knowledge-Tracing (BKT) model [13] is used to calculate the probability that the student knows a specific skill at a specific time, based on the student's history of success on problems or problem steps involving that skill. The BKT model is updated every time the student responds to a problem step, based on the correctness of the response, allowing for an aggregate estimate of student knowledge over time.

Then, the estimation of student knowledge and the parameters of the BKT model are combined using Bayesian formulas (discussed in mathematical detail in [3]), to infer the probability that a student learned a skill or a concept at a specific step during the problem-solving process, by looking at the probability of future actions if the student had learned the skill at this point. This probability is referred to as P(J) (J stands for "just learned"). That is to say, instead of assessing the probability that a skill is known by the time the student reaches the Nth step that involves that skill, the model assesses the probability that the skill was learned between time N-1 and time N. At an intuitive level, high values of P(J) are seen when a student's performance shifts from being mostly incorrect to mostly correct, but precise values are obtained using current estimates of the probability the student knows the skill, along with model estimates of the probability of correct answers being due to guessing, and incorrect answers being due to slipping or carelessness. This model uses information on past, current, and future performance, to predict the probability that learning occurred during each step of the student's work within the computer-based learning environment.

Once these predictions have been obtained, a machine-learned model is built, using a set of features of student data (such as the recent history of help and errors on this skill, and time taken on the current and recent attempts) to predict P(J) values based on past and current information only. Within the work presented here, the same feature set as was used for the Cognitive Tutor in [3] was used. The list of features inputted into the machine learning algorithm is:

- Assessments of correctness:
  - Percent of all past problems that were wrong on this KC.
  - Total number of past problems that were wrong on this KC.
  - Number of last 5 problems that were wrong.
  - Number of last 8 problems that were wrong.
- Measurements of time:
  - Time taken (SD faster/slower than average across all students).
  - Time taken in last 3 actions (SD off average) Time taken in last 5 actions (SD off average)
  - Total time spent on this KC across all problems
  - Time since the current KC was last seen.
- Data on hint usage:
  - First response is a help request.
  - Bottom-out hint is used.
  - Number of last 8 problems that used the bottom-out hint.
  - Number of last 5 problems that included a help request.
  - Number of last 8 problems that included a help request.
- Other measurements:
  - Total problems attempted in the tutor so far.
  - Total practice opportunities on this KC so far.

- o Response is chosen from a list of answers (Multiple choice, etc).
- o Response is filled in (No list of answers available).

This model serves the joint purposes of enabling the model to be used for eventual intervention, and smoothing the sometimes extreme values of P(J) that can be obtained when the BKT model's parameters for guessing and slipping are low. The model used here is built using linear regression with M5' feature selection [30], in RapidMiner Version 4.6 [20]. To validate the generalizability of our models, 6-fold cross-validation at the student level was used (i.e., detectors are trained on five groups of students and tested on a sixth group of students). By cross-validating at this level, we increase confidence that detectors will be accurate for new students.

The goodness of the models was assessed using the Pearson correlation coefficient between the training labels of P(J) for each opportunity to learn each KC, and the values predicted for P(J) for the same opportunity by the machine-learned models. As both set of values are quantitative, and there is a one-to-one mapping between training labels and predicted values, linear correlation is a reasonable metric.

The P(J) model achieved solid correlations to the training labels under 6-fold student-level cross-validation, with values of 0.68 for Study 1 (college genetics 3-factor cross lesson; reported in [6]), 0.65 for Study 2 (college genetics gene-interaction lesson), and 0.48 for Study 3 (high school genetics 3-factor cross lesson). These values are moderately higher than those seen for P(J) models built for the Middle School Cognitive Tutor or ASSISTments, probably due to the more diverse collection of lessons used in these earlier studies (e.g. [3]). The difference in correlation between the college studies and the high-school study might suggest between-population differences; perhaps the high school students differed more from each other than the college students, all of whom had been accepted and chose to attend the same university.

3.2 Previous Studies: Association with PFL

In prior work, Moment-by-Moment Learning Graphs were created for the Genetics Tutor and then visually analyzed by human coders; the coders examined the graphs and chose for each instance the visual patterns that can be observed in it (either a single pattern or multiple patterns). In specific, seven specific visual patterns of the MBMLG were identified, coded by human coders (achieving high inter-rater reliability), and then those human labels were correlated with scores on a PFL test [7]. In that work, it was found that two patterns of the MBMLG are statistically significantly associated with PFL, specifically (see Figure 3): 1) *Plateau* - three or more sequential problem steps that have significantly higher values for P(J) than the rest of the student's behavior. This form represents students who have steady learning (e.g., steady improvement in performance) during only part of the learning activity. The plateau visual form was found to be negatively associated with PFL (r=-0.27, statistically significant when controlling for multiple comparisons). 2) *Immediate drop* – the first problem step for the skill has a high value for P(J), which then immediately falls to low values for the rest of the learning. Immediate drop most likely represents a student who already knows the relevant skill and simply must transfer it into the learning system, or a student immediately mastering a very easy skill. Immediate drop is positively associated with PFL (r=0.29, statistically significant when controlling for multiple comparisons), suggesting that students who already know a skill are more likely to be prepared for future learning when they start the tutor, or that the over-practice that the tutor represents for these students may be enhancing their preparation for future learning. This suggests the hypothesis that over-practice can lead students to not only develop greater speed of performance [23] and lower probability of forgetting [24], but also to deeper conceptual knowledge required to prepare them for future learning.

Additionally, "Spikiness" in the MBMLG – that is, the extent to which there is a prominent peak in the graph, which might indicate a "eureka" moment (cf. [3,4]) – was shown to be correlated with PFL and as a significant factor in a PFL machine-learned prediction model [5]. These results suggested that the visual or functional form of the Moment-By-Moment Learning Graph can be highly associated with preparation for future learning. However, the results in [7], as they rely upon human labels, are not sufficient for use to improve the automatic adaptive behavior of educational software; also, human labels are not easily available at scale for larger studies. The model developed in [5] uses measures other than the MBMLG, and only simple measures of the MBMLG; hence it does not fully demonstrate the potential of the graph to individually predict PFL. In this paper, we attempt to extend these approaches by assessing the mathematical properties of the MBMLG in an automatic fashion, and developing a model that relies solely on these properties to predict PFL.

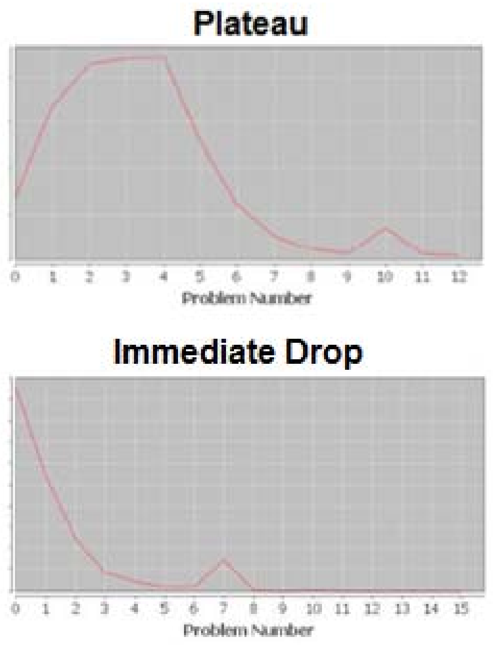


Figure 3. Examples of MBMLG patterns that were found to be significantly related to PFL: plateau (top) and immediate drop (bottom)

4. FEATURE ENGINEERING

In this paper, we attempted to distill quantitative features of the MBMLG to use in automated prediction of PFL. 15 features were computed for each MBMLG, and used as potential predictors of PFL. The full list of features is given here. Features included in the prediction model are highlighted in boldface; in square brackets, a short name is given for each variable, to be used later in the article:

- Average moment-by-moment learning [avgMBML]
- Sum of moment-by-moment learning values [sumMBML]
- Number of opportunities to learn the KC [graphLen]
- **Area under the graph [area]**
- Height of the largest peak [peak]
- Height of the 2nd-largest peak [2ndPeak]
- **Height of the 3rd-largest peak [3rdPeak]**
- First index of the largest peak (index = 1 equals the first step involving the skill, index = 2 equals the second step involving the skill, and so on) [peakIndex]
- First index of the 2nd-largest peak [2ndPeakIndex].
- Distance (in terms of number of problem steps) between the largest and the 2nd-largest peaks [2PeaksDist]
- Distance between the largest and the 2nd-largest peaks, divided by the total number of steps involving the KC [2PeakRelDist]
- Decrease [%] of magnitude from largest to 2nd-largest peak [2PeakDecr]
- Decrease [%] from largest to 2nd-largest peak, divided by the distance (# steps) between them [2PeakRelDecr]
- **Decrease [%] from largest to 3rd-largest peak [3PeakDecr]**
- **Decrease [%] from largest to 3rd-largest peak, divided by the distance (# of steps) between them [3PeakRelDecr]**

5. PFL PREDICTION MODEL

To predict the PFL test results using the 15 MBMLG features, we averaged values of these variables across the sets of genetics problem-solving skills in each of the two tutor modules. Using this data set, we built a model to predict the PFL test from the quantitative attributes of the MBMLGs, using linear regression with forward selection of model features (cf. [21]). The model was validated using student-level leave-one-out cross-validation. In addition, a first pass was conducted prior to model selection where features were eliminated if, taken individually, they had cross-validated correlation below zero. This procedure was adopted as an extra control and over-fitting. This first pass eliminated five variables ([peakIndex], [2ndPeakIndex], [graphLen], [2PeaksDist], [2PeakRelDist]).

Goodness of fit was assessed using the Pearson correlation between the predicted PFL score and the actual score. The best-fitting model has a cross-validated correlation of r=0.532 with actual PFL scores, substantially better than the cross-validated correlations previously found (e.g., [4]) for models based on meta-cognitive and behavioral features (0.360) or models assessing student skill within the software (0.285). The best model is presented in Table 1.

Interestingly, the 3rd-largest learning peak is involved in three of the four variables selected into the best predictive model. In specific, the magnitude of the 3rd-largest learning peak is positively associated with PFL, and large gaps in the size between the largest and 3rd-largest learning peak (measured by *decrease [%] from largest to 3rd-largest peaks*) are positively associated with PFL. These relationships may suggest that single "eureka" moments might indeed be meaningful for robust learning. Having the steepness of the decrease between the largest and to the 3rd-largest peaks (*decrease [%] from largest to 3rd-largest peaks, divided by the distance between them*) with a negative coefficient emphasizes the importance of multiple (though more minor than the most prominent one) learning events which are spread out over time (as opposed to occurring more rapidly). As such, the best pattern appears to be a pattern where the student has multiple substantial moments of learning (at least three) of comparable magnitude, spread out over time.

Table 1. Best-fitting linear regression model predicting PFL

Variable	Coefficient
Area under the graph [area]	-8.459
Height of the 3rd-largest peak [3rdPeak]	+2.634
Decrease [%] from largest to 3rd-largest peaks [3rdPeakDecr]	+0.641
Decrease [%] from largest to 3rd-largest peaks, divided by the distance between them [3rdPeakRelDecr]	-0.296
(Constant)	+0.607

Another finding worth discussing is the negative coefficient of total area under the graph, which indicates that students who have relatively higher learning in the environment have generally lower PFL (when controlling for the other features). While this finding is at a surface-level non-intuitive, it is worth noting that in an effective learning environment, most students learn the skills being taught if they do not already know them. As such, many or most of the students who do not learn the skills being taught already knew the skills to begin with. Indeed, problem-solving pre-test and area are statistically significantly negatively correlated in our study, as shown earlier. These students may have therefore been able to focus on developing more robust learning while using the environment, rather than needing to focus on the exact skills directly taught by the environment.

5.1 Correlations of MBMLG Feature with PFL

Having built the full model, it may be worth examining the types of relationships between the individual MBMLG features and PFL score. In particular, some variables may reverse direction in a complex model. Therefore, in order to understand individual features' relationship to PFL, we computed correlations between each feature the PFL score. In addition, we computed correlation with pre-test scores, in order to explore the possibility that some of these relationships might be explained by prior knowledge. Results are summarized in Table 2 (full names of the variables, along with their shortened names, appear in Section 4, Feature Engineering), and are discussed in this section.

Three features that aggregate overall measures of moment-by-moment learning were found to be significantly negatively correlated with both pre-test and PFL scores; these features are: *average moment-by-moment learning [avgMBML]*, *sum of moment-by-moment learning values [sumMBML]*, and *area under the graph [area]*. That is, low values of learning as reflected in the MBMLG are indicators of high prior knowledge, which is in turn a good predictor of PFL.

The features that measure values of the largest peaks of the MBMLG are significantly negatively correlated with PFL. That is, the larger the values of the graph's highest peaks, the lower the PFL score is. Interestingly, the correlation between the 2nd-largest peak and PFL is stronger than that of the largest peak; and the correlation between the 3rd-largest peak and PFL is stronger than that of the top two largest peaks. It is important to notice that *height of the 3rd-largest [3rdPeak]* is significantly (though mildly) negatively correlated with prior knowledge, while the *height of the 2nd-largest peak [2ndPeak]* is marginally negatively correlated with prior knowledge, and *height of the largest peak [peak]* is not significantly correlated with prior knowledge. Hence, based only on a few meaningful learning events (three, to be more specific), we can conclude that the student was not properly prepared to the learning to begin with, and as such – the student is probably not prepared for future learning as well.

Lastly, the four features that measure the decrease in the graph largest peaks – both absolute and relative to the distant between the peaks – are significantly positively correlated with both pre-test and PFL scores. It is important to first fully understand the meaning of these four features. The larger the absolute decrease between the largest peak values of the graph – measured by *decrease [%] of magnitude from largest to 2nd-largest peaks [2PeakDecr]*, *decrease [%] of magnitude from largest to 3rd-largest peaks [3PeakDecr]* – the more likely it is that there was a single meaningful learning event across the learning process. The higher the value of the relative decrease between the largest peak values – measured by *decrease [%] from largest to 2nd-largest peaks, divided by the distance between them [2PeakRelDecr]*, *decrease [%] from largest to 3rd-largest peaks, divided by the distance between them [3PeakRelDecr]* – the more likely it is that the graph peaks are either different in value from each other, or that they are close to each other. So, these features' positive correlations with PFL suggest that either single learning events, or temporally close multiple learning events are associated with robust learning.

That said, there are two interesting sign-flips observed between the individual features' correlations with PFL scores and their coefficients in the prediction model: the coefficient of the *3rd-largest peak [3rdPeak]* is positive in the model while it is negative when correlation is examined individually; and the coefficient of the steepness of the decrease from the largest to the 3rd-largest peaks *[3rdPeakRelDecr]* is negative in the model while it is positive when correlation is examined individually. Considering these results along with the full model, we might conclude that when controlling for overall learning (*area under the graph [area]*) and for the prominence of the most meaningful learning event *(measured by Decrease [%] from largest to 3rd-largest peaks [3rdPeakDecr])*, another pattern emerges as an indication to PFL, which is having multiple learning events spread over the learning process.

Another interesting finding regarding the individual feature correlations is that *Number of opportunities to learn the KC [graphLen]* is significantly positively correlated with pre-test, but is not significantly correlated with PFL. This might suggest that over-practice within the tutor is not necessarily associated with robust learning; however, practice within the tutor after gaining knowledge in another fashion might be useful (as immediate drop was found to be positively associated with PFL in Baker, Hershkovitz, et al., in press).

Table 2. Correlations between MBMLG features (N=179) and Pre-test, PFL scores; significant results (two-tailed) are boldfaced (* p<0.05, ** p<0.01), marginally significant results (p<0.1) are italicized

Feature	Pre-test	PFL
[avgMBML]	**-0.35****	**-0.48****
[sumMBML]	**-0.19****	**-0.40****
[graphLen]	**0.30****	0.00
[area]	**-0.35****	**-0.48****
[peak]	-0.09	**-0.35****
[2ndPeak]	*-0.13*	**-0.41****
[3rdPeak]	**-0.20****	**-0.44****
[peakIndex]	*-0.15*	-0.09
[2ndPeakIndex]	-0.03	0.03
[2PeaksDist]	*0.15*	0.05
[2PeakRelDist]	*0.14*	0.07
[2PeakDecr]	**0.29****	**0.45****
[2PeakRelDecr]	**0.26****	**0.40****
[3PeakDecr]	**0.35****	**0.49****
[3PeakRelDecr]	**0.21****	**0.38****

6. CONCLUSIONS AND DISCUSSION

In this paper, we present a new method of predicting preparation for future learning (PFL), based on quantitative analysis of the Moment-by-Moment Learning Graph (MBMLG; [2-3]) the patterns of which were previously shown to be associated with

PFL [6]. Overall, we find that using MBMLG features in machine-learned prediction models outperforms previous attempts to predict PFL using BKT parameters and behavioral and metacognitive variables [4-5].

A crucial part of many EDM applications is the feature engineering. In this case, the features we defined and tested were derived from two main streams of literature. First, this work is a natural continuation of previous studies that showed that certain patterns of the MBMLG were strongly associated with PFL, or robust learning in general (cf. [6]). In particular, the presence of *immediate drop* and *plateau* were shown as good indicators of better/poorer PFL, respectively. Hence, the use of measures of the three largest peaks of the graph – the decrease in their values, and the distances between the first and the second. A second relevant line of work is the broader educational research of robust learning. As many studies suggest, learner characteristics have a strong influence on robust learning; of these characteristics, cognitive ability [1,18] can be easily measured via the student model, hence the use of learning measures (which are at the core of the MBMLG).

Another potentially interesting line of future work might be to present MBMLG graphs to teachers and content developers to find irregularities in the learning process (an idea that had previously inspired the creation of *learnograms*, cf. [22]). Teachers and content developers may have insights about the meaning of the MBMLG graphs; they may also find ways to incorporate these graphs in their work to understand their students better.

Overall, our findings suggest that the pattern most associated with a better PFL consists of a process where the student has at least three substantial moments of learning of comparable magnitude, spread out over time. One limitation of the current approach, as it is implemented here, is that PFL prediction is made only after practice has been completed. That is, data cannot be used in real-time like it was used in [4]. However, truncated forms of the MBMLG might be explored for that purpose.

Analyzing the MBMLG qualitatively and using its features to predict PFL is an instance of "discovery with models", that is of using an existing model (in our case, the MBMLG) in a new analysis (predicting PFL). Discovery with models was suggested as a promising EDM approach in [7]. We build on previous studies involving the same model (e.g., [2-3,6]), and advance the potential automatization of the use of the MBMLG as a component in future studies.

7. ACKNOWLEDGMENTS

This research was supported by grants "NSF #DRL-0910188 and IES #R305A090549, and by the Pittsburgh Science of Learning Center, NSF #SBE-0836012. We would also like to thank Adam Goldstein, Neil Heffernan, Lisa Rossi, Angela Wagner, Zakkai Kauffman-Rogoff, and Aaron Mitchell for their help in this study.

8. REFERENCES

[1] Brandsford, J., & Schwartz, D. (1999). Rethinking transfer: A simple proposal with multiple implications. *Review of Research Education, 24*, 61-100.

[2] Alexander, P. A., & Murphy, P. K. (1999). Nurturing the seeds of transfer: A domain-specific perspective. *International journal of educational research*, *31*(7), 561-576.

[3] Baker, R.S.J.d., Goldstein, A., & Heffernan, N. (2010). Detecting the moment of learning. In *Proc. Intelligent Tutoring Systems* (pp. 25-34). Springer Berlin/Heidelberg.

[4] Baker, R. S., Goldstein, A. B., & Heffernan, N. T. (2011). Detecting learning moment-by-moment. *International Journal of Artificial Intelligence in Education*, *21*(1), 5-25.

[5] Baker, R.S.J.d., Gowda, S. M., & Corbett, A.T. (2011). Automatically detecting a student's preparation for future learning: Help use is key. In *Proc. of the 4th International Conference on Educational Data Mining* (pp. 179-188).

[6] Baker, R.S.J.d., Gowda, S., & Corbett, A.T. (2011). Towards predicting future transfer of learning. In *Proc. Artificial Intelligence in Education* (pp. 23-30).

[7] Baker, R.S.J.d., Hershkovitz, A., Rossi, L.M., Goldstein, A.B., & Gowda, S.M. (in press). Predicting robust learning with the visual form of the moment-by-moment learning curve. To appear in *Journal of the Learning Sciences*.

[8] Baker, R. S. J. D., & Yacef, K. (2009). The state of educational data mining in 2009: A review and future visions. *Journal of Educational Data Mining*, *1*(1), 3-17.

[9] Belenky, D. M., & Nokes-Malach, T. J. (2012). Motivation and transfer: The role of mastery-approach goals in preparation for future learning. *Journal of the Learning Sciences*, *21*(3), 399-432.

[10] Cen, H., Koedinger, K., & Junker, B. (2006). Learning factors analysis–a general method for cognitive model evaluation and improvement. In *Proc. Intelligent Tutoring Systems* (pp. 164-175). Springer Berlin/Heidelberg.

[11] Chin, D. B., Dohmen, I. M., Cheng, B. H., Oppezzo, M. A., Chase, C. C., & Schwartz, D. L. (2010). Preparing students for future learning with Teachable Agents. *Educational Technology Research and Development*, *58*(6), 649-669.

[12] Corbett, A.T. (2001). Cognitive computer tutors: Solving the two-sigma problem. In *Proc. User Modeling*, 137-147.

[13] Corbett, A.T., MacLaren, B., Kauffman, L., Wagner, A., & Jones, E. (2010). A Cognitive Tutor for genetics problem solving: Learning gains and student modeling. *Journal of Educational Computing Research*, 42, 219-239.

[14] Haskell, R. E. (2000). *Transfer of Learning: Cognition and instruction*. San Diego, CA: Academic Press.

[15] Jastrzembski, T. S., Gluck, K. A., & Gunzelmann, G. (2006, January). Knowledge tracing and prediction of future trainee performance. In *The Interservice/Industry Training, Simulation & Education Conference (I/ITSEC)*(Vol. 2006, No. -1). National Training Systems Association.

[16] 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.

[17] Li, N., Cohen, W., & Koedinger, K. (2010). A computational model of accelerated future learning through feature

recognition. In *Proc. Intelligent Tutoring Systems* (pp. 368-370). Springer Berlin/Heidelberg.

[18] Mazur, J. E., & Hastie, R. (1978). Learning as accumulation: A reexamination of the learning curve. *Psychological Bulletin*, *85*(6), 1256.

[19] Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., & Euler, T. (2006). YALE: Rapid Prototyping for Complex Data Mining Tasks. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (KDD 2006), 935-940.

[20] Myers, R. H. (1990). *Classical and modern regression with applications* (Vol. 2). Belmont, CA: Duxbury Press.

[21] Nachimas, R., & Hershkovitz, A. (2007). A case study of using visualization for understanding the behavior of the online learner. In *Proceedings of the International Workshop on Applying Data Mining in e-Learning 2007*.

[22] Newell, A., & Rosenbloom, P. S. (1981). Mechanisms of skill acquisition and the law of practice. *Cognitive skills and their acquisition*, 1-55.

[23] Pavlik, P. I., & Anderson, J. R. (2010). Practice and Forgetting Effects on Vocabulary Memory: An Activation-Based Model of the Spacing Effect. *Cognitive Science*, *29*(4), 559-586.

[24] Roll, I., Aleven, V., & Koedinger, K. R. (2009). Helping students know 'further'-increasing the flexibility of students' knowledge using symbolic invention tasks. In *Proc. of the Cognitive Science Society* (pp. 1169-74).

[25] Schwartz, D. L., & Martin, T. (2004). Inventing to prepare for future learning: The hidden efficiency of encouraging original student production in statistics instruction. *Cognition and Instruction*, *22*(2), 129-184.

[26] Segedy, J. R., Kinnebrew, J. S., & Biswas, G. (2011). Modeling learner's cognitive and metacognitive strategies in an open-ended learning environment. In *AAAI Fall Symposium on Advances in Cognitive Systems, Arlington, VA*.

[27] Tan, J., & Biswas, G. (2006). The role of feedback in preparation for future learning: A case study in learning by teaching environments. In *Proc. Intelligent Tutoring Systems* (pp. 370-381). Springer Berlin/Heidelberg.

[28] Wang, Y., & Heffernan, N. T. (2011). Towards Modeling Forgetting and Relearning in ITS: Preliminary Analysis of ARRS Data. In *Proc. Fourth International Conference on Educational Data Mining* (pp. 351-352).

[29] Wang, Y., & Witten, I. H. (1996). Induction of model trees for predicting continuous classes.

InVis: An Interactive Visualization Tool for Exploring Interaction Networks

Matthew W. Johnson
University of North Carolina at Charlotte
Charlotte, NC
mjokimoto@gmail.com

Michael Eagle
University of North Carolina at Charlotte
Charlotte, NC
maikuusa@gmail.com

Tiffany Barnes
North Carolina State University
Raleigh, NC
tmbarnes@ncsu.edu

ABSTRACT
We introduce *InVis*, a novel visualization technique and tool for exploring, navigating, and understanding user interaction data. *InVis* creates a interaction network from student-interaction data extracted from large numbers of students using educational systems, and enables instructors to make new insights and discoveries about student learning. Here we present our novel interaction network model and *InVis* tool. We also demonstrate that *InVis* is an effective tool for providing instructors with useful and meaningful insights to how students solve problems.

1. INTRODUCTION
The advancement of personalized learning has been declared a Grand Challenge by the National Academy of Engineers [12]. With increasing use of the web in education and learning management systems the amount of data that can be brought to bear on this important challenge is growing rapidly. For example, the PSLC DataShop, a repository for educational data, has collected logs from over 42,000 students from different tutors with a wide range of topics, from algebra to Chinese [8]. However, such large datasets can be unwieldy, and deciding just how to use them to improve learning is a challenge, as illustrated by the emergence of the new field and conference on Educational Data Mining.

We have developed a visualization tool, called *InVis*, to enable educators to interactively explore our novel interaction network representing the interactions students perform in problem-solving environments. *InVis* displays student behavior across an entire class, enabling educators to develop insights from common strategies and mistakes that groups of students apply in a software tutoring environment. While this work will concentrate on data from computer-aided instructional environments, we have also used *InVis* for exploring user interactions in games and plan to use it for other applications that record sequential interactions.

Educational data mining tools require "good visualization facilities to make their results meaningful to educators and e-learning designers." [15]. If done well, visualizing problem-solving interaction logs can provide insight into how users solve problems and what errors they encounter; and provide more information than a purely summative approach.

To address these issues, we have developed a model that represents the interactions of large groups of users in problem-solving environments, called the Interaction Network model. We developed *InVis* to allow educators to visualize and interactively explore interaction networks to better understand student interactions in problem-solving software tutors. We used three methods of evaluation: 1) a guidelines review, where we compare *InVis* to the commonly accepted visualization guidelines; 2) a set of case-study like success stories with expert users, and 3) a summative usability study, where educators explored data from a university-level logic tutor using guided tasks and completed a validated usability scale, and reflected on their experience through a qualitative survey. Our 'triangulated evaluation' is inspired by Plaisant's aptly named paper, the 'Challenges of Visualization Evaluation', where she described the difficulties of performing evaluations with tools that can "answer questions you didn't know you had." [13] Since there is no single proven technique for visualization evaluation, Plaisant recommends using several complementary methods that can mitigate the weaknesses of single techniques used alone. Our guidelines review shows how *InVis* adheres to commonly accepted visualization guidelines. In the case-studies, educators were able to generate and confirm hypotheses, and discover insights into their data. The results of the usability study showed that educators were able to complete tasks at 85% accuracy with minimal training time.

We show that a Interaction Network is an effective and reasonable description of student interaction data from computer-aided instructional software for problem solving. We also show that *InVis*, an effective interactive visualization designed for visualizing Interaction Networks, can be made and can provide useful and meaningful insights to student behavior in software tutors.

1.1 Related Work
InVis was inspired by work in exploring student data on solving logic proofs. In 2007, Barnes and Stamper created a frequency-annotated behavior graph and a method to convert it into a Markov decision process (MDP) from student logic proof data, and loaded this data into the GraphViz visualization tool to visualize student problem solving sequences [1]. From this exploration, an experienced logic instructor discovered surprising student behaviors and unexpected difficulties with the logic tutor interface. The instructor also found that some students demonstrated expert-like solutions, and that students did not flounder as much as expected while solving problems.

Student tutor-log data sets are large (representing hundreds of problem attempts with hundreds of problem states) and teachers, who are not necessarily savvy with graphs, spreadsheets, or statistics, need support in navigating these large domain models to learn about student behavior. Ben-Naim and colleagues [3] have developed an authoring tool that allows teachers to create simulators for science and explore small graphs of student actions in the simulator. However, this visualization is restricted to teacher-created states, where teachers label a step in the simulator as one of interest. It is not fully derived from student data and does not facilitate exploration of a large, diverse dataset from other tutors.

SpaceTree [14] is software that might enable educators to explore our static interaction networks more interactively than graph visualization tools like GraphViz [7] and Gephi [2]. However, our networks are not always trees and can contain cycles and loops. The Spacetree layout also particularly highlights the children of a node, while we are interested in the full path from start to finish for problem-solving sequence, and in seeing a whole set of student behavior at once, to provide an overview and support pattern-finding.

CourseVis is a visualization tool produces graphical representations of student tracking data collected by a Content Management System, and helps teachers gain an understanding of how students are behaving in their online courses. In CourseVis, the focus is on the behavior of a student over the course of an entire system, where as in our work the focus is more fine grained, as we are interested in the behavior of students in single problems. CourseVis does support some techniques to look at student performance but the focus is on visualizing knowledge components and assessment performance, not problem-solving behavior as in our work [10].

In VisMod students are provided with a visualization tool for representing and interacting with their own student-model allowing students to develop their meta-cognitive skills [19]. The focus of this tool is not the behavior of the students but instead what the students think about their own behavior.

TADA-Ed is a tool designed for mining educational data generated from digital tutors, much like our work. TADA-Ed's focus is on visualizing the results of several data-mining techniques, such as k-means clustering and decision trees applied to educational data [11]. Our work is different in that our focus is on student problem solving behavior.

2. INTERACTION NETWORK MODEL

In this section we describe our novel Interaction Network Model, which we believe can be used across many domains to describe sequences of user interactions in software interfaces. This is the first work to model student problem-solving in a tutoring environment in this way, and to use the Interaction Network model to better understand student tutor-log interactions. As we describe later, we've applied the Interaction Network model to two widely different domains (logic proofs and drawing pictures on a grid) to understand student behavior. The novelty lies in the combination of several ideas: 1) how to define a 'state' and 'action', 2) how to combine multiple sequences into one Interaction Network, and 3) how to represent actions that are valid within the interface (e.g. clicking an active button) but not within the problem-solving context (e.g. clicking the wrong button).

We model a solution attempt as a sequence of states (vertices) and actions (edges). *Case* refers to a single student's solution attempt.

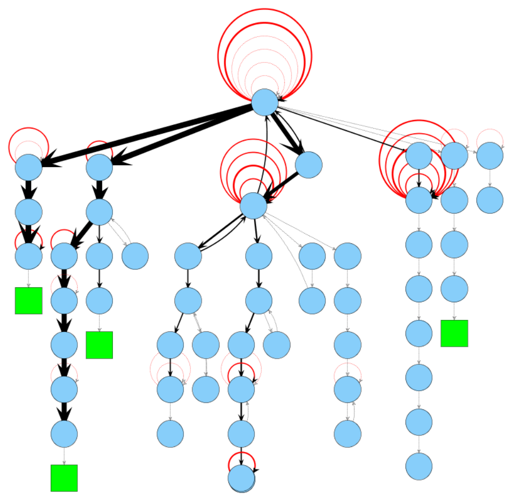


Figure 1: An example of a small Interaction Network.

The interaction network for a problem is the conjoining set of all of its solution attempts. *State* describes the state of the software environment, representing enough information so the program's state could be regenerated in the interface. *Actions* describe user interactions and their relevant parameters, to move a user from one state to the next. We also store the set of all cases who visited any particular state-vertex or action-edge, allowing us to count frequencies and connect case specific information to the interaction network representation. This representation results in a connected, directed, labeled multi-graph with states as vertices, actions as directed-edges to connect the states, and cases that provide additional information about states and edges. (See figure 1.)

This representation allows to us build a interaction network model from any system that logs interactions in state, action, resulting-state tuples. For the intelligent tutoring system community, we encourage the use of logging standards such as the PSLC's Datashop [8]. We recommend preservation of the action's parameters and results, since they are useful for labeling the visualization.

To build the interaction network for a problem we combine the interaction sequences, from each case into a single network, where states are combined when they are considered equal. In different tutors and interfaces, two states could be considered equal as long as the screen looks the same, or all the same actions have been performed, regardless of order, but in other cases, states arrived at by taking the same actions in any order could be considered distinct.

InVis will handle either case, and the logic data in our experiments preserved order. Frequency information and information about which cases have visited, is embedded into the edges and vertices. This results in a network graph which shows the interactions of a large number of users in a relatively small space.

The basic example is whether or not order matters. For example, in the Deep Thought data should the state A,B be distinct from the state B,A? An order-matters approach results in more distinct program states, and thus more vertices. However, preserving the order of premises derived increases the information of the visualization by making the order of steps a student takes more obvious. By contrast, an unordered approach reduces the number of states and reduces the size of the overall graph-layout. While the unordered graph provides a more generalized view, it can be harder to follow the precise steps of an individual case.

2.1 Visual Representation & Graph Layout

A graph is a natural representation for our Interaction Network model. However, there are still a wide variety of graph layouts, and the primary visualization view should be one that is easiest for the novice user to understand. *InVis* uses a tree-like graph layout to present Interaction Networks. The root node is the starting state (the problem 'givens'), and is placed at the top of the view, with student interactions branching downward. This layout makes it easy to follow a student's individual solution-path, as the vertex depth effectively preserves the number of (non-error) steps in the solution-attempt.

State vertices can be labeled with the entire state description, or simply the result of the latest interaction, since reading sequential states should reveal the entire state description. If they exist in the data, final or goal states are represented with a different color and shape, a green rectangle here. Each edge is labeled by its action, but not its parameters, to keep it more readable. Edge thickness is determined by the frequency of observed interactions, with most-frequent edges being thickest. *InVis* uses JUNG [18] to efficiently place nodes in a graph layout.

2.2 Modeling Program States

We have successfully built interaction networks from a variety of sequence oriented interaction data. Here we describe two systems and concentrate on how we modeled the state description.

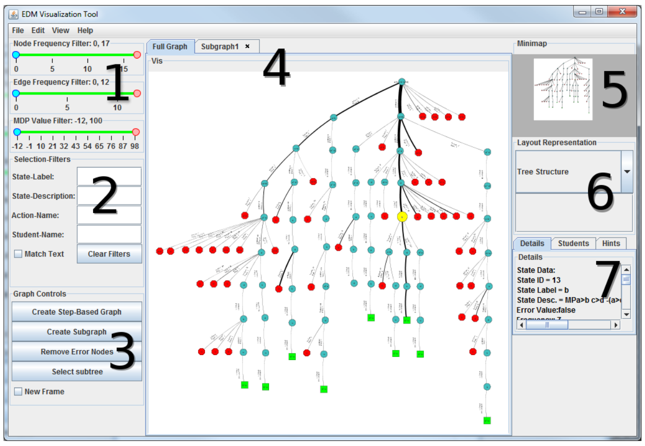
2.2.1 *BeadLoom Game*

The BeadLoom Game (BLG) [4] is a game extension of the CSDT: Virtual Bead Loom, an educational tool for teaching Cartesian coordinates to middle school students. The BLG added game elements in order to increase motivation and learning [4]. In the BeadLoom Game, players place beads in a 41x41 Cartesian grid using six different tools and an undo command. All actions take a color (12 different options) parameter. The loom starts empty and once beads are added they cannot be removed, aside from the undo action. However, beads can be overwritten by future actions. The goal of the game is to create a target image with the tools available. Figure 4 shows an example from the BeadLoom Game.

To gain a better understanding of the BLG data, we used the *InVis* to explore player solutions. The state representation is a 41x41 array containing the 12 color values. Actions are represented by the six bead-placement tools and their parameters. We also store the set of all cases who visited any particular state-vertex or action-edge. We use an image depicting the player's state as the state label. The BLG does not have error data, meaning users cannot submit invalid parameters as actions. However, users are able to undo previous actions, which we can interpret as an unintended action.

2.2.2 *The Deep Thought Tutor*

Deep Thought is a propositional logic tutor in which students are tasked with performing first-order logic proofs [6]. Students are given a set of premises and a conclusion; students must use basic logic axioms to prove the conclusion. As the student works through the proof, the tutor records each interaction. We model the application of axioms as the actions. We model the state of the logic tutor as the conjoined set of each premise and derived proposition.



Details
State Data:
State ID = 13
State Label = b
State Desc. = MPa>b c>d -(a>
Error Value:false

Figure 2: An example screenshot of the default view in *InVis*.

For example a student starts at state $A \vee D, A \rightarrow (B \wedge C), \neg D \wedge E$, where each premise is separated by a comma. The student performs the interaction $SIMP(\neg D \wedge E)$, applying the simplification rule of logic to the premise $\neg D \wedge E$ and derives $\neg D$. This leads to the resulting-state of $A \vee D, A \rightarrow (B \wedge C), \neg D \wedge E, \neg D$.

Errors are actions performed by students that are illegal operations of logic and the tutor, so the student cannot exist in that error-state. As a result they are returned to their previous state. In the case where a student makes an error, the action is marked as an error. For example: The student is in state $A \vee D, A \rightarrow (B \wedge C), \neg D \wedge E, \neg D$. The student performs the interaction $SIMP(A \vee D)$ in an attempt to derive A. The resulting-state would remain $A \vee D, A \rightarrow (B \wedge C), \neg D \wedge E, \neg D$, the log-file would mark this edge as an error.

3. DESIGN OF THE VISUALIZATION

Here we address the design of the interactive visualization, *InVis*. The tool was designed with the visual information-seeking mantra [16] as our guide, based on two reasons. First, Schneiderman states from his own extensive experience that an effective visualization contains those elements, and second Cairns has shown the sheer number of examples where it has been used to guide visualization design [5]. Thus we feel this to be a reasonable approach to developing such an interactive visualization. Note our visualization is just one method for designing an interactive tool for exploring Interaction Networks, and later we provide evidence that it is effective at its goal, but many different types of implementations for exploring Interaction Networks are likely to exist.

Referring to figure 2 we will explain the major features of *InVis*.

1. Frequency filters allows the user to filter edges and nodes based on the frequency. The filter removes nodes or edges based on the range. Frequency is a useful metric for investigating the behavior of a large number of users. Nodes and edges with high frequency identify common behaviors, while low frequency nodes represent less common behaviors. Because frequency is an intuitive domain-independent metric we choose to add this filter to the default view.

2. The selection filters, allow users to select states, actions, or cases by entering a search string; the user can select to match with *contains* or *direct match*. This provides an easy measure

to select large numbers of specific states, and is powerful when combined with the subgraph extraction feature.

3. Graph controls allow users to create subgraphs, which are then loaded into a separate tab. The user can also remove the hanging error nodes from the current graph, if they are interested in correct behaviors more than errors. Creating a subgraph, copies all of the currently selected nodes to a new tab (in the interface) and re-applies the graph-layout to the subgraph. This can be used to clear up clutter. By first selecting desired states, edges, or cases with the selection filters, and then generating a subgraph a wide range of options for exploring the data is possible. An example of the subgraph generation is shown in figure 3.

4. The interaction network is displayed here. This panel has mouse controls for panning, zooming, and selecting, controlled by right-click, mouse-wheel, and left-click respectively. Multiple graphs or subgraphs can be loaded at once, each placed in a separate tab.

5. A Mini-map, helps users stay oriented even in large data sets and provides context for the user. The mini-map provides a white box which represents the current view of the main visualization panel described above, and updates as the user pans and zooms.

6. In figure 2 the interaction network is shown using the default graph layout; however, there are several layouts available via this drop down menu.

7. Additional information about the states, actions, and cases are available here. Users are able to view the complete state of the node and a list of all case IDs who visited the state or edge. Other information, such as tutor hint messages or test scores can also be displayed here.

3.1 Guidelines Review

Our visualization was designed with the visual information seeking mantra in mind. As such the tool supports functionality for overview, zooming and filtering, details on demand, view relationships, and extraction. We are going to describe why the element is important to viewing interaction networks, how each element was included and supported, and improvements which can be made. Craft and Cairns state that many other developers cite the mantra as their guiding source for the development of their visualization but often forgo explaining how and where they used it [5]. We use this evaluation to find strengths and weaknesses in our approach, and confirm we have an implementation based on these principles.

3.1.1 Overview First

The hierarchical graph offers an overview representation of the students' behavior as they work through a single problem. Combined with the edge width representing frequency we provide a quick understanding of student behavior trends. In addition the mini-map consistently orients the user within the context they are working, always providing an overview for the user to reference as they navigate the Interaction Network graph. A possible improvement would be to apply other visual components to other variables, future studies will show the affects of these changes.

3.1.2 Zoom and Filter

Zooming and filtering on the interaction network means users can focus on specific behaviors they are interested in. Zooming is not only physically zooming in on the graph-layout, but also zooming in on students who performed a specific action, or visited a particular state. Zoom and filter are supported through a variety of filter and selection tools which allow manipulation based on states, student-IDs and actions. Additionally selection combined with creating subgraphs allows for alternative approaches for filtering and zooming, while the mouse wheel allows the user to zoom on the graph layout. Frequency based filters allow users to focus on either common trends or atypical behaviors exhibited by the students. Identifying common sequences in the interaction network could help identify sub-goals to problems and is one type of improvement that could be made to filtering and selection.

3.1.3 Detail on Demand

The details tab is where the user can find specific information regarding a state, action or student. Details are available in a set of tabs, displaying text information about the selected node. Including students who visited the node, the frequency of the state, actions leading from the node, whether the state is a goal or error state and the description of the state. These details are the finest granularity we can provide from the log data which was read in. One improvement for details is to provide aggregate user statistics regarding the current graph, for example number of error states, total number of states, number of actions, average actions per student, and more.

3.1.4 View Relationships

To relate in our tool is to compare behaviors between students and to compare sub-graphs of the interaction network. Students can be compared to other students by selecting their nodes via the student selection tool and generating sub-graphs. Sub-graphs allow users to compare, at once, all the times a specific action was used from all the students. Viewing the sub-graphs of two frequent paths let a user view how two different strategies were applied to solving the problem. View relationships is supported through the selection and filtering tools combined with creating new sub-graphs in separate tabs. An example of this type of comparison is shown in figure 3. Multiple problems can be loaded into separate tabs and problems can be compared, as well as Interaction Networks for groups of students. Much of the Interaction Network maintains a hierarchical structure, so it is possible to do slight comparison between similar approaches, but a desirable improvement for comparing could be to allow users to more easily compare strategies between students, more on this in the discussion section.

3.1.5 History

Shneiderman notes that History is the most often ignored element due to its high cost of development and is rare in prototypes [16]. In our prototype interface this feature is weakly supported. In *InVis*, when users edit the main graph in significant ways, such as filtering vertices, the new graph is placed in a new tab. Users are able to keep track of each in the tabbed interface. This allows some measure of preserving the history of the user actions. However, this could be improved further by allowing users to undo and redo actions such as selection even when they do not generate new subgraphs.

3.1.6 Extract

Sharing ones findings about student behavior is important, particularly for teachers, so challenging issues for students can be addressed. In *InVis* users can save the image of the visualization panel so that it can be shared. Teachers can show the sequence of steps to their students and highlight situations where errors were common.

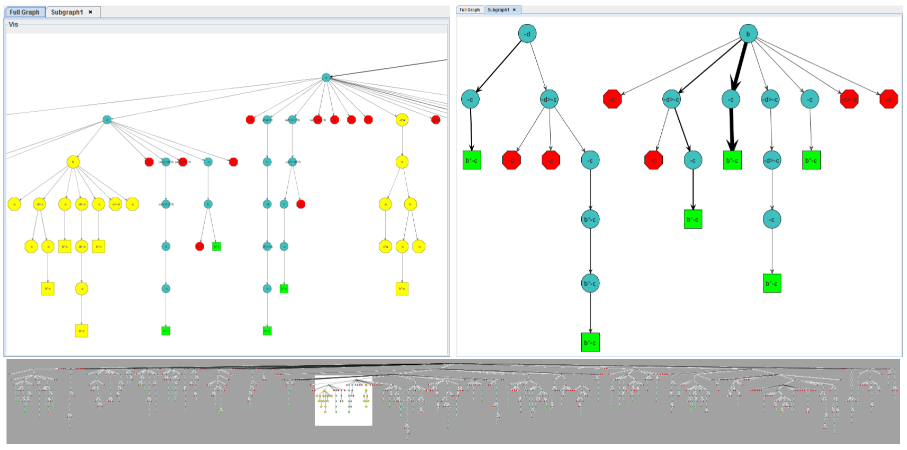


Figure 3: Left) The user has selected several nodes. By constructing a sub-graph, *InVis* presents the selected nodes in a new tab, shown on the Right. Below is an example of the mini-map, with the white box representing the view frame of the left hand image. Note the size of the graph when visualizing 170 students.

Improved sharing between colleagues could be supported by saving the current layouts, subgraphs, and graph annotations, so they can be stored and shared with others easily and they too can interact with the data via *InVis*.

4. CASE STUDIES AND SUCCESS STORIES

Plaisant comments on the usefulness of case studies and success stories in her work, 'The Challenge of Information Visualization Evaluation' [13]. These methods are common ways to aid in the evaluation of visualizations. These evaluations provide useful information from experts and can help address whether a user was able to find answers to questions they did not know they had. As mentioned by Plaisant, evaluation of tools meant to discover features which you did not know existed is difficult. Tory and Moller offer some support for ways in which this can be done [17]; and argue that expert reviews are useful because they can identify aspects of systems which non-experts may not recognize. In order for tools like *InVis*, to be adopted it is important to have success stories so that the early majority will adopt the tool [13], for this reason we provide four *InVis* success stories.

We met with the developers of two separate propositional logic tutors. Both developers are University professors of logic and have extensive knowledge about how their tutors and their logs. We modeled their logic tutor data and provided each with *InVis*. We observed as they used the visualization tool to explore their tutor log-data. We also met with the developers of an educational game and visualized student-player data in a similar way.

4.1 Case 1: Deep Thought

We visualized data from Deep Thought [6] and interviewed the professor responsible for its development. We met for one hour and had him explore tutor data and inform us of different insights and hypothesis he was able to discover or confirm from using *InVis*.

We prepared a data set of thirty students, representing a classroom of students. The professor noticed a student had performed addition rather than conjunction in order to derive $A \wedge B$, which is an incorrect application of the rule. After he recognized this, he mentioned that it was a common mistake made by students; this was his hypothesis. He then used the action selector and entered ADD which selected all instances of students performing the ADD action in his logic tutor. Next he built a sub-graph, moving all those actions and their corresponding nodes to a new tab. Last, he was able to confirm his hypothesis; the data showed that eight of nine applications of the addition rule (ADD) were errors, five of which would have been correct had the student performed conjunction instead, the action the student actually needed. With the same hypothesis in mind, a larger separate data set of Deep Thought tutor data was loaded, this time 170 students.

Again all addition actions were selected, a sub-graph generated, and the hypothesis confirmed. This time, 16 out of 18 applications of the addition rule were in fact errors, and 8 of the 16 would have been correct had the students used conjunction instead. Within minutes, our user was able to identify a hypothesis, check the data using *InVis* and confirm the hypothesis. In the second data set, the task certainly would have been time prohibitive had he scoured the 170 student logs of the data.

4.2 Case 2: Proof Solver

In this example, the professor was interested in the general behavioral trends of students. By including frequencies on nodes and edges, we are able to identify strategies that students perform in order to solve problems. In this case the hypothesis was that students who change all the implications into 'ORs' likely had no true strategy for completely solving the problem. The reason for this is over the years the professor has recognized students who employ this strategy often have difficulty actually solving the problem and thus students are explicitly instructed in class not to use this approach.

After loading that data into *InVis*, the professor looks for the two main strategies performed by the students. The strategies being the two most frequent sets of steps performed. The first strategy, having the highest frequency, is the strategy which she teaches to her students in class, we call the professor's strategy, the first node in this strategy has 74 students. The next most common first step has 29 students, and is the start of the prohibited strategy, that is to change an implication into an OR. Next the professor selected the first node of a strategy and performed the select sub-tree action, which selected all states derived after the current state, effectively selecting all the different variations of the professor's strategy. Then she created a sub-graph. The same was done for the prohibited strategy. Next she selected all of the goal nodes of each sub-graph in turn and looked at the combined frequency of the goal nodes for each sub-graph. For the 74 students who applied the professor's strategy, 55 of those students arrived at the goal, giving a 74% success rate. For the prohibited strategy approach, the sub-graph has a combined goal node frequency of 17 out of the 29 students, resulting in a 59% success rate which is noticeably lower. In total there are 174 students, and the two strategies highlighted above are the most common. The next two most common strategies have frequencies of 11 with 9 and 7 students successfully solving the problem with their respective strategies. Again suggesting the prohibited strategy approach has a particularly low success rate.

4.3 Case 3: Debugging Tutors

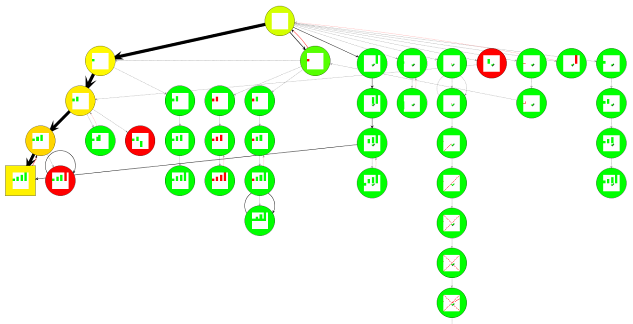
One interesting application of *InVis* is found in the debugging of tutors. In the previous examples, the Interaction Network uncovered bugs in the tutor systems; that is, places where the recorded interactions should not have been legal actions. This is interesting, as both of these tutors have been used for many years and their data have been subject to extensive analysis. However, these bugs were not discovered until their data were visualized with *InVis*. Viewing the entire group of user behaviors at once improves the ability to spot 'peculiar' behaviors. In *ProofSolver* several solutions were noticeably shorter than the average, or skipped to the goal in 'strange' and invalid ways. After examining the series of actions these students performed, the professor confirmed that the interactions were illegal and should not have been permitted.

In the case of *Deep Thought*, some students were able to reach the goal by repeatedly performing the same action. In this case, the students were able to use the instantiation-action inappropriately to add any proposition to the proof. As a result of this, students could simply add items directly to the proof rather than use the axioms, allowing them to game the system and illegally solve the problem.

4.4 Case 4: BeadLoom Game

We collected game log-files from a study performed on the BeadLoom Game (BLG) in 2010. Data came from a total of 6 classes, ranging from 6th to 8th grade; for a total of four sessions. There were 132 students, and 2,438 game-log files. The students were split into two groups (called A-day and B-day) and were presented with BLG features in different orders. The A-Day students were given access to custom puzzles (a free play option,) while B-Day students were given a competitive game element in the form of a leader board. Due to differences in student time lines some B-Day classes missed session three. These students followed an abbreviated A-Day schedule during session four. In order to investigate whether or not there were different problem solving patterns between the groups, we colored vertices based on the percentage of students who visited from each group. The values were normalized from green (A-Day) to red (B-Day.) We loaded the data into *InVis* and presented it to the BeadLoom Game developers.

Next we met with the BeadLoom Game developers and asked them to explore their log data using our prototype visualization tool. In figure 4 we have a set of students who worked on the same problem on two different days, the first and third day of the study. By looking at the number of states we can see a more diverse set of attempts on the first day. As mentioned before, edge width represents frequency, green vertices are from one set of students and red vertices are from another set based on how the study was run, the goal has a square vertex. At the start of our investigation we colored the


(a) Students on Day One

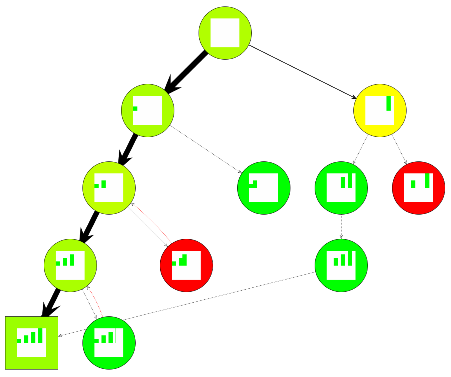

(b) Students on Day Three

Figure 4: This image shows the students attempt on the first day on the top, and their third day attempt on the bottom. This image suggests that as students become more familiar with the tool they are better at solving the problem and make fewer mistakes, thus fewer states are visited.

vertices to see if we could discover differences but it does not seem the classes had any significant differences between them. It is possible that the change in the number of states over time is the visual representation of learning, which figure 4 might suggest, additional research will be necessary to determine if that is so. The designers were able to identify a variety of design changes they would like to make to the game after spending roughly 20 minutes using *InVis*. The most surprising detail the developers were interested in was the number of students who seemed to participate in off-task behavior. Off-task behavior is easily identified as student solution-paths with low frequency and unusual length. For example, a student may opt to draw a picture rather than solve the puzzle. This will result in a path visually jutting out of the interaction network.

5. USABILITY STUDY

One goal of *InVis* is to make complex interaction data accessible to non-experts in the field of user-modeling. To test the usability of *InVis* we created a quantitative task-based test as a measure of summative usability testing. We developed 15 tasks based on the propositional logic tutor interaction data. These questions were designed based on common use-cases, and cover the range of features in the tool. Noting the inherent difficulties involved with evaluating a visualization tool, we ran a user study consisting of three sections, a quantitative portion, usability portion and qualitative portion, each supporting different types of evidence that the tool is effective at allowing users to make discoveries about their data.

5.1 Methods and Materials

For the design of the quantitative portion we created 15 questions which were atomic and had a set of correct responses, we determined how well users were able to answer the questions in this portion of the survey. Table 1 highlights 10 of the 15 questions as a representative sample of the nature of the tasks.

For the usability portion, we used a validated survey written by James Lewis at IBM [9]. Questions 9, 10 and 11 from his survey CSUQ were removed because they were deemed irrelevant to *InVis* because it does not contain any error messages. One difference between Lewis' survey and ours was in our CSUQ, the score has high scores being preferable rather than low scores; Strongly Agree is equal to seven instead of one. Our overall CSUQ score was 4.65. CSUQ stands for The Computer System Usability Questionnaire, and is divided into four scores, an Overall CSUQ score, a SYSUSE, INFOQUAL, and INTERQUAL scores. SYSUSE stands for system use, INFOQUAL is information quality and INTERQUAL is interface quality. The qualitative portion of the survey allowed users to provide open ended responses directed towards functionality that they would like to see in future versions of *InVis*.

We ran a user study with seven participants to determine the level of usability of our design of *InVis* for understanding interaction networks. In our study we used the logic domain as our target audience, so we collected data from how undergraduate students simplified problems in the logic domain using a computerized logic tutor. All teaching material was conducted in the classroom, and the logic tutor is strictly used for conducting homework. The students were given a set of premises, $[(A \rightarrow B), (C \rightarrow D), \neg(A \rightarrow D)]$, and were tasked with generating a first-order logic proof for the conclusion of $B \wedge \neg C$. Next we duplicated or removed students to ensure each task-question had a single correct answer.

Of the seven participants, we met with four individually and they used our computer with *InVis* and target dataset loaded. We gave them a brief overview of how the tool worked, how to zoom, pan and select. We also demonstrated how to generate a subgraph (a GUI button), and how to use the selection tools (text boxes). The demonstration lasted 5–10 minutes. Due to logistical issues, the other three participants were emailed a 2.5 page description of how the interactive elements of the tool works. This document served as the resource for the 5–10 minute demonstration. These three participants conducted the study by themselves via the Internet. Participants were instructed to contact us if any issues arose that they felt were unintended or prevented them from conducting the study. If any task took more than five minutes, they were asked to stop and move to the next one. Participants were instructed not to ask how to complete any of the tasks. After the quantitative section was complete they were asked to do the usability survey then the qualitative survey. Notably, a sample size of seven is low, but we recruited educators who have taught a course which uses the Deep Thought logic tutor, so we are limited by aspect. This decision is based on the assumption that a person who is unfamiliar with a domain, and related tutor, would not understand the logs of the tutor environment well enough to recognize meaningful tutor-based student behaviors.

5.2 Results

In the quantitative portion of the study the group of participants completed 85% of the tasks successfully ($M = 12.71, SD = 2.66$). From the questions in table 1 the most commonly missed questions were Q4, 64% success and Q6, 71% success. Participants spent

Table 1: Quantitative Questions/Tasks

Q1	Find the shortest correct solution-path to this problem.
Q3	Find the most frequent solution-path to this problem.
Q4	Find the error(s) with the highest frequency and write the State ID(s).
Q7	Write the action-label and the corresponding final state ID for the last action of student X?
Q9	After filtering nodes and edges to frequency 5 and greater, how many complete solution paths exist?
Q11	What is the student ID of the student(s) with longest solution to the goal?
Q12	Who are the students on the node with the following node label: –a*-d (note the label is: minus minus a * minus d)
Q14	Highlight the node with node-label: $\neg\neg A \wedge \neg D$ and select the sub-tree, then build a sub-graph. How many error nodes exist in this sub graph?
Q15	Answer yes or no, did student 81 find a goal solution?

an average of about 43 minutes on the quantitative survey, with a SD of about 20 minutes, from survey start and end time stamps. However, participants reported the tasks taking an average of 23 minutes, with a SD of 13 minutes.

To evaluate the relationship between the quantitative skills test and the usability survey we submitted the results to a bivariate correlation analysis. The quantitative skills test strongly correlated($M = 12.71, SD = 2.70$), $r = 0.87, n = 7, p = 0.01$, with the overall usability survey($M = 4.65, SD = 1.89$). With an *r-squared* of 0.76, which means that 76% of the variance in the quantitative skills test-scores is accounted for by variance in usability scores. We examined the three subsections of the usability survey. The quantitative skills test strongly correlated, $r = 0.78, n = 7, p = 0.04$, with the Sysuse score($M = 4.84, SD = 2.04$); it also strongly correlated, $r = 0.90, n = 7, p < 0.01$, with the Infoqual score($M = 4.54, SD = 1.98$); and it strongly correlated, $r = 0.96, n = 7, p < 0.01$, with the Interqual score($M = 4.39, SD = 1.70$).

This result provides evidence for the validity of using the quantitative test as a measure of the visualization quality. This provides insight into what functionalities the developers should focus on in future development of *InVis*. We cannot make strong casual inferences between the quantitative test and the usability scale, i.e., were participants able to complete the skills test because *InVis* was usable, or did they report that it was usable because they were able to complete the tasks? The fact that 85% of the tasks were correctly completed, with little time spent on training, provides evidence that our technique is usable by our target population. The questions that were most commonly missed, were tasks related to understanding the most frequent error, and the state from which the most *unique* errors were made. This highlights a potential problem with the current error-state representation; which seems to make differentiation between unique errors and frequent errors difficult to separate. It is difficult to interpret the results of the CSUQ survey, however this score is useful for comparing to future versions of *InVis*.

5.2.1 Qualitative

In the qualitative section of the study we asked participants about specific ways to improve *InVis*. We mention some of the most important suggestions we received from their experiences. These suggestions are from the conclusions we can make from their qualitative survey results and the comments made while using *InVis*.

An important issue to address is graph layouts, two issues regarding the layout were raised. First the layout would be more intuitive if it were possible to order the layout in some manner along the breadth (x-axis), either based on frequency, or other metric. By applying a more informed layout to the graph, we could order the states in some manner along the X-axis, for example making the most frequent path on the left, and the least frequent on the right.

The second problem is in regard to strategies, sub-strategies and ordered states. Participants mentioned they would like the layout to group or cluster approaches based on similar strategies. If two students each have nine identical steps, but the first step in each approach is different, then the layout does not necessarily put the states from these two approaches close to one another. When looking at 100 plus students, this makes understanding the number of strategies difficult to understand. A graph layout which places similar paths next to each other could provide a more intuitive visualization of the interaction network.

All participants reported a positive response for whether or not they would use the tool to augment their understanding of current classes' behavior and learning. This suggests a need for these types of tools for exploring, and understanding student behaviors in software tutors. We will conclude with a quote from one user who said, 'The tool provides a sense of how broadly varying students are in their approaches, how many get stuck, and how many make similar mistakes.' Which we feel is a good representation of the kinds of insights *InVis* was designed for detecting.

6. CONCLUSION

The main contribution of this work is the discovery and implementation of visualization techniques for user-interaction data from educational systems. This led to new insights into problem-solving in the deep thought logic tutoring environment, for example the conclusions drawn in the case studies. The use of interactive visualization techniques combined with a interaction network model in *InVis* allows users to explore and gain insight from interaction-log data. We performed a user study on *InVis* to show that users can successfully complete relevant tasks, and paired these results with a standardized method for testing the usability of a software tool. Users were able to explore an entire class' set of interactions and were able to confirm some of the hypothesis they had about students, which was a primary goal. This suggests that our technique is effective at allowing users to explore and learn information from the data. This is the first step in creating a domain independent visualization tool for understanding student behavior in software tutors, and our initial results seem promising for the future development of *InVis*.

7. REFERENCES

[1] T. Barnes and J. Stamper. Toward the extraction of production rules for solving logic proofs. In *Proceedings of the 13th International Conference on Artificial Intelligence in Education, Educational Data Mining Workshop (AIED2007)*, pages 11–20, 2007.

[2] M. Bastian, S. Heymann, and M. Jacomy. Gephi: An open source software for exploring and manipulating networks, 2009.

[3] D. Ben-Naim, M. Bain, and N. Marcus. A User-Driven and Data-Driven Approach for Supporting Teachers in Reflection and Adaptation of Adaptive Tutorials, 2009.

[4] A. Boyce and T. Barnes. Beadloom game: using game elements to increase motivation and learning. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*, FDG '10, pages 25–31, New York, NY, USA, 2010. ACM.

[5] B. Craft and P. Cairns. Beyond guidelines: What can we learn from the visual information seeking mantra? In *Proceedings of the Ninth International Conference on Information Visualisation*, pages 110–118, Washington, DC, USA, 2005. IEEE Computer Society.

[6] M. J. Croy. Graphic interface design and deductive proof construction. *J. Comput. Math. Sci. Teach.*, 18:371–385, December 1999.

[7] J. Ellson, E. R. Gansner, E. Koutsofios, S. C. North, and G. Woodhull. Graphviz - Open Source Graph Drawing Tools. *Graph Drawing*, pages 483–484, 2001.

[8] K. Koedinger, R. Baker, K. Cunningham, A. Skogsholm, B. Leber, and J. Stamper. *A Data Repository for the EDM community: The PSLC DataShop*. Boca Raton, FL: CRC Press, 2010.

[9] J. R. Lewis. Ibm computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. *Int. J. Hum.-Comput. Interact.*, 7:57–78, January 1995.

[10] R. Mazza and V. Dimitrova. Visualising student tracking data to support instructors in web-based distance education. In *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, WWW Alt. '04, pages 154–161, New York, NY, USA, 2004. ACM.

[11] A. Merceron and K. Yacef. Tada-ed for educational data mining. *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning*, 7(1):267–287, 2005.

[12] N. A. of Engineering. Grand Challenges for Engineering, 2008.

[13] C. Plaisant. The challenge of information visualization evaluation. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '04, pages 109–116, New York, NY, USA, 2004. ACM.

[14] C. Plaisant, J. Grosjean, and B. B. Bederson. Spacetree: Supporting exploration in large node link tree, design evolution and empirical evaluation. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis'02)*, INFOVIS '02, pages 57–, Washington, DC, USA, 2002. IEEE Computer Society.

[15] C. Romero and S. Ventura. Educational data mining: A survey from 1995 to 2005. *Expert Syst. Appl.*, 33:135–146, July 2007.

[16] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. *Visual Languages, IEEE Symposium on*, 0:336, 1996.

[17] M. Tory, I. Lab, and T. Moller. Evaluating visualizations: Do expert reviews work. *IEEE Computer Graphics and Applications*, 25:8–11, 2005.

[18] S. White, D. Fisher, P. Smyth, S. White, and Y. B. Boey. Analysis and visualization of network data using jung. *Journal Of Statistical Software*, VV(Ii):1–35, 2005.

[19] D. Zapata-Rivera and J. E. Greer. Exploring various guidance mechanisms to support interaction with inspectable learner models. In *Proceedings of the 6th International Conference on Intelligent Tutoring Systems*, ITS '02, pages 442–452, London, UK, UK, 2002. Springer-Verlag.

Tag-Aware Ordinal Sparse Factor Analysis for Learning and Content Analytics

Andrew S. Lan, Christoph Studer, Andrew E. Waters, Richard G. Baraniuk
Rice University, TX, USA
{mr.lan, studer, waters, richb}@sparfa.com

ABSTRACT
Machine learning offers novel ways and means to design *personalized learning systems* wherein each student's educational experience is customized in real time depending on their background, learning goals, and performance to date. SPARse Factor Analysis (SPARFA) is a novel framework for machine learning-based *learning analytics*, which estimates a learner's knowledge of the concepts underlying a domain, and *content analytics*, which estimates the relationships among a collection of questions and those concepts. SPARFA jointly learns the associations among the questions and the concepts, learner concept knowledge profiles, and the underlying question difficulties, solely based on the correct/incorrect graded responses of a population of learners to a collection of questions. In this paper, we extend the SPARFA framework significantly to enable: (i) the analysis of graded responses on an ordinal scale (partial credit) rather than a binary scale (correct/incorrect); (ii) the exploitation of tags/labels for questions that partially describe the question–concept associations. The resulting *Ordinal SPARFA-Tag* framework greatly enhances the interpretability of the estimated concepts. We demonstrate using real educational data that Ordinal SPARFA-Tag outperforms both SPARFA and existing collaborative filtering techniques in predicting missing learner responses.

Keywords
Factor analysis, ordinal regression, matrix factorization, personalized learning, block coordinate descent

1. INTRODUCTION
Today's education system typically provides only a "one-size-fits-all" learning experience that does not cater to the background, interests, and goals of individual learners. Modern machine learning (ML) techniques provide a golden opportunity to reinvent the way we teach and learn by making it more personalized and, hence, more efficient and effective. The last decades have seen a great acceleration in the development of *personalized learning systems* (PLSs), which can be grouped into two broad categories: (i) high-quality, but labor-intensive rule-based systems designed by domain experts that are hard-coded to give feedback in pre-defined scenarios [8], and (ii) more affordable and scalable ML-based systems that mine various forms of learner data in order to make performance predictions for each learner [15, 18, 30].

1.1 Learning and content analytics
Learning analytics (LA, estimating what a learner understands based on data obtained from tracking their interactions with learning content) and *content analytics* (CA, organizing learning content such as questions, instructional text, and feedback hints) enable a PLS to generate automatic, targeted feedback to learners, their instructors, and content authors [23]. Recently we proposed a new framework for LA and CA based on SPARse Factor Analysis (SPARFA) [24]. SPARFA consists of a statistical model and convex-optimization-based inference algorithms for analytics that leverage the fact that the knowledge in a given subject can typically be decomposed into a small set of latent knowledge components that we term *concepts* [24]. Leveraging the latent concepts and based only on the graded binary-valued responses (i.e., correct/incorrect) to a set of questions, SPARFA jointly estimates (i) the associations among the questions and the concepts (via a "concept graph"), (ii) learner concept knowledge profiles, and (iii) the underlying question difficulties.

1.2 Contributions
In this paper, we develop *Ordinal SPARFA-Tag*, a significant extension to the SPARFA framework that enables the exploitation of the additional information that is often available in educational settings. First, Ordinal SPARFA-Tag exploits the fact that responses are often graded on an ordinal scale (partial credit), rather than on a binary scale (correct/incorrect). Second, Ordinal SPARFA-Tag exploits tags/labels (i.e., keywords characterizing the underlying knowledge component related to a question) that can be attached by instructors and other users to questions. Exploiting pre-specified tags within the estimation procedure provides significantly more interpretable question–concept associations. Furthermore, our statistical framework can discover new concept–question relationships that would not be in the pre-specified tag information but, nonetheless, explain the graded learner–response data.

We showcase the superiority of Ordinal SPARFA-Tag compared to the methods in [24] via a set of synthetic "ground truth" simulations and on a variety of experiments with real-world educational datasets. We also demonstrate that Ordinal SPARFA-Tag outperforms existing state-of-the-art collaborative filtering techniques in terms of predicting missing ordinal learner responses.

2. STATISTICAL MODEL
We assume that the learners' knowledge level on a set of abstract latent *concepts* govern the responses they provide to a set of questions. The SPARFA statistical model charac-

terizes the probability of learners' binary (correct/incorrect) graded responses to questions in terms of three factors: (i) question–concept associations, (ii) learners' concept knowledge, and (iii) intrinsic question difficulties; details can be found in [24, Sec. 2]. In this section, we will first extend the SPARFA framework to characterize *ordinal* (rather than binary-valued) responses, and then impose additional structure in order to model real-world educational behavior more accurately.

2.1 Model for ordinal learner response data

Suppose that we have N learners, Q questions, and K underlying concepts. Let $Y_{i,j}$ represent the graded response (i.e., score) of the j^{th} learner to the i^{th} question, which are from a set of P ordered labels, i.e., $Y_{i,j} \in \mathcal{O}$, where $\mathcal{O} = \{1, \ldots, P\}$. For the i^{th} question, with $i \in \{1, \ldots, Q\}$, we propose the following model for the learner–response relationships:

$$\begin{aligned} Z_{i,j} &= \mathbf{w}_i^T \mathbf{c}_j + \mu_i, \ \forall (i,j), \\ Y_{i,j} &= \mathcal{Q}(Z_{i,j} + \epsilon_{i,j}), \ \epsilon_{i,j} \sim \mathcal{N}(0, 1/\tau_{i,j}), \ (i,j) \in \Omega_{\text{obs}}, \end{aligned} \tag{1}$$

where the column vector $\mathbf{w}_i \in \mathbb{R}^K$ models the *concept associations*; i.e., it encodes how question i is related to each concept. Let the column vector $\mathbf{c}_j \in \mathbb{R}^K$, $j \in \{1, \ldots, N\}$, represent the latent *concept knowledge* of the j^{th} learner, with its k^{th} component representing the j^{th} learner's knowledge of the k^{th} concept. The scalar μ_i models the *intrinsic difficulty* of question i, with large positive value of μ for an easy question. The quantity $\epsilon_{i,j}$ models the uncertainty of learner j answering question i correctly/incorrectly and $\mathcal{N}(0, 1/\tau_{i,j})$ denotes a zero-mean Gaussian distribution with precision parameter $\tau_{i,j}$, which models the *reliability* of the observation of learner j answering question i. We will further assume $\tau_{i,j} = \tau$, meaning that all the observations have the same reliability.[1] The slack variable $Z_{i,j}$ in (1) governs the probability of the observed grade $Y_{i,j}$. The set $\Omega_{\text{obs}} \subseteq \{1, \ldots, Q\} \times \{1, \ldots, N\}$ contains the indices associated to the observed learner–response data, in case the response data is not fully observed.

In (1), $\mathcal{Q}(\cdot) : \mathbb{R} \to \mathcal{O}$ is a scalar quantizer that maps a real number into P ordered labels according to

$$\mathcal{Q}(x) = p \quad \text{if } \omega_{p-1} < x \leq \omega_p, \ p \in \mathcal{O},$$

where $\{\omega_0, \ldots, \omega_P\}$ is the set of quantization bin boundaries satisfying $\omega_0 < \omega_1 < \cdots < \omega_{P-1} < \omega_P$, with ω_0 and ω_P denoting the lower and upper bound of the domain of the quantizer $\mathcal{Q}(\cdot)$.[2] This quantization model leads to the equivalent input–output relation

$$\begin{aligned} Z_{i,j} &= \mathbf{w}_i^T \mathbf{c}_j + \mu_i, \quad \forall (i,j), \quad \text{and} \\ p(Y_{i,j} = p \mid Z_{i,j}) &= \int_{\omega_{p-1}}^{\omega_p} \mathcal{N}(s | Z_{i,j}, 1/\tau_{i,j})\, \mathrm{d}s \\ &= \Phi(\tau(\omega_p - Z_{i,j})) - \Phi(\tau(\omega_{p-1} - Z_{i,j})), (i,j) \in \Omega_{\text{obs}}, \end{aligned} \tag{2}$$

where $\Phi(x) = \int_{-\infty}^{x} \mathcal{N}(s|0,1) \mathrm{d}s$ denotes the *inverse probit function*, with $\mathcal{N}(s|0,1)$ representing the value of a standard normal distribution evaluated at s.[3]

[1]Accounting for learner/question-varying reliabilities is straightforward and omitted for the sake of brevity.

[2]In most situations, we have $\omega_0 = -\infty$ and $\omega_P = \infty$.

[3]Space limitations preclude us from discussing a corresponding logistic-based model; the extension is straightforward.

We can conveniently rewrite (1) and (2) in matrix form as

$$\begin{aligned} &\mathbf{Z} = \mathbf{W}\mathbf{C}, \quad \forall (i,j), \quad \text{and} \\ &p(Y_{i,j} \mid Z_{i,j}) = \Phi(\tau(U_{i,j} - Z_{i,j})) \\ &\qquad\qquad - \Phi(\tau(L_{i,j} - Z_{i,j})), \ (i,j) \in \Omega_{\text{obs}}, \end{aligned} \tag{3}$$

where $\mathbf{Y}$ and $\mathbf{Z}$ are $Q \times N$ matrices. The $Q \times (K+1)$ matrix $\mathbf{W}$ is formed by concatenating $[\mathbf{w}_1, \ldots, \mathbf{w}_Q]^T$ with the intrinsic difficulty vector $\boldsymbol{\mu}$ and $\mathbf{C}$ is a $(K+1) \times N$ matrix formed by concatenating the $K \times N$ matrix $[\mathbf{c}_1, \ldots, \mathbf{c}_N]$ with an all-ones row vector $\mathbf{1}_{1 \times N}$. We furthermore define the $Q \times N$ matrices $\mathbf{U}$ and $\mathbf{L}$ to contain the upper and lower bin boundaries corresponding to the observations in $\mathbf{Y}$, i.e., we have $U_{i,j} = \omega_{Y_{i,j}}$ and $L_{i,j} = \omega_{Y_{i,j}-1}$, $\forall (i,j) \in \Omega_{\text{obs}}$.

We emphasize that the statistical model proposed above is significantly more general than the original SPARFA model proposed in [24], which is a special case of (1) with $P = 2$ and $\tau = 1$. The precision parameter τ does not play a central role in [24] (it has been set to $\tau = 1$), since the observations are binary-valued with bin boundaries $\{-\infty, 0, \infty\}$. For ordinal responses (with $P > 2$), however, the precision parameter τ significantly affects the behavior of the statistical model and, hence, we estimate the precision parameter τ directly from the observed data.

2.2 Fundamental assumptions

Estimating $\mathbf{W}$, $\boldsymbol{\mu}$ and $\mathbf{C}$ from $\mathbf{Y}$ is an ill-posed problem, in general, since there are more unknowns than observations and the observations are ordinal (and not real-valued). To ameliorate the illposedness, [24] proposed three assumptions accounting for real-world educational situations:

(A1) *Low-dimensionality*: Redundancy exists among the questions in an assessment, and the observed graded learner responses live in a low-dimensional space, i.e., $K \ll N, Q$.

(A2) *Sparsity*: Each question measures the learners' knowledge on only a few concepts (relative to N and Q), i.e., the question–concept association matrix $\mathbf{W}$ is sparse.

(A3) *Non-negativity*: The learners' knowledge on concepts does not reduce the chance of receiving good score on any question, i.e., the entries in $\mathbf{W}$ are non-negative. Therefore, large positive values of the entries in $\mathbf{C}$ represent good concept knowledge, and vice versa.

Although these assumptions are reasonable for a wide range of educational contexts (see [24] for a detailed discussion), they are hardly complete. In particular, additional information is often available regarding the questions and the learners in some situations. Hence, we impose one additional assumption:

(A4) *Oracle support*: Instructor-provided tags on questions provide prior information on some question–concept associations. In particular, associating each tag with a *single concept* will partially (or fully) determine the locations of the non-zero entries in $\mathbf{W}$.

As we will see, assumption (A4) significantly improves the limited interpretability of the estimated factors $\mathbf{W}$ and $\mathbf{C}$ over the conventional SPARFA framework [24], which relies on a (somewhat ad-hoc) post-processing step to associate instructor provided tags with concepts. In contrast, we utilize the tags as "oracle" support information on $\mathbf{W}$ *within* the model, which enhances the explanatory performance of the statistical framework, i.e., it enables to associate each concept directly with a predefined tag. Note that user-specified tags might not be precise or complete. Hence, the proposed estimation algorithm must be capable of discovering new question–concept associations and removing predefined associations that cannot be explained from the observed data.

3. ALGORITHM

We start by developing *Ordinal SPARFA-M*, a generalization of SPARFA-M from [24] to ordinal response data. Then, we detail *Ordinal SPARFA-Tag*, which considers prespecified question tags as oracle support information of $\mathbf{W}$, to estimate $\mathbf{W}$, $\mathbf{C}$, and τ, from the ordinal response matrix $\mathbf{Y}$ while enforcing the assumptions (A1)–(A4).

3.1 Ordinal SPARFA-M

To estimate $\mathbf{W}$, $\mathbf{C}$, and τ in (3) given $\mathbf{Y}$, we maximize the log-likelihood of $\mathbf{Y}$ subject to (A1)–(A4) by solving

$$\text{(P)}\quad \underset{\mathbf{W},\mathbf{C},\tau}{\text{minimize}} -\textstyle\sum_{i,j:(i,j)\in\Omega_{\text{obs}}} \log p(Y_{i,j}|\tau\mathbf{w}_i^T\mathbf{c}_j) + \lambda \sum_i \|\mathbf{w}_i\|_1 \;\text{ subject to }\; \mathbf{W}\geq 0, \tau>0, \|\mathbf{C}\|\leq\eta.$$

Here, the likelihood of each response $p(Y_{i,j}|\tau\mathbf{w}_i^T\mathbf{c}_j)$ is given by (2). The regularization term $\lambda\sum_i\|\mathbf{w}_i\|_1$ imposes sparsity on each vector $\mathbf{w}_i$ to account for (A2). To prevent arbitrary scaling between $\mathbf{W}$ and $\mathbf{C}$, we gauge the norm of the matrix $\mathbf{C}$ by applying a matrix norm constraint $\|\mathbf{C}\|\leq\eta$. For example, the Frobenius norm constraint $\|\mathbf{C}\|_F\leq\eta$ can be used. Alternatively, the nuclear norm constraint $\|\mathbf{C}\|_*\leq\eta$ can also be used, promoting low-rankness of $\mathbf{C}$ [9], motivated by the facts that (i) reducing the number of degrees-of-freedom in $\mathbf{C}$ helps to prevent overfitting to the observed data and (ii) learners can often be clustered into a few groups due to their different demographic backgrounds and learning preferences.

The log-likelihood of the observations in (P) is concave in the product $\tau\mathbf{w}_i^T\mathbf{c}_j$ [36]. Consequently, the problem (P) is *tri-convex*, in the sense that the problem obtained by holding two of the three factors $\mathbf{W}, \mathbf{C}$, and τ constant and optimizing the third one is convex. Therefore, to arrive at a practicable way of solving (P), we propose the following computationally efficient block coordinate descent approach, with $\mathbf{W}$, $\mathbf{C}$, and τ as the different blocks of variables.

The matrices $\mathbf{W}$ and $\mathbf{C}$ are initialized as i.i.d. standard normal random variables, and we set $\tau = 1$. We then iteratively optimize the objective of (P) for all three factors in round-robin fashion. Each (outer) iteration consists of three phases: first, we hold $\mathbf{W}$ and τ constant and optimize $\mathbf{C}$; second, we hold $\mathbf{C}$ and τ constant and separately optimize each row vector $\mathbf{w}_i$; third, we hold $\mathbf{W}$ and $\mathbf{C}$ fixed and optimize over the precision parameter τ. These three phases form the outer loop of Ordinal SPARFA-M.

The sub-problems for estimating $\mathbf{W}$ and $\mathbf{C}$ correspond to the following *ordinal regression* (OR) problems [12]:

$$\text{(OR-W)}\quad \underset{\mathbf{w}_i:\,W_{i,k}\geq 0\ \forall k}{\text{minimize}} -\textstyle\sum_j \log p(Y_{i,j}|\tau\mathbf{w}_i^T\mathbf{c}_j)+\lambda\|\mathbf{w}_i\|_1,$$

$$\text{(OR-C)}\quad \underset{\mathbf{C}:\|\mathbf{C}\|\leq\eta}{\text{minimize}} -\textstyle\sum_{i,j}\log p(Y_{i,j}|\tau\mathbf{w}_i^T\mathbf{c}_j).$$

To solve (OR-W) and (OR-C), we deploy the iterative first-order methods detailed below. To optimize the precision parameter τ, we compute the solution to

$$\underset{\tau>0}{\text{minimize}} \; -\textstyle\sum_{i,j:(i,j)\in\Omega_{\text{obs}}} \log\left(\Phi\left(\tau\left(\mathbf{U}_{i,j}-\mathbf{w}_i^T\mathbf{c}_j\right)\right)-\Phi\left(\tau\left(\mathbf{L}_{i,j}-\mathbf{w}_i^T\mathbf{c}_j\right)\right)\right),$$

via the secant method [26].

Instead of fixing the quantization bin boundaries $\{\omega_0,\ldots,\omega_P\}$ introduced in Sec. 2 and optimizing the precision and intrinsic difficulty parameters, one can fix $\tau = 1$ and optimize the bin boundaries instead, an approach used in, e.g., [21]. We emphasize that optimization of the bin boundaries can also be performed straightforwardly via the secant method, iteratively optimizing each bin boundary while keeping the others fixed. We omit the details for the sake of brevity. Note that we have also implemented variants of Ordinal SPARFA-M that directly optimize the bin boundaries, while keeping τ constant; the associated prediction performance is shown in Sec. 4.3.

3.2 First-order methods for regularized ordinal regression

As in [24], we solve (OR-W) using the FISTA framework [4]. (OR-C) also falls into the FISTA framework, by re-writing the convex constraint $\|\mathbf{C}\|\leq\eta$ as a penalty term $\delta(\mathbf{C}:\|\mathbf{C}\|>\eta)$ and treat it as a non-smooth regularizer, where $\delta(\mathbf{C}:\|\mathbf{C}\|>\eta)$ is the delta function, equaling 0 if $\|\mathbf{C}\|\leq\eta$ and ∞ otherwise. Each iteration of both algorithms consists of two steps: A gradient-descent step and a shrinkage/projection step. Take (OR-W), for example, and let $f(\mathbf{w}_i) = -\sum_j \log p(Y_{i,j}|\tau\mathbf{w}_i^T\mathbf{c}_j)$. Then, the gradient step is given by[4]

$$\nabla f = \nabla_{\mathbf{w}_i}(-\textstyle\sum_j \log p(Y_{i,j}|\tau\mathbf{w}_i^T\mathbf{c}_j)) = -\mathbf{C}\mathbf{p}. \tag{4}$$

Here, $\mathbf{p}$ is a $N\times 1$ vector, with the j^{th} element equal to

$$\frac{\mathcal{N}(\tau(U_{i,j}-\mathbf{w}_i^T\mathbf{c}_j))-\mathcal{N}(\tau(L_{i,j}-\mathbf{w}_i^T\mathbf{c}_j))}{\Phi(\tau(U_{i,j}-\mathbf{w}_i^T\mathbf{c}_j)-\Phi(\tau(L_{i,j}-\mathbf{w}_i^T\mathbf{c}_j))},$$

where $\Phi(\cdot)$ is the inverse probit function. The gradient step and the shrinkage step for $\mathbf{w}_i$ corresponds to

$$\hat{\mathbf{w}}_i^{\ell+1}\leftarrow\mathbf{w}_i^\ell - t_\ell\nabla f, \tag{5}$$

and

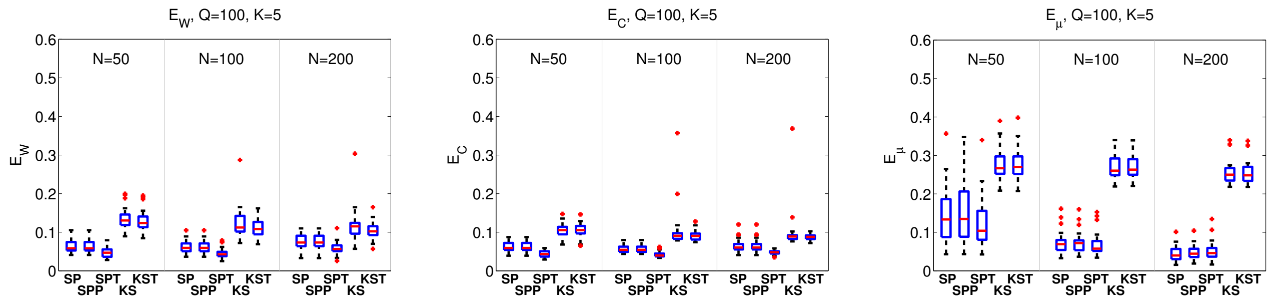
$$\mathbf{w}_i^{\ell+1}\leftarrow\max\{\hat{\mathbf{w}}_i^{\ell+1}-\lambda t_\ell, 0\}, \tag{6}$$

respectively, where t_ℓ is a suitable step-size. For (OR-C), the gradient with respect to each column $\mathbf{c}_j$ is given by substituting $\mathbf{W}^T$ for $\mathbf{C}$ and $\mathbf{c}_j$ for $\mathbf{w}_i$ in (4). Then, the gradient for $\mathbf{C}$ is formed by aggregating all these individual gradient vectors for $\mathbf{c}_j$ into a corresponding gradient matrix.

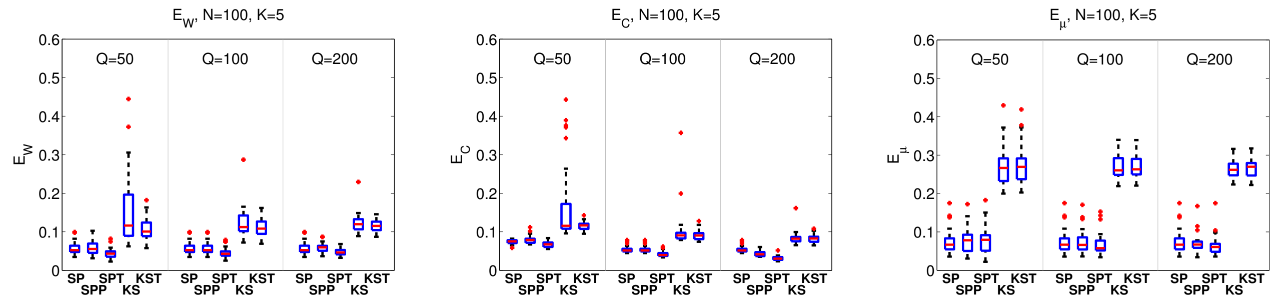
For the Frobenius norm constraint $\|\mathbf{C}\|_F\leq\eta$, the projection step is given by [7]

$$\mathbf{C}^{\ell+1}\leftarrow\begin{cases}\hat{\mathbf{C}}^{\ell+1} & \text{if } \|\hat{\mathbf{C}}^{\ell+1}\|_F\leq\eta\\ \eta\frac{\hat{\mathbf{C}}^{\ell+1}}{\|\hat{\mathbf{C}}^{\ell+1}\|_F} & \text{otherwise.}\end{cases} \tag{7}$$

[4]Here, we assume $\Omega_{\text{obs}} = \{1,\ldots,Q\}\times\{1,\ldots,N\}$ for simplicity; a generalization to the case of missing entries in $\mathbf{Y}$ is straightforward.



(a) Impact of the number of learners, $N \in \{50, 100, 200\}$, with the number of questions Q fixed.



(b) Impact of the number of questions, $Q \in \{50, 100, 200\}$, with the number of learners N fixed.

Figure 1: Performance comparison of Ordinal SPARFA-M vs. K-SVD$_+$. "SP" denotes Ordinal SPARFA-M without given support Γ of W, "SPP" denotes the variant with estimated precision τ, and "SPT" denotes Ordinal SPARFA-Tag. "KS" stands for K-SVD$_+$, and "KST" denotes its variant with given support Γ.

For the nuclear-norm constraint $\|\mathbf{C}\|_* \leq \eta$, the projection step is given by

$$\mathbf{C}^{\ell+1} \leftarrow \mathbf{U}\text{diag}(\mathbf{s})\mathbf{V}^T, \text{ with } \mathbf{s} = \text{Proj}_\eta(\text{diag}(\mathbf{S})), \quad (8)$$

where $\hat{\mathbf{C}}^{\ell+1} = \mathbf{USV}^T$ denotes the singular value decomposition, and $\text{Proj}_\eta(\cdot)$ is the projection onto the ℓ_1-ball with radius η (see, e.g., [16] for the details).

The update steps (5), (6), and (7) (or (8)) require a suitable step-size t_ℓ to ensure convergence. We consider a constant step-size and set t_ℓ to the reciprocal of the Lipschitz constant [4]. The Lipschitz constants correspond to $\tau^2\sigma^2_{\max}(\mathbf{C})$ for (OR-W) and $\tau^2\sigma^2_{\max}(\mathbf{W})$ for (OR-C), with $\sigma_{\max}(\mathbf{X})$ representing the maximum singular value of $\mathbf{X}$.

3.3 Ordinal SPARFA-Tag

We now develop the Ordinal SPARFA-Tag algorithm that incorporates (A4). Assume that the total number of tags associated with the Q questions equal K (each of the K concepts correspond to a tag), and define the set $\Gamma = \{(i,k) :$ question i has tag $k\}$ as the set of indices of entries in $\mathbf{W}$ identified by pre-defined tags, and $\bar{\Gamma}$ as the set of indices not in Γ, we can re-write the optimization problem (P) as:

$$(\text{P}_\Gamma) \quad \underset{\mathbf{W},\mathbf{C},\tau}{\text{minimize}} \quad -\textstyle\sum_{i,j:(i,j)\in\Omega_{\text{obs}}} \log p(Y_{i,j}|\tau\mathbf{w}_i^T\mathbf{c}_j) + \lambda \textstyle\sum_i \|\mathbf{w}_i^{(\bar{\Gamma})}\|_1 + \gamma \textstyle\sum_i \frac{1}{2}\|\mathbf{w}_i^{(\Gamma)}\|_2^2$$
$$\text{subject to } \; \mathbf{W} \geq 0, \tau > 0, \|\mathbf{C}\| \leq \eta.$$

Here, $\mathbf{w}_i^{(\Gamma)}$ is a vector of those entries in $\mathbf{w}_i$ belonging to the set Γ, while $\mathbf{w}_i^{(\bar{\Gamma})}$ is a vector of entries in $\mathbf{w}_i$ not belonging to Γ. The ℓ_2-penalty term on $\mathbf{w}_i^{(\Gamma)}$ regularizes the entries in $\mathbf{W}$ that are part of the (predefined) support of $\mathbf{W}$; we set $\gamma = 10^{-6}$ in all our experiments. The ℓ_1-penalty term on $\mathbf{w}_i^{(\bar{\Gamma})}$ induces sparsity on the entries in $\mathbf{W}$ that are not predefined but might be in the support of $\mathbf{W}$. Reducing the parameter λ enables one to discover new question–concept relationships (corresponding to new non-zero entries in $\mathbf{W}$) that were not contained in Γ.

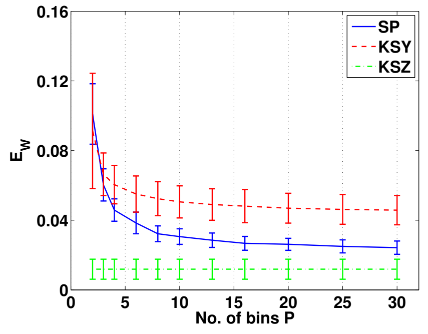
The problem (P_Γ) is solved analogously to the approach described in Sec. 3.2, except that we split the $\mathbf{W}$ update step into two parts that operate separately on the entries indexed by Γ and $\bar{\Gamma}$. For the entries in Γ, the projection step corresponds to

$$\mathbf{w}_i^{(\Gamma),\ell+1} \leftarrow \max\{\hat{\mathbf{w}}_i^{(\Gamma),\ell+1}/(1+\gamma t_\ell), 0\}.$$

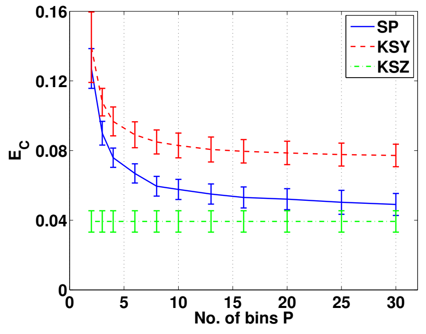
The step for the entries indexed by $\bar{\Gamma}$ is given by (6). Since Ordinal SPARFA-Tag is tri-convex, it does not necessarily converge to a global optimum. Nevertheless, we can leverage recent results in [24, 35] in order to show that Ordinal SPARFA-Tag converges to a local optimum from an arbitrary starting point. Furthermore, if the starting point is within a close neighborhood of a global optimum of (P), then Ordinal SPARFA-Tag converges to this global optimum.

4. EXPERIMENTS

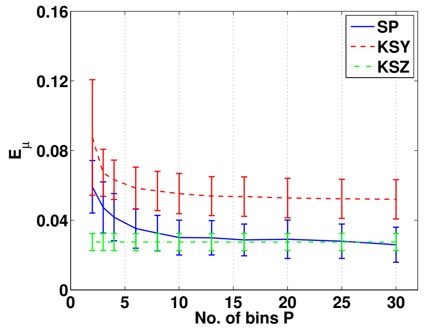
We first showcase the performance of Ordinal SPARFA-Tag on synthetic data to demonstrate its convergence to a known ground truth. We then demonstrate the ease of interpretation of the estimated factors by leveraging instructor provided tags in combination with a Frobenius or nuclear norm constraint for two real educational datasets. We finally compare the performance of Ordinal SPARFA-M to state-of-the-art collaborative filtering techniques on predicting unobserved ordinal learner responses.



(a) $E_{\mathbf{W}}$ versus P



(b) $E_{\mathbf{C}}$ versus P



(c) $E_{\boldsymbol{\mu}}$ versus P

Figure 2: Performance comparison of Ordinal SPARFA-M vs. K-SVD$_+$ by varying the number of quantization bins. "SP" denotes Ordinal SPARFA-M, "KSY" denotes K-SVD$_+$ operating on Y, and "KSZ" denotes K-SVD$_+$ operating on Z in (3) (the unquantized data).

4.1 Synthetic data

In order to show that Ordinal SPARFA-Tag is capable of estimating latent factors based on binary observations, we compare the performance of Ordinal SPARFA-Tag to a non-negative variant of the popular K-SVD dictionary learning algorithm [1], referred to as K-SVD$_+$, which we have detailed in [24]. We consider both the case when the precision τ is known a-priori and also when it must be estimated. In all synthetic experiments, the algorithm parameters λ and η are selected according to Bayesian information criterion (BIC) [17]. All experiments are repeated for 25 Monte-Carlo trials.

In all synthetic experiments, we retrieve estimates of all factors, $\widehat{\mathbf{W}}$, $\widehat{\mathbf{C}}$, and $\hat{\boldsymbol{\mu}}$. For Ordinal SPARFA-M and K-SVD$_+$, the estimates $\widehat{\mathbf{W}}$ and $\widehat{\mathbf{C}}$ are re-scaled and permuted as in [24]. We consider the following error metrics:

$$E_{\mathbf{W}} = \frac{\|\mathbf{W}-\widehat{\mathbf{W}}\|_F^2}{\|\mathbf{W}\|_F^2}, \quad E_{\mathbf{C}} = \frac{\|\mathbf{C}-\widehat{\mathbf{C}}\|_F^2}{\|\mathbf{C}\|_F^2}, \quad E_{\boldsymbol{\mu}} = \frac{\|\boldsymbol{\mu}-\hat{\boldsymbol{\mu}}\|_2^2}{\|\boldsymbol{\mu}\|_2^2}.$$

We generate the synthetic test data $\mathbf{W}$, $\mathbf{C}$, $\boldsymbol{\mu}$ as in [24, Eq. 10] with $K = 5$, $\mu_0 = 0$, $v_\mu = 1$, $\lambda_k = 0.66\ \forall k$, and $\mathbf{V}_0 = \mathbf{I}_K$. $\mathbf{Y}$ is generated according to (3), with $P = 5$ bins and $\{\omega_0, \ldots, \omega_5\} = \{-\infty, -2.1, -0.64, 0.64, 2.1, \infty\}$, such that the entries of $\mathbf{Z}$ fall evenly into each bin. The number of concepts K for each question is chosen uniformly in $\{1, 2, 3\}$. We first consider the impact of problem size on estimation error in Fig. 2. To this end, we fix $Q = 100$ and sweep $N \in \{50, 100, 200\}$ for $K = 5$ concepts, and then fix $N = 100$ and sweep $Q \in \{50, 100, 200\}$.

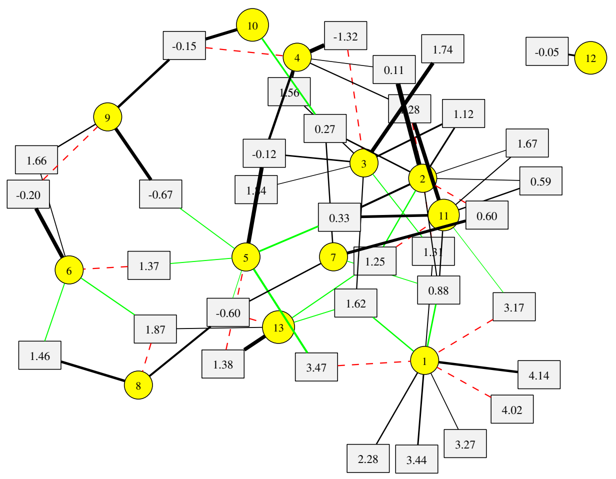
Impact of problem size: We first study the performance of Ordinal SPARFA-M versus K-SVD$_+$ while varying the problem size parameters Q and N. The corresponding box-and-whisker plots of the estimation error for each algorithm are shown in Fig. 1. In Fig. 1(a), we fix the number of questions Q and plot the errors $E_{\mathbf{W}}$, $E_{\mathbf{C}}$ and $E_{\boldsymbol{\mu}}$ for the number of learners $N \in \{50, 100, 200\}$. In Fig. 1(b), we fix the number of learners N and plot the errors $E_{\mathbf{W}}$, $E_{\mathbf{C}}$ and $E_{\boldsymbol{\mu}}$ for the number of questions $Q \in \{50, 100, 200\}$. It is evident that $E_{\mathbf{W}}$, $E_{\mathbf{C}}$, and $E_{\boldsymbol{\mu}}$ decrease as the problem size increases for all considered algorithms. Moreover, Ordinal SPARFA-M has superior performance to K-SVD$_+$ in all cases and for all error metrics. Ordinal SPARFA-Tag and the oracle support provided versions of K-SVD outperform Ordinal SPARFA-M and K-SVD$_+$. We furthermore see that the variant of Ordinal SPARFA-M without knowledge of the precision τ performs as well as knowing τ; this implies that we can accurately learn the precision parameter directly from data.

Impact of the number of quantization bins: We now consider the effect of the number of quantization bins P in the observation matrix $\mathbf{Y}$ on the performance of our algorithms. We fix $N = Q = 100$, $K = 5$ and generate synthetic data as before up to $\mathbf{Z}$ in (3). For this experiment, a different number of bins P is used to quantize $\mathbf{Z}$ into $\mathbf{Y}$. The quantization boundaries are set to $\{\Phi^{-1}(0), \Phi^{-1}(1/P), \ldots, \Phi^{-1}(1)\}$. To study the impact of the number of bins needed for Ordinal SPARFA-M to provide accurate factor estimates that are comparable to algorithms operating with real-valued observations, we also run K-SVD$_+$ directly on the $\mathbf{Z}$ values (recall (3)) as a base-line. Figure 2 shows that the performance of Ordinal SPARFA-M consistently outperforms K-SVD$_+$. We furthermore see that all error measures decrease by about half when using 6 bins, compared to 2 bins (corresponding to binary data). Hence, ordinal SPARFA-M clearly outperforms the conventional SPARFA model [24], when ordinal response data is available. As expected, Ordinal SPARFA-M approaches the performance of K-SVD$_+$ operating directly on $\mathbf{Z}$ (unquantized data) as the number of quantization bins P increases.

4.2 Real-world data

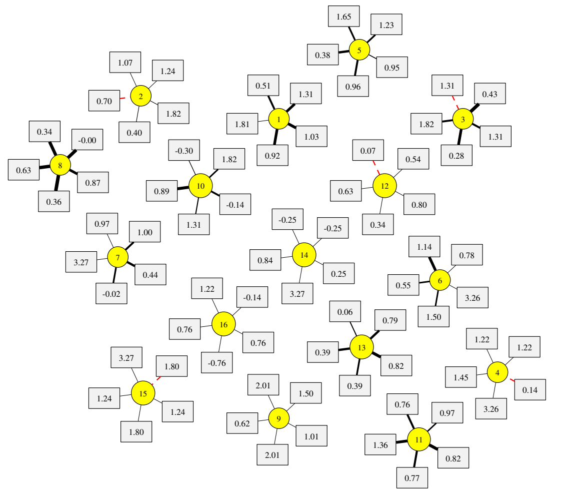
We now demonstrate the superiority of Ordinal SPARFA-Tag compared to regular SPARFA as in [24]. In particular, we show the advantages of using tag information directly within the estimation algorithm and of imposing a nuclear norm constraint on the matrix $\mathbf{C}$. For all experiments, we apply Ordinal SPARFA-Tag to the graded learner response matrix $\mathbf{Y}$ with oracle support information obtained from instructor-provided question tags. The parameters λ and η are selected via cross-validation.

Algebra test: We analyze a dataset from a high school algebra test carried out on Amazon Mechanical Turk [2], a crowd-sourcing marketplace. The dataset consists of $N = 99$ users answering $Q = 34$ multiple-choice questions covering topics such as geometry, equation solving, and visualizing function graphs. The questions were manually labeled with a set of 13 tags. The dataset is fully populated, with no



Concept 1	Concept 2	Concept 3	Concept 4	Concept 5
Arithmetic	Simplifying expressions	Solving equations	Fractions	Quadratic functions
Concept 6	**Concept 7**	**Concept 8**	**Concept 9**	**Concept 10**
Geometry	Inequality	Slope	Trigonometry	Limits
Concept 11	**Concept 12**	**Concept 13**		
Polynomials	System equations	Plotting functions		

Figure 3: Question–concept association graph for a high-school algebra test with $N = 99$ users answering $Q = 34$ questions. Boxes represent questions; circles represent concepts. We furthermore show the unique tag associated with each concept.



Concept 1	Concept 2	Concept 3	Concept 4	Concept 5	
Classifying matter	Properties of water	Mixtures and solutions	Changes from heat	Uses of energy	
Concept 6	**Concept 7**	**Concept 8**	**Concept 9**	**Concept 10**	
Circuits and electricity	Forces and motion	Formation of fossil fuels	Changes to land	Evidence of the past	
Concept 11	**Concept 12**	**Concept 13**	**Concept 14**	**Concept 15**	**Concept 16**
Earth, sun and moon	Alternative energy	Properties of soil	Earth's forces	Food webs	Environmental changes

Figure 4: Question–concept association graph for a grade 8 Earth Science course with $N = 145$ learners answering $Q = 80$ questions (Y is highly incomplete with only 13.5% entries observed). We furthermore show the unique tag associated with each concept.

missing entries. A domain expert manually mapped each possible answer to one of $P = 4$ bins, i.e., assigned partial credit to each choice as follows: totally wrong ($p = 1$), wrong ($p = 2$), mostly correct ($p = 3$), and correct ($p = 4$).

Figure 3 shows the question–concept association map estimated by Ordinal SPARFA-Tag using the Frobenius norm constraint $\|\mathbf{C}\|_F \leq \eta$. Circles represent concepts, and squares represent questions (labelled by their intrinsic difficulties μ_i). Large positive values of μ_i indicate easy questions; negative values indicate hard questions. Connecting lines indicate whether a concept is present in a question; thicker lines represent stronger question–concept associations. Black lines represent the question–concept associations estimated by Ordinal SPARFA-Tag, corresponding to the entries in $\mathbf{W}$ as specified by Γ. Red, dashed lines represent the "mislabeled" associations (entries of $\mathbf{W}$ in Γ) that are estimated to be zero. Green solid lines represent new discovered associations, i.e., entries in $\mathbf{W}$ that were not in Γ that were discovered by Ordinal SPARFA-Tag.

By comparing Fig. 3 with [24, Fig. 9], we can see that Ordinal SPARFA-Tag provides unique concept labels, i.e., one tag is associated with one concept; this enables precise interpretable feedback to individual learners, as the values in $\mathbf{C}$ represent directly the tag knowledge profile for each learner. This tag knowledge profile can be used by a PLS to provide targeted feedback to learners. The estimated question–concept association matrix can also serve as useful tool to domain experts or course instructors, as they indicate missing and inexistent tag–question associations.

Grade 8 Earth Science course: As a second example of Ordinal SPARFA-Tag, we analyze a Grade 8 Earth Science course dataset [31]. This dataset contains $N = 145$ learners answering $Q = 80$ questions and is highly incomplete (only 13.5% entries of $\mathbf{Y}$ are observed). The matrix $\mathbf{Y}$ is binary-valued; domain experts labeled all questions with 16 tags.

The result of Ordinal SPARFA-Tag with the nuclear norm constraint $\|\mathbf{C}\|_* \leq \eta$ on $\mathbf{Y}$ is shown in Fig. 4. The estimated question–concept associations mostly matches those pre-defined by domain experts. Note that our algorithm identified some question–concept associations to be non-existent (indicated with red dashed lines). Moreover, no new associations have been discovered, verifying the accuracy of the pre-specified question tags from domain experts. Comparing to the question–concept association graph of the high school algebra test in Fig. 3, we see that for this dataset, the pre-specified tags represent disjoint knowledge components, which is indeed the case in the underlying question set. Interestingly, the estimated concept matrix $\mathbf{C}$ has rank 3; note that we are estimating $K = 16$ concepts. This observation suggests that all learners can be accurately represented by a linear combination of only 3 different "eigen-learner" vectors. Further investigation of this clustering phenomenon is part of on-going research.

4.3 Predicting unobserved learner responses

We now compare the prediction performance of ordinal SPARFA-M on unobserved learner responses against state-of-the-art collaborative filtering techniques: (i) SVD++ in [20], which treats ordinal values as real numbers, and

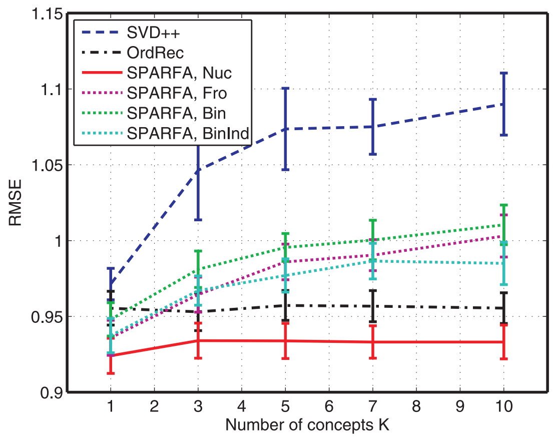


Figure 5: Prediction performance on the Mechanical Turk algebra test dataset. We compare the collaborative filtering methods SVD++ and OrdRec to various Ordinal SPARFA-M based methods: "Nuc" uses the nuclear norm constraint, "Fro" uses the Frobenius norm constraint, "Bin" and "BinInd" learn the bin boundaries, whereas "Bin" learns one set of bin boundaries for the entire dataset and "BinInd" learns individual bin boundaries for each question.

(ii) OrdRec in [21], which relies on an ordinal logit model. We compare different variants of Ordinal SPARFA-M: (i) optimizing the precision parameter, (ii) optimizing a set of bins for all learners, (iii) optimizing a set of bins for each question, and (iv) using the nuclear norm constraint on **C**. We consider the Mechanical Turk algebra test, hold out 20% of the observed learner responses as test sets, and train all algorithms on the rest. The regularization parameters of all algorithms are selected using 4-fold cross-validation on the training set. Figure 5 shows the root mean square error (RMSE) $\sqrt{\frac{1}{|\bar{\Omega}_{\text{obs}}|}\sum_{i,j:(i,j)\in\bar{\Omega}_{\text{obs}}}\|\hat{Y}_{i,j}-Y_{i,j}\|_2^2}$ where $\hat{Y}_{i,j}$ is the predicted score for $Y_{i,j}$, averaged over 50 trials.

Figure 5 demonstrates that the nuclear norm variant of Ordinal SPARFA-M outperforms OrdRec, while the performance of other variants of ordinal SPARFA are comparable to OrdRec. SVD++ performs worse than all compared methods, suggesting that the use of a probabilistic model considering ordinal observations enables accurate predictions on unobserved responses. We furthermore observe that the variants of Ordinal SPARFA-M that optimize the precision parameter or bin boundaries deliver almost identical performance.

We finally emphasize that Ordinal SPARFA-M not only delivers superior prediction performance over the two state-of-the-art collaborative filtering techniques in predicting learner responses, but it also provides interpretable factors, which is key in educational applications.

5. RELATED WORK

A range of different ML algorithms have been applied in educational contexts. Bayesian belief networks have been successfully used to probabilistically model and analyze learner response data in order to trace learner concept knowledge and estimate question difficulty (see, e.g., [13, 22, 33, 34]). Such models, however, rely on predefined question–concept dependencies (that are not necessarily accurate), in contrast to the framework presented here that estimates the dependencies solely from data.

Item response theory (IRT) uses a statistical model to analyze and score graded question response data [25, 29]. Our proposed statistical model shares some similarity to the Rasch model [28], the additive factor model [10], learning factor analysis [19, 27], and the instructional factors model [11]. These models, however, rely on pre-defined question features, do not support disciplined algorithms to estimate the model parameters solely from learner response data, or do not produce interpretable estimated factors. Several publications have studied factor analysis approaches on learner responses [3, 14, 32], but treat learner responses as real and deterministic values rather than ordinal values determined by statistical quantities. Several other results have considered probabilistic models in order to characterize learner responses [5, 6], but consider only binary-valued responses and cannot be generalized naturally to ordinal data.

While some ordinal factor analysis methods, e.g., [21], have been successful in predicting missing entries in datasets from ordinal observations, our model enables interpretability of the estimated factors, due to (i) the additional structure imposed on the learner–concept matrix (non-negativity combined with sparsity) and (ii) the fact that we associate unique tags to each concept within the estimation algorithm.

6. CONCLUSIONS

We have significantly extended the SPARse Factor Analysis (SPARFA) framework of [24] to exploit (i) ordinal learner question responses and (ii) instructor generated tags on questions as oracle support information on the question–concept associations. We have developed a computationally efficient new algorithm to compute an approximate solution to the associated ordinal factor-analysis problem. Our proposed *Ordinal SPARFA-Tag* framework not only estimates the strengths of the pre-defined question–concept associations provided by the instructor but can also discover new associations. Moreover, the algorithm is capable of imposing a nuclear norm constraint on the learner concept knowledge matrix, which achieves better prediction performance on unobserved learner responses than state-of-the-art collaborative filtering techniques, while improving the interpretability of the estimated concepts relative to the user-defined tags.

The Ordinal SPARFA-Tag framework enables a PLS to provide readily interpretable feedback to learners about their latent concept knowledge. The tag-knowledge profile can, for example, be used to make personalized recommendations to learners, such as recommending remedial or enrichment material to learners according to their tag (or concept) knowledge status. Instructors also benefit from the capability to discover new question–concept associations underlying their learning materials.

7. ACKNOWLEDGMENTS

Thanks to Daniel Calderon for administering the Algebra test on Amanzon's Mechanical Turk. This work was supported by the National Science Foundation under Cyberlearning grant IIS-1124535, the Air Force Office of Scientific

Research under grant FA9550-09-1-0432, the Google Faculty Research Award program, and the Swiss National Science Foundation under grant PA00P2-134155.

8. REFERENCES

[1] M. Aharon, M. Elad, and A. M. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans Sig. Proc.*, 54(11):4311–4322, Dec. 2006.

[2] Amazon Mechanical Turk. http://www.mturk.com/mturk/welcome, Sep. 2012.

[3] T. Barnes. The Q-matrix method: Mining student response data for knowledge. In *Proc. AAAI EDM Workshop*, July 2005.

[4] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. on Imaging Science*, 2(1):183–202, Mar. 2009.

[5] B. Beheshti, M. Desmarais, and R. Naceur. Methods to find the number of latent skills. In *Proc. 5th Intl. Conf. on EDM*, pages 81–86, June 2012.

[6] Y. Bergner, S. Droschler, G. Kortemeyer, S. Rayyan, D. Seaton, and D. Pritchard. Model-based collaborative filtering analysis of student response data: Machine-learning item response theory. In *Proc. 5th Intl. Conf. on EDM*, pages 95–102, June 2012.

[7] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[8] P. Brusilovsky and C. Peylo. Adaptive and intelligent web-based educational systems. *Intl. J. of Artificial Intelligence in Education*, 13(2-4):159–172, Apr. 2003.

[9] J. F. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM J. on Optimization*, 20(4):1956–1982, Mar. 2010.

[10] H. Cen, K. R. Koedinger, and B. Junker. Learning factors analysis–a general method for cognitive model evaluation and improvement. In M. Ikeda, K. D. Ashley, and T. W. Chan, editors, *Intelligent Tutoring Systems*, volume 4053 of *Lecture Notes in Computer Science*, pages 164–175. Springer, June 2006.

[11] M. Chi, K. Koedinger, G. Gordon, and P. Jordan. Instructional factors analysis: A cognitive model for multiple instructional interventions. In *Proc. 4th Intl. Conf. on EDM*, pages 61–70, July 2011.

[12] W. Chu and Z. Ghahramani. Gaussian processes for ordinal regression. *J. of Machine Learning Research*, 6:1019–1041, July 2005.

[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, Dec. 1994.

[14] M. Desmarais. Conditions for effectively deriving a Q-matrix from data with non-negative matrix factorization. In *Proc. 4th Intl. Conf. on EDM*, pages 41–50, July 2011.

[15] J. A. Dijksman and S. Khan. Khan Academy: The world's free virtual school. In *APS Meeting Abstracts*, page 14006, Mar. 2011.

[16] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the ℓ1-ball for learning in high dimensions. In *Proc. 25th Intl. Conf. on ML*, pages 272–279, July 2008.

[17] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning.* Springer, 2010.

[18] D. Hu. How Khan academy is using machine learning to assess student mastery. *Online: http://david-hu.com*, Nov. 2011.

[19] K. R. Koedinger, E. A. McLaughlin, and J. C. Stamper. Automated student model improvement. In *Proc. 5th Intl. Conf. on EDM*, pages 17–24, June 2012.

[20] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, Aug. 2009.

[21] Y. Koren and J. Sill. OrdRec: an ordinal model for predicting personalized item rating distributions. In *Proc. of the 5th ACM Conf. on Recommender Systems*, pages 117–124, Oct. 2011.

[22] G. A. Krudysz and J. H. McClellan. Collaborative system for signal processing education. In *Proc. IEEE ICASSP*, pages 2904–2907, May 2011.

[23] J. A. Kulik. Meta-analytic studies of findings on computer-based instruction. *Technology assessment in education and training*, pages 9–33, 1994.

[24] A. S. Lan, A. E. Waters, C. Studer, and R. G. Baraniuk. Sparse factor analysis for learning and content analytics. Oct. 2012, submitted.

[25] F. M. Lord. *Applications of Item Response Theory to Practical Testing Problems.* Erlbaum Associates, 1980.

[26] J. Nocedal and S. Wright. *Numerical Optimization.* Springer Verlag, 1999.

[27] P. I. Pavlik, H. Cen, and K. R. Koedinger. Learning factors transfer analysis: Using learning curve analysis to automatically generate domain models. In *Proc. 2nd Intl. Conf. on EDM*, pages 121–130, July 2009.

[28] G. Rasch. *Probabilistic Models for Some Intelligence and Attainment Tests.* MESA Press, 1993.

[29] M. D. Reckase. *Multidimensional Item Response Theory.* Springer Publishing Company Incorporated, 2009.

[30] C. Romero and S. Ventura. Educational data mining: A survey from 1995 to 2005. *Expert Systems with Applications*, 33(1):135–146, July 2007.

[31] STEMscopes Science Education. http://stemscopes.com, Sep. 2012.

[32] N. Thai-Nghe, T. Horvath, and L. Schmidt-Thieme. Factorization models for forecasting student performance. In *Proc. 4th Intl. Conf. on EDM*, pages 11–20, July 2011.

[33] K. Wauters, P. Desmet, and W. Van Den Noortgate. Acquiring item difficulty estimates: a collaborative effort of data and judgment. In *Proceedings of the 4th Intl. Conf. on EDM*, pages 121–128, July 2011.

[34] B. P. Woolf. *Building Intelligent Interactive Tutors: Student-centered Strategies for Revolutionizing E-learning.* Morgan Kaufman Publishers, 2008.

[35] Y. Xu and W. Yin. A block coordinate descent method for multi-convex optimization with applications to nonnegative tensor factorization and completion. Technical report, Rice University CAAM, Sep. 2012.

[36] A. Zymnis, S. Boyd, and E. Candès. Compressed sensing with quantized measurements. *IEEE Sig. Proc. Letters*, 17(2):149–152, Feb. 2010.

Discovering Student Models with a Clustering Algorithm Using Problem Content

Nan Li
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15232
nli1@cs.cmu.edu

William W. Cohen
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15232
wcohen@cs.cmu.edu

Kenneth R. Koedinger
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15232
koedinger@cs.cmu.edu

ABSTRACT

One of the key factors that affects automated tutoring systems in making instructional decisions is the quality of the student model built in the system. A student model is a model that can solve problems in various ways as human students. A good student model that matches with student behavior patterns often provides useful information on learning task difficulty and transfer of learning between related problems, and thus often yields better instruction on intelligent tutoring systems. However, traditional ways of constructing such models are often time consuming, and may still miss distinctions in content and learning that have important instructional implications. Automated methods can be used to find better student models, but usually require some engineering effort, and can be hard to interpret. In this paper, we propose an automated approach that finds student models using a clustering algorithm based on automatically-generated problem content features. We demonstrate the proposed approach using an algebra dataset. Experimental results show that the discovered model is as good as one of the best existing models, which is a model found by a previous automated approach, but without the knowledge engineering effort.

Keywords

student model, machine learning, learner modeling

1. INTRODUCTION

A student model is an essential component in intelligent tutoring systems. It encodes how to solve problems in various ways as human students do. One common way of representing such student models is a set of *knowledge components (KC)* encoded in intelligent tutors to model how students solve problems. As defined in [9], a knowledge component is an *acquired* unit of cognitive function or structure that can be *inferred* from performance on *a set of related tasks*. The set of KCs includes the component skills, concepts, or percepts that a student must acquire to be successful on the target tasks. For example, a KC in algebra can be how students should proceed given problems of the form Nv=N (e.g. $3x = 6$). A student model provides automated tutoring systems with important information on how to make instructional decisions. Better student models are capable of predicting task difficulty and transfer of learning between related problems. Thus, intelligent tutoring systems with better student models often provide more effective learning experience.

Traditional ways to construct student models include structured interviews, think-aloud protocols, rational analysis, and so on. However, these methods are often time-consuming, and require domain expertise. More importantly, they are highly subjective. Previous studies [11] have shown that human engineering of these models often miss components of knowledge acquisition (e.g., that learning to read algebraic sentences is difficult) that have important instructional implications. Other methods that apply machine learning techniques to generate student models [16, 27] can find models that are better than human-generated ones, but may suffer from challenges in interpreting the results. For example, Learning Factor Analysis (LFA) [6] apply an automated search technique to discover student models. Nevertheless, one key limitation of LFA is that it carries out the search process only within the space of human-provided factors. If a better model exists but requires unknown factors, LFA will not find it. Another approach is to use a learning agent, *SimStudent*, to automatically discover student models [15]. Although this method is less dependent on human-provided factors, it still needs some knowledge engineering effort in constructing the learning agent.

To address this issue, we formulate the student model discovery approach as a clustering problem, and propose another automated method that discovers student models using a machine learning algorithm, *k-means*, based on automatically-generated features. To accommodate for both the performance prediction accuracy and the interpretability of the discovered model, the features include both problem content features and performance features, so that problem steps in the same cluster are of similar forms and are associated with similar performance on human students. Each cluster corresponds to a KC that students need to learn. We evaluated the approach in algebra using real student data. Experiment results show that the discovered model fits with real student data as good as the model found by SimStudent.

Table 1: An Example List of Steps with Their Content Features.

Step	Tokenized Step	-N	Nv	v=	=N	-Nv	Nv=	v=N	v+	+N	N=	Nv+	v+N	+N=	N=N
-3x = 6	-Nv=N	1	1	1	1	1	1	1	0	0	0	0	0	0	0
2y+5=7	Nv+N=N	0	1	0	1	0	0	0	1	1	1	1	1	1	1

In the following sections, we start by describing how to statistically evaluate the quality of a student model. Then, we explain how to generate features and to apply a clustering algorithm to find student models that meet such criteria. Next, we report experimental results on the comparison between the clustering-based model and the SimStudent model, along with an in-depth study using a recently developed analysis technique, Focused Benefits Investigation (FBI) [10]. After this, we discuss the generality of the proposed approach, and possible improvements that can be made using SimStudent. In closing, we describe some related work as well as conclusions drawn from this work.

2. STATISTICAL EVALUATION OF STUDENT MODEL QUALITY

As we have mentioned before, a student model can be represented by a set of knowledge components, where each problem step is associated with one KC that encodes how to proceed given the current step. Therefore, the problems we have is that given a dataset recording how human students solve problems in one domain, how to find a set of KCs that matches with student behavior well.

There are various ways of matching a student model with student data. Although other models are also possible (e.g. [8], in our case, we use the Additive Factor Model (AFM) [6] to measure the quality of a student model. AFM is an instance of logistic regression that models student success using each student, each KC, and the KC by opportunity interaction as independent variables,

$$\ln \frac{p_{ij}}{1-p_{ij}} = \theta_i + \sum_k \beta_k Q_{kj} + \sum_k \beta_k Q_{kj}(\gamma_k N_{ik})$$

Where:

i represents a student i.

j represents a step j.

k represents a skill or KC k.

p_{ij} is the probability that student i would be correct on step j.

θ_i is the coefficient for proficiency of student i.

β_k is coefficient for difficulty of the skill or KC k.

Q_{kj} is the Q-matrix cell for step j using skill k.

γ_k is the coefficient for the learning rate of skill k.

N_{ik} is the number of practice opportunities student i has had on the skill k.

Hence, the better the student model is; the more accurate the predictions are. To train the parameters, we use maximum-likelihood estimation (MLE). In order to avoid overfitting, we use cross-validation (CV) to validate the quality of the student model.

3. STUDENT MODEL DISCOVERY USING A MACHINE LEARNING ALGORITHM

Given the above evaluation method, we would like to note that our task here is not only to find a model that predicts student behavior well, we also want to find a model that is conceptually meaningful. In other words, steps within the same KC should be both conceptually similar and performance-wise similar. In fact, finding a student model over a set of problem steps is a *clustering task*, where the algorithm groups a set of problem steps in a way that steps in the same group (called cluster) are more similar in some sense to each other than to those in other groups (clusters). In our case, each cluster corresponds to a KC in the student model. From this clustering point of view, if the metric of similarity measures both content similarity and performance similarity, the KCs that the algorithm finds would have the desired properties we discussed above. Therefore, we use two types of features for clustering, *content features* and *performance features.*

3.1 Preprocessing

Before generalizing the features, we first tokenize the problem steps, so that all numbers are replaced by N, and all variables are represented as v. For example, the tokenized representation of $-3x = 6$ is $-Nv = N$. In fact, the level of tokenization affects the result of the discovered model, since this preprocessing step removes the difference among steps that are of the same form but with different numbers. This may cause problems in some cases. For instance, solving $-3x = 6$ can be potentially much easier than solving $-452x = 904$, but the preprocessing step gives both steps the same tokenized representation $-Nv = N$. As we will discuss later, by making use of SimStudent, we could automatically get different levels of tokenization.

3.2 Feature Generation

After preprocessing, we now generate features for these tokenized steps. There are two types of features, content features and performance features.

3.2.1 Content Features

Content features are defined based on the problem content information of the tokenized steps. More specifically, we generate all of the bigrams and trigrams in each of the tokenized steps. For each bigram or trigram, we set the value of that feature to be 1 if the bigram or trigram appears in the current step, and 0 otherwise.

Table 2: The List of Performance Features Used for Clustering.

Feature	Meaning
Avg. Incorrects	Average number of incorrect attempts for the current step
Avg. Hints	Average number of the student asking for a hint for the current step
Avg. Corrects	Average number of correct attempts for the current step
% First Attempt Incorrects	The percentage of times that the first attempt is incorrect
% First Attempt Hints	The percentage of times that the first attempt is asking hint
% First Attempt Corrects	The percentage of times that the first attempt is correct
Avg. Step Duration (sec)	Average number of seconds the student spending on this step
Avg. Correct Step Duration (sec)	Average number of seconds the student spending on this step when the student gets this step correct
Avg. Error Step Duration (sec)	Average number of seconds the student spending on this step when the student gets this step incorrect
Total Students	Average number of total students working on this step
Total Opportunities	Average number of total opportunities that the student has in solving the current step

For example, for step $-Nv = N$, all of the bigram features it generates are $-N$, Nv, $v =$, and $= N$, and all of the trigram features it generates are $-Nv$, $Nv =$, and $v = N$. Considering a bigram feature Nv, and a trigram feature $+N =$, for step $-Nv = N$, the value of the feature Nv is 1, whereas the value of the feature $+N =$ is 0, since $+N =$ does not appear in $-Nv = N$. But for another step $Nv + N = N$, both the value of Nv and the value of $+N =$ are 1, since both of them appear in $Nv + N = N$. The trigram feature $+N =$ here can be used to identify the steps for subtracting both sides with N. Table 1 shows an example list of steps with their content features.

By using these content features, we make sure that the steps in the same cluster share some common content features, and thus look similar to each other. This satisfies the property of having conceptually-similar steps in the same cluster. Moreover, by having steps that share content features clustered in one KC, it is easier for human to interpret the results.

3.2.2 *Performance Features*

The second set of features we used in the algorithm are performance features. These features measure the average performance of human students on each format of the tokenized steps. Examples of such measurements are the time to response, and whether the student's first attempt was correct. Table 2 shows the full list of performance features used for clustering.

Note that performance features are only used to create the clusters of the training data. Since we are predicting the performance of human students, performance data should not be used in testing. For testing data, we only use the content features to assign the cluster of the current step. In other words, for each testing data point, we calculate the distance of the data point to all of the training data points based on perceptual features, and assign the testing data point to the cluster associated with the closest training data point.

3.3 Principal Component Analysis

Before clustering, we normalize all the features to range from 0 to 1. Then, we perform a principal component analysis

Algorithm 1: K-Means

```
Input: Points to be clustered $P$, Number of clusters $k$
Output: Cluster centroids $C$, cluster membership $M$.
1  initialize $C$ with $k$ randomly selected data points in $P$
2  forall the $p_i \in P$ do
3      $m_i := argmin_{j \in 1..k} distance(p_i, c_j)$
4  end
5  while $m$ changed do
6      foreach $i \in \{1..n\}$ do
7          Recompute $c_i$ as the centroid of $\{p_j | m_j = i\}$
8      end
9      sum_ratios := 0
10     forall the $p \in c \cap d$ do
11         sum_ratios += $w_c(p)/w_d(p)$
12     end
13     forall the $p_i \in P$ do
14         $m_i := argmin_{j \in 1..k} distance(p_i, c_j)$
15     end
16 end
17 return $C$, $M$
```

over the features we generated. Principal component analysis is a mathematical procedure that projects a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables. These linearly uncorrelated variables are called principal components. The first principal component points to the direction that accounts for the largest possible variance. The succeeding components are orthogonal to the previous components, and account for smaller variance.

After this transformation process, all of the features in the projected space are orthogonal to each other. Moreover, in order to remove less informative features, we only select the first 40 principal components in the projected space. It covers approximately 95% of the variance in the data.

3.4 Student Model Discovery with a Clustering Algorithm

To discover student models, we use k-means to cluster the data over the automatically-generated features. The dis-

Table 3: Cross Validation Results on the Clustering-Based Model and the SimStudent Model.

	SimStudent RMSE	Clustering RMSE
Run 1	0.4105	**0.4102**
Run 2	0.4109	**0.4106**
Run 3	0.4113	**0.4105**
Run 4	**0.4107**	0.4111
Run 5	0.4106	**0.4095**
Run 6	0.4109	**0.4102**
Average	0.4108	**0.4104**

tance between data points is measured by the Euclidean distance in the feature space.

The algorithm uses an expectation-maximization style approach. Algorithm 1 shows the psuedocode of the clustering procedure. Fristly, the algorithm randomly selects k points as the initial centers of each cluster. Then, in the assignment step, the rest of the points are assigned to the cluster whose mean is closest to it among all of the existing clusters. Next, in the update step, the algorithm calculates the new means of the new clusters as the centroids of the data points. This process continues until converge.

K-means needs the number of clusters k to be given as input. Since we do not know how many clusters are there, we set the number of clusters to be 20, 25, and 30. The algorithm then picks the one with the best cross validation result[1]. Note that even with the same number of clusters, different initialization of the clusters can lead to different clustering results. In this study, we just run k-means once for each value k. In future study, we could run the clustering algorithm multiple times, and select the clusters that have the smallest intra-cluster difference and the largest inter-cluster difference.

4. EXPERIMENT STUDY

In order to evaluate the effectiveness of the proposed approach, we carried out a study using an algebra dataset. We compared the clustering-based model with a SimStudent model. The SimStudent model is discovered by a learning agent, which is also one of the best student models we have in the database.

4.1 Method

To generate the SimStudent model, SimStudent was tutored on how to solve linear equations by interacting with a Carnegie Learning Algebra I Tutor like a human student. We selected 40 problems that were used to teach real students as the training set for SimStudent. Given all of the acquired production rules, for each step a real student performed, we assigned the applicable production rule as the KC associated with that step. In cases where there was no applicable production rule, we coded the step using a human-generated KC model (Balanced-Action-Typein). The human-generated model is the best model constructed by domain experts. It has been shown that the SimStudent

[1]We tried smaller numbers, but it turns out that when k is between 20 and 30, the cross validation result is often better.

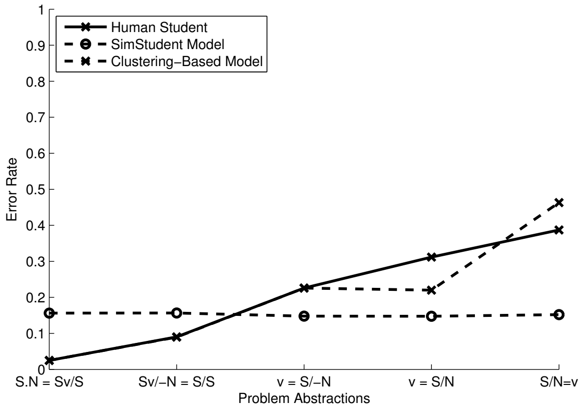


Figure 1: Error rates of human students and predicted error rates of two student models. S stands for a signed number, N represents an integer, and v is a variable.

model is better than the human-generated model, and provides useful instructional implications.

The clustering-based model was discovered using the approach described above. Each time a student encounters a step using some KC is considered as an "opportunity" for that student to show mastery of that KC. In both models, a total of 6507 steps are coded.

In order to get a better understanding on how the clustering-based model differs from other student models, we further utilized DataShop, a large repository that contains datasets from various educational domains as well as a set of associated visualization and analysis tools, to facilitate the process of evaluation, which includes generating learning curve visualization, AFM parameter estimation, and evaluation statistics including AIC (Akaike Information Criterion) and cross validation.

4.2 Dataset

We analyzed data from 71 students who used an Carnegie Learning Algebra I Tutor unit on equation solving. The students were typical students at a vocational-technical school in a rural/suburban area outside of Pittsburgh, PA. The problems varied in complexity, for example, from simpler problems like *3x=6* to harder problems like *x/-5+7=2*. A total of 19,683 transactions between the students and the Algebra Tutor were recorded, where each transaction represents an attempt or inquiry made by the student, and the feedback given by the tutor.

4.3 Measurements

To test whether the generated model fits with real student data, we used 10-fold cross validation. The cross validation was performed over ten folds with the constraint that each of the three training sets must have data points for each student and KC. For the clustering-based model, performance features of the testing steps were not used in constructing the KCs. We calculated the root mean-squared error (RMSE) averaged over ten test sets. Due to the random nature of the fold generation process in cross validation, we repeated this process six times.

Table 4: **FBI Results on Selected KCs That Are Improved in the Clustering-Based Model.**

SimStudent KCs	SimStudent Model RMSE	Clustering-Based Model RMSE	% change of RMSE
ctat-divide	0.5289	0.3984	-24.67
ctat-distribute	0.4292	0.3553	-17.21
ctat-multiply	0.4634	0.3962	-14.50
ctat-clt	0.3757	0.3445	-8.325
ctat-divide-typein	0.3674	0.3368	-8.321

In order to better understand this machine learning approach, we carried out an in-depth study using FBI [10] on the clustering-based model and the SimStudent. FBI is a recently developed technique. It is designed to analyze which of the differences between the models improves the prediction the most, and by how much.

4.4 Experimental Results

As shown in Table 3, in five out of the six runs, the clustering-based models get lower RMSEs than the SimStudent model, which indicates that the clustering-based model is at least as good as the SimStudent model. Averaged over the six runs, the clustering-based models get an average RMSE of 0.4104, while the SimStudent model gets a slightly higher RMSE (i.e., 0.4108).

As you may have noticed, the difference between the RMSEs of the two models is small, but this does not mean that the difference between the two models is small. Instead of using cross validation to measure the quality of the model as a whole, we applied FBI to evaluate the difference at the knowledge component level. Table 4 shows the top five KCs in the SimStudent model that are improved in the clustering-based model. As we can see that all of these KCs' names start with "ctat", which means these KCs are from the human-generated model. Recall that in the SimStudent model, if SimStudent could not find any applicable production rule to a step, the step would be coded by the human-generated model. This suggests that the clustering-based approach is more general than the SimStudent approach in the sense that it is able to code steps that are not supported by SimStudent. Among the nine KCs generated by SimStudent, three of them were improved in the clustering-based student model.

In these five KCs, the clustering-based model successfully reduced the RMSE by at least 8%. In the KC "ctat-divide", the RMSE was reduced by around 25%. This indicates that the clustering-based approach is able to find KCs that are better than the existing ones. We can inspect the data more closely to get a better qualitative understanding of how the two models are different and what implications there might be for improved instruction.

We took a closer look at the KC "ctat-divide-typein". In the SimStudent model, all steps that require division are assigned to the "ctat-divide-typein" skill. However, there are differences among these steps. We checked the KCs in the clustering-based model associated with these steps, and found out that these steps were split into different KCs in the clustering-based model. Table 5 shows the five biggest KCs associated with the "ctat-divide-typein" steps. Since we used problem content based features, the KCs in the model were relatively easy to interpret. Each KC name (e.g., 25) in the table is followed by the most common form of the division steps in the KC (e.g., $S.N = Sv/S$), where N represents an integer, S means a signed number, and v stands for a variable. We calculated the average error rate of human students solving these steps, as well as the predicted error rates of the steps based on the two student models. As presented in Figure 1, the predicted error rates of the clustering-based model are closer to human students' actual error rates than the predicated error rates of the SimStudent model. Since the SimStudent model considers all these steps correspond to one KC, it predicts that they should have similar error rates, which is reflected by the flat line in Figure 1. The clustering-based model, on the other hand, predicts different error rates for problem steps of different forms.

More specifically, according to human student performance, steps associated with KC 25 are easier than problem steps from other KCs. A careful inspection at the data shows that KC 25 is associated with problem steps of the form $S.N = Sv/S$, which means that the left side of the equation is a decimal number. On the other hand, the problem steps in other KCs are associated with fractions. For steps with fractions, human students may have to simplify the fractions in order to get the final solution, whereas for steps with decimal numbers, students only need to copy the decimal numbers as the solution. Therefore, steps associated with KC 25 have a lower error rate than the other steps, which are correctly modeled by the clustering-based model. Moreover, among KC 6, KC 22, and KC 29, human students have a higher error rate when the variable is on the right side of the equation (i.e., steps associated with KC 6). This is also correctly captured by the clustering-based model, while the SimStudent model again incorrectly predicts similar error rates.

These results are confirmed in the FBI study as well. As shown in Table 5, the largest improvement comes from KC 25 reaches 40%, partially because it separates divide-typein problems with decimal numbers from problems with fractions. KC 15 further models problem steps that have the variable with coefficients from the other steps that have the single variable in one side of the equation. This contributes to an improvement around 8%. The other three KCs differentiate problem steps that have the variable in the left side of the equation from the ones that have the variable in the right side of the equation. Two out of these three KCs get better RMSE. The third KC's increase in RMSE is mainly caused by other none divide-typein steps.

Table 5: Selected KCs in the Clustering-Based Model That Correspond to KC "ctat-divide-typein".

Clustering KCs	Clustering-Based Model RMSE	SimStudent Model RMSE	% change of RMSE
25 (S.N = Sv/S)	0.1547	0.2170	40.34
6 (S/N=v)	0.4654	0.5516	18.54
15 (Sv/-N = S/S or S/S = Sv/-N)	0.2969	0.3205	7.939
22 (v = S/-N)	0.4194	0.4279	2.016
29 (v = S/N)	0.4238	0.4073	-3.898

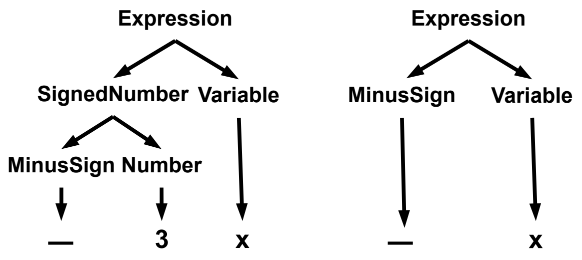


Figure 2: Different parse trees for -3x and -x.

The clustering algorithm's split of the original divide-typein KC into five KCs suggests that human students should be taught separately on each type of problems. More specifically, intelligent tutors cannot make the assumption that if students have learned the divide-typein KC for decimal problems, they will also know how to solve the fraction problems. Furthermore, the tutoring system should teach human students not only with problems that have the variables on the left side, but also with problems that have the variables at the right side of the equation, so that students get familiar with the concept that variables can be at the either side of the equation.

5. DISCUSSION

Given the promising results, we would like to further discuss some interesting future steps for this algorithm.

5.1 Automated vs. Manual Model Discovery

One question we should ask is that why we should use automated student model discovery approach rather than manual construction. This is mainly due to the fact that much of human expertise is only tacitly known. In many of the cases, we know how to solve the problems, while it can be hard to explain how we solved the problem. For instance, in language learning, native speakers can accurately select the correct article in a sentence, but do not know why they pick that article. Similarly, most algebra experts have no explicit awareness of subtle transformations they have acquired. Even though such instructional designers may be experts in a domain, they may still have some blind spots regarding subtle perceptual differences like this one, which may make a real difference for novice learners. A machine learning approach can help get past such blind spots by revealing challenges in the learning process that experts may not be aware of. In addition, these discovered KCs can serve as a basis for traditional ways of student model discovery.

5.2 Feature Generation Using SimStudent

Furthermore, in this paper, we simply use bigrams and trigrams of the tokenized steps as the content features. Some of these features may not be very helpful in differentiating KCs needed for the steps. Moreover, it is possible that different tokenization procedures and longer n-grams would lead to better results in other domains. We can, of course, keep adding new features in the feature space, and let the learning algorithm search through the larger space. However, this is not ideal due to the curse of dimensionality. In previous work, we have shown that hierarchical representations of the problem steps as shown in Figure 2 can be acquired by grammar induction techniques [14]. These hierarchical representations capture "deep features" in solving problems at different levels of abstractions. In the future, it would be interesting to see that whether we can make use of such representations to automatically generate high-quality content features, and lead to the discovery of better student models.

Moreover, the problem content features used in this paper are perceptual features. This is sufficient for domains like algebra, since the structure of the problem steps is enough to decide which skill to apply. But in other domains such as fraction addition, deciding whether two numbers are coprimed or not is impossible if using only perceptual features. In these cases, being able to generate operational features is required.

In response to this, we propose to use SimStudent to generate these features. SimStudent is an intelligent learning agent that uses machine learning techniques to acquire skills. It has three sets of prior knowledge, *a perceptual hierarchy*, *a set of operator functions*, and *a set of feature predicates.* Previous work has shown that by integrating representation learning with skill learning, instead of manually encoding this prior knowledge, the learning agent can learn, automatically generate, or partially reduce the need of such prior knowledge. The extended learning agent becomes a better model of student learning. To get a more general approach, we plan to make use of the acquired prior knowledge as well as the learned skills to generate both perceptual features and operational features.

5.3 Objective Function Guided Clustering

In this paper, the student model is discovered purely based on the clustering procedure. The discovered model is then used to fit with student data in the AFM model. In other words, the student model does not change once the clustering process is completed. Another interesting approach is to guide the student model discovery / clustering process by the fit to student performance data. Since the second approach is fully guided by the objective function, presumably, we could get a model with better predictions than the approach proposed in this paper.

However, there are two major issues with this objective function guided approach. As mentioned before, each knowledge component in a student model is an *acquired* unit of cognitive function or structure that can be *inferred* from performance on *a set of related tasks* [9]. If the student model is discovered purely based on the fit to student performance data, the KCs discovered may not be able to provide meaningful instructional insights. For example, if human students found both problems of the form $-v = N$ and $-N/v = N$ hard, does that mean that the tutor should teach these problem steps together? One possible way to address this issue is to also include the problem content similarity measurement in the objective function, so that the search is guided not only by performance, but also by task similarity.

Another issue is that this objective function guided approach often takes longer, as it has to fit the model with the data on each node expansion during the search. Therefore, in this paper, we take the clustering approach since it is more efficient, and can find KCs that are easier to interpret. In the next study, it would be interesting to compare the proposed approach with the objective function guided approach.

5.4 Other Clustering Techniques

One additional possible study is to try other clustering techniques. In this work, we only applied k-means to discover student models. There are other clustering algorithms such as hierarchical agglomerative clustering and spectral clustering [19]. These clustering algorithms have different properties, and may be better fit with the student model discovery task. In the future, we would like to further explore in this direction with other clustering techniques.

5.5 Generality

The last study we are interested in carrying out is to test the generality of the proposed approach. The Pittsburgh of Science of Learning Center's DataShop contains over 200 datasets in algebra and other domains that could be used for such cross-dataset or cross-domain validation. The current study used a single dataset in a single domain. The generality and validity of the proposed student-modeling technique could be extended by clustering problem steps in one dataset and applying the discovered KC model to other datasets. For example, the dataset we used is associated with students in one high school. It would be interesting to see whether the generated student model applies to other high schools at the same level.

In addition, we plan to apply this approach in other domains such as stoichiometry, fraction addition and so on. As we have mentioned above, it is possible that operational features are also needed in these domains. In this case, extending the current approach with other learning techniques such as SimStudent would be a promising future step. On the other hand, the language learning domain does not require complex problem solving, but needs complex perceptual knowledge and large amounts of background knowledge. An interesting future work is to apply existing linguistic tools to English sentences, and then automatically generate problem content features based on the parsed sentences.

6. RELATED WORK

The objective of this paper is using a clustering algorithm to automatically construct student models. A lot of efforts have also been put toward comparing the quality of alternative student models. LFA automatically discovers student models, but is limited to the space of the human-provided factors. SimStudent is less dependent on human-provided factors, but still needs some knowledge engineering effort in constructing the agent. Moreover, as we have shown in the experiments, the clustering based algorithm is able to find KCs that are better than those found by SimStudent. Other works such as [16, 27] are less dependent on human labeling, but may suffer from challenges in interpreting the results. In contrast, the clustering-based approach has the benefit that the acquired KCs usually have a straightforward interpretation. Baffes and Mooney [2] apply theory refinement to the problem of modeling incorrect student behavior. Other systems [23, 3] use Q-matrix to find knowledge structure from student response data. Our approach also uses machine learning algorithms to discover student models. In addition to model student performance, we emphasize on the interpretability of the models by adding content features to the clustering approach.

There has also been considerable amount of research on using artificial intelligence and machine learning techniques to model human students. Langley and Ohlsson's [13] ACM applies symbolic machine learning techniques to automatically construct student models. Brown and Burton's [5] DEBUGGY, and Sleeman and Smith's [20] LMS also make use of artificial intelligent tools to construct models that explain student's behavior in math domains. VanLehn's [25] Sierra models the impasse-driven acquisition of hierarchical procedures for multi-column subtraction from sample solutions. Research on models of high-level learning [12, 1, 22, 21, 24, 18] is also closely related to our work, but to the best of our knowledge, has not been evaluated by the fit to student learning curve data as we do in this work. In addition, most of these work took a more symbolic approach, while our algorithm is more statistical based.

Other research on creating simulated students [26, 7, 17] also share some resemblance to our work. VanLehn [25] created a learning system and evaluated whether it was able to learn procedural "bugs" like real students. Biswas et al.'s [4] system learns causal relations from a conceptual map created by students. None of the above approaches except for the SimStudent model discovery approach compared the system with learning curve data. To the best of our knowledge, our work is the very few who combines the two whereby we use cognitive model evaluation techniques to assess the quality of a simulated learner.

7. CONCLUSION

In this paper, we introduced an innovative application of a machine learning algorithm for an automatic discovery of student models. In order to discover KCs that are effective in predicting human student performance, while being easy to interpret, we added problem content features in the feature space, and applied a clustering algorithm to find student models. Our evaluation demonstrated that discovering student models based on problem content features was able to produce models of good prediction accuracies, and showed how the discovered model could provide important instructional implications. We further discussed possible extensions to the existing approach, and described how a learning agent such as SimStudent can be used to automatically generate content features as well as operational features to improve the generality of the proposed approach. This work is one step forward in applying machine learning techniques to construct student model. We believe that there are a lot of of fruitful future steps in this direction. They are natural extensions under the current framework.

8. ACKNOWLEDGEMENTS

The research reported here was supported by the Pittsburgh Science of Learning Center, which is funded by the National Science Foundation Award No. SBE-0836012. The authors would also like to thank Mike Wixon and Dandiel Seaton for helpful discussions, and Hui Cheng for running the experiments.

9. REFERENCES

[1] J. R. Anderson. *Rules of the Mind*. Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1993.

[2] P. T. Baffes and R. J. Mooney. A novel application of theory refinement to student modeling. In *Proceedings of the thirteenth national conference on Artificial intelligence*, pages 403–408. AAAI Press, 1996.

[3] T. Barnes. The Q-matrix method: Mining student response data for knowledge. In *Proceedings AAAI Workshop Educational Data Mining*, pages 1–8, Pittsburgh, PA, 2005.

[4] G. Biswas, D. Schwartz, K. Leelawong, and N. Vye. Learning by teaching: A new agent paradigm for educational software. *Applied Artificial Intelligence*, 19:363–392, March 2005.

[5] R. R. Burton. Diagnosing bugs in a simple procedural skill. In *Intelligent Tutoring Systems*, pages 157–184. Academic Press, 1982.

[6] H. Cen, K. Koedinger, and B. Junker. Learning factors analysis - a general method for cognitive model evaluation and improvement. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, pages 164–175, 2006.

[7] T.-W. Chan and C.-Y. Chou. Exploring the design of computer supports for reciprocal tutoring. *International Journal of Artificial Intelligence in Education*, 8:1–29, 1997.

[8] Y. Gong, J. E. Beck, and N. T. Heffernan. Comparing knowledge tracing and performance factor analysis by using multiple model fitting procedures. In *Proceedings of the 10th international conference on Intelligent Tutoring Systems - Volume Part I*, pages 35–44, Berlin, Heidelberg, 2010. Springer-Verlag.

[9] K. R. Koedinger, A. T. Corbett, and C. Perfetti. The Knowledge-Learning-Instruction (KLI) Framework : Toward Bridging the Science-Practice Chasm to Enhance Robust Student Learning. *Cognitive Science*, 36(5):757–798, 2012.

[10] K. R. Koedinger, E. A. McLaughlin, and J. C. Stamper. Automated student model improvement. In *Proceedings of the 5th International Conference on Educational Data Mining*, pages 17–24, Chania, Greece, 2012.

[11] K. R. Koedinger and M. J. Nathan. The real story behind story problems: Effects of representations on quantitative reasoning. *The Journal of Learning Sciences*, 13(2):129–164, 2004.

[12] J. E. Laird, P. S. Rosenbloom, and A. Newell. Chunking in soar: The anatomy of a general learning mechanism. *Machine Learning*, 1:11–46, 1986.

[13] P. Langley and S. Ohlsson. Automated cognitive modeling. In *Proceedings of the Fourth National Conference on Artificial Intelligence*, pages 193–197, Austin, TX, 1984. Morgan Kaufmann.

[14] N. Li, W. W. Cohen, and K. R. Koedinger. Efficient cross-domain learning of complex skills. In *Proceedings of the Eleventh International Conference on Intelligent Tutoring Systems*, pages 493–498, Berlin, 2012. Springer-Verlag.

[15] N. Li, N. Matsuda, W. W. Cohen, and K. R. Koedinger. A machine learning approach for automatic student model discovery. In *EDM*, pages 31–40, 2011.

[16] P. I. Pavlik, H. Cen, and K. R. Koedinger. Learning Factors Transfer Analysis: Using Learning Curve Analysis to Automatically Generate Domain Models. In *Proceedings of 2nd International Conference on Educational Data Mining*, pages 121–130, 2009.

[17] T. N. Pentti Hietala. The competence of learning companion agents. *International Journal of Artificial Intelligence in Education*, 9:178–âĂŞ192, 1998.

[18] U. Schmid and E. Kitzelmann. Inductive rule learning on the knowledge level. *Cognitive System Research*, 12(3-4):237–248, Sept. 2011.

[19] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 1997.

[20] D. H. Sleeman and M. J. Smith. Modeling students' problem solving. *Artificial Intelligence*, 16:171–187, 1981.

[21] R. Sun. Cognitive social simulation incorporating cognitive architectures. *IEEE Intelligent Systems*, 22(5):33–39, Sept. 2007.

[22] N. A. Taatgen and F. J. Lee. Production compilation: A simple mechanism to model complex skill acquisition. *Human Factors*, 45(1):61–75, 2003.

[23] K. K. Tatsuoka. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement*, pages 345–354, 1983.

[24] J. B. Tenenbaum and T. L. Griffiths. Generalization, similarity, and bayesian inference. *Behavioral and Brain Sciences*, 24:629–640, 2001.

[25] K. VanLehn. *Mind Bugs: The Origins of Procedural Misconceptions*. MIT Press, Cambridge, MA, USA, 1990.

[26] K. Vanlehn, S. Ohlsson, and R. Nason. Applications of simulated students: an exploration. *Journal of Artificial Intelligence in Education*, 5:135–175, February 1994.

[27] M. Villano. Probabilistic student models: Bayesian belief networks and knowledge space theory. In *Proceedings of the 2nd International Conference on Intelligent Tutoring Systems*, pages 491–498, Heidelberg, 1992.

Predicting Player Moves in an Educational Game: A Hybrid Approach

Yun-En Liu[1], Travis Mandel[1], Eric Butler[1], Erik Andersen[1],
Eleanor O'Rourke[1], Emma Brunskill[2], and Zoran Popović[1]
[1]Center for Game Science, Computer Science & Engineering, University of Washington
[2]School of Computer Science, Carnegie Mellon University
{yunliu, tmandel, edbutler, eland, eorourke, zoran}@cs.washington.edu, ebrun@cs.cmu.edu

ABSTRACT

Open-ended educational tools can encourage creativity and active engagement, and may be used beyond the classroom. Being able to model and predict learner performance in such tools is a critical component to assist the student, and enable tool refinement. However, open-ended educational domains typically allow an extremely broad range of learner input. As such, building the same kind of cognitive models often used to track and predict student behavior in existing systems is challenging. In addition, the resulting large spaces of user input coupled with comparatively sparse observed data, limits the applicability of straightforward classification methods. We address these difficulties with a new algorithm that combines Markov models, state aggregation, and player heuristic search, dynamically selecting between these methods based on the amount of available data. Applied to a popular educational game, our hybrid model achieved greater predictive accuracy than any of the methods alone, and performed significantly better than a random baseline. We demonstrate how our model can learn player heuristics on data from one task that accurately predict performance on future tasks, and explain how our model retains parameters that are interpretable to non-expert users.

Categories and Subject Descriptors

K.8.0 [**Personal Computing**]: General – Games; H.5.0 [**Information interfaces and presentation**]: General

Keywords

Educational games, user modeling

1. INTRODUCTION

Open-ended learning environments offer promises of increased engagement, deep learning, transfer of skills to new tasks, and opportunities for instructors to observe the learning process. One example of such environments is educational games, where players have an opportunity to explore and experiment with a particular educational domain [12]. However, many of these exciting potential applications require low-level behavioral models of how players behave. For example, if we can predict that a player will struggle with a particular concept, we could try to preempt this confusion with tutorials or choose specific levels designed to address those problems. Additionally, as forcing players to complete an explicit knowledge test often breaks the game flow and causes many players to quit, we could estimate a player's knowledge of target concepts by predicting performance on test levels that are carefully designed to measure understanding of those concepts. Finally, we might even be able to compare user populations by examining models learned from their data and hypothesize optimal learning pathways for each population.

Accurate predictions of user behavior have been achieved in existing educational software such as intelligent tutors [10, 9, 11]. However, we cannot directly apply such methods to educational games for two reasons. First, educational games often have very large state and action spaces. For instance, a game involving building one of 10 different structures on 100 locations has a state space of size 10^{100}. Second, games often increase engagement through the addition of game mechanics that are not directly linked to the main educational objectives. One option is to use expert insight to define skills and behavior associated with these skills for the educational game. However, doing so can be extremely labor intensive: for intelligent tutors for structured domains that often include activities labeled with skills, it has been estimated that 200-300 hours of expert development are necessary to produce one hour of content for intelligent tutors [4]. As educational games are more open-ended, allowing students to input a much wider variety of input compared to many popular intelligent tutoring systems, we expect that tagging and building structure models for them would be even more time consuming than for structured topics such as Algebra.

Given these limitations, we would like a method requiring minimal expert authoring, capable of inferring likely user behavior based on collected data. One popular approach with these properties from the field of recommendation systems is collaborative filtering [18, 21]. Collaborative filtering can be effective with no expert authoring at all if there is enough data; however, the large state space of many educational games often results in high degrees of data sparsity. To maintain accuracy in spite of such sparsity, there has been an emergence of hybrid models that supplement collaborative filtering with limited context-specific information when there is not enough data [16, 24]. Though we are inspired by this work, such methods are not applicable to educational games: we cannot ask users for ranked preferences and are restricted to using behavioral models only, making our task significantly more difficult.

To address these challenges, we create a new ensemble algorithm that leverages the various strengths of multiple disparate models for predicting player behavior. We propose a tripartite methodology that combines elements of collaborative filtering with state-space clustering and modeling players as parameterized heuristic searchers. Using all three methods, we are able to achieve better performance than using any one of these approaches individually. The model reduces the log-likelihood to 68% of a random baseline, outperforming any of its components, which achieve between 73% and 80% log-likelihood of random. Because it uses both a mix of data-driven and model-based approaches, we are able to predict how people will react to any situation in the game, a capability that continues to improve as we observe more players. The model also retains interpretable parameters which we demonstrate by discovering differences in behavior between populations from different websites. Finally, we show that unlike pure collaborative filtering approaches, we can train our model on data from one level and use it to accurately predict behavior on future levels. This allows us to predict how players will respond in situations where we have no data at all, opening up a host of new applications such as adaptive level ordering or invisible assessment based on prediction of player performance on test levels.

2. RELATED WORK

2.1 Educational Technology

There has been substantial research on predicting student outcomes on tests. Some of these methods are based on dynamic assessment, an alternative testing paradigm in which the student receives assistance while working on problems [8, 13]. Intelligent Tutoring Systems (ITSs) include built-in scaffolding and hinting systems, and are therefore an ideal platform for studying dynamic assessment [10]. Studies have shown that this data has strong predictive power. Feng et al. show that 40 minutes of dynamic assessment in the ASSISTment system is more predictive of grades on an end-of-year standaradized test than the same amount of static assessment [9]. Feng et al. also showed that longitudinal dynamic assessment data is more effective at predicting standardized test scores for middle school students than short-term dynamic assessment data [10]. Fuchs et al. showed that dynamic assessment data from third-grade students was useful for predicting scores on far-transfer problem-solving questions [11]. These methods are useful for predicting student outcomes on tests. However, we require much finer granularity for applications such as predicting how students will respond to new levels without any training data or offering just-in-time hints only when we predict the player is about to make a particular type of move.

2.2 Collaborative Filtering

Machine learning classification is often used to predict user behavior. However, many standard classification techniques are ill-suited for the educational game domain, due to the enormous set of possible inputs (classes) from which a player can choose. We also require a way to predict a player may make a new move that is possible, but has not been done by any previous player.

Another promising approach for predicting user behavior is collaborative filtering. It relies on the assumption that if two users have a similar state, and one user behaves in a particular way in response to a new situation, the other user will likely show the same behavior. A good survey of collaborative filtering approaches can be found in [21]. Several researchers have used these methods in the educational data mining domain, including using matrix or tensor factorization models to predict student item responses [7], or student performance on problems [22]. Unfortunately, their methods do not easily transfer to our problem, which involves predicting choices of transitions between game states instead of performance on problems given out one at a time; our data is simply much more sparse.

Of course, data sparsity is known to offer a key challenge to collaborative filtering, making it unable to issue accurate positions given very limited data [21]. Data sparsity is particularly problematic in our domain, an educational puzzle game with a large search space, because users diverge very quickly and most states are rarely visited. One way of alleviating data sparsity is to combine collaborative filtering with external information. Content-based approaches, which consider users or items to be similar if they share similarity in respect to features selected by a designer [19], can be used to augment collaborative filtering in situations where there is not enough data. For example, Melville et al. [16] use collaborative filtering in situations where there is enough data and look for similarities in content in cases where data is lacking. Ziegler et al. [24] propose a different model in which items are categorized into a taxonomy, and recommendations are made not only through user ratings but also categories of demonstrated interest. These methods do not directly apply in our domain, where we have only behavioral data and must predict transitions between states instead of rankings of items. However, we are inspired by their general ideas; more specifically, our method allows the designer to specify similarity functions to combat data sparsity and takes advantage of our domain structure by modeling players as heuristically guided probabilistic searchers.

2.3 Modeling Players in Interactive Games

Game reseachers have tried to estimate player preferences, skills, and behaviors based on in-game activities [6, 20]. Many of these approaches rely on expert-authored player models, although some have used data-driven techniques. For example, Pedersen et al. tried to predict the player's emotional state in Super Mario Bros by training a neural network on features such as number of deaths [17]. Weber et al. modeled player retention in a sports game with regressions to rank expert-chosen features such as playing style [23]. Our method differs by modeling low-level actions directly, a significantly more complicated task on which standard classification techniques are difficult to apply.

Some work has tried to predict low-level actions. Albrecht et al. used Dynamic Belief Networks to predict players' next actions, next locations, and current goals in an adventure game [3]. This work only applies to games with a very specific structure involving locations and quests, which we lack. Our work is probably closest to that of Jansen et al [15], who predicted moves of chess grandmasters by modeling them as low-depth heuristic searchers. Unfortunately, this method alone is not very accurate, and as we show later does not tend to improve as we collect more data. Our ensemble method relies on collected data wherever possible and models players as heuristic searchers only as a last resort, giving us significantly better predictive power.

3. GAME DESCRIPTION

We will first describe the game used in our analysis. Refraction is a educational fractions game that involves splitting lasers into fractional amounts. The player interacts with a grid that contains laser sources, target spaceships, and asteroids, as shown in Figure 1. The goal is to satisfy the target spaceships and avoid asteroids by placing pieces on the grid. Some pieces change the laser direction and others split the laser into two or three equal parts. To win, the player must correctly satisfy all targets at the same time, a task that requires both spatial and mathematical problem solving skills. Some levels contain coins, optional rewards that can be collected by satisfying all target spaceships while a laser of the correct value passes through the coin.

At any point in a level, the player may pick up a piece on the board or drop a piece currently not in play onto the board on any location. Let b be the number of open board locations (about 100), and p the number of available pieces (usually at least 6). Then the size of the state space is approximately b permute p, the number of permutations of open board locations for the pieces, and has a branching factor of about bp. Thus the overwhelming majority of game states and transitions have never been observed, a situation common in open-ended educational environments.

In the analysis that follows, we primarily use player data gathered from level 8 of Refraction, the first non-tutorial level. This level was chosen because it was the non-tutorial level for which we had the most data. The layout of the level can be seen in Figure 1.

4. PREDICTIVE TASK

Our objective is to predict player behavior in the educational game Refraction, similar to how student models in intelligent tutoring systems can be used to predict student input. We now define some of the notation we use in the rest of the paper. For a given level, our task is the following. Let $\mathbf{S}$ be the set of all possible *game states* on the level. A game state is a particular configuration of pieces on the board, independent of time. Each player i in a set of players $\mathbf{P}$ of a level goes through a series of game states. We are concerned with predicting the next substantive move class the player will try, so we preprocess the data to eliminate consecutive duplicate states, leaving us with the list of player's states, $S_{i,1}, \dots, S_{i,m_i}$. We define a set of collapsed states, $\mathbf{C}$, and a *collapse* function mapping $\mathbf{S} \rightarrow \mathbf{C}$. These are selected by the designer to reduce states to features of interest, as in Table 1. For $s \in \mathbf{S}$, define $succ(s)$ to be the set of collapsed states reachable in one action from s, i.e., $succ(s) = \{collapse(s') \mid s'$ is reachable in one move from $s\}$. The predictive model M assigns a probability that the player will enter a collapsed state depending on his history. Given player i's sequence of states up to time $j \leq m_i$, $S_{i,1}, \dots, S_{j-1}$, we want to predict the probability of them entering a collapsed state at time j, $Pr(collapse(S_{i,j}) \mid S_{i,1}, \dots, S_{i,j-1})$, where $\sum_{c \in succ(S_{i,j-1})} Pr(c \mid S_{i,1}, ..., S_{i,j-1}) = 1$. The total probability of the player's sequence of states, P_i, under the model is then $Pr(P_i \mid M) = \prod_{j=1}^{m_i} Pr(collapse(S_{i,j}) \mid S_{i,1}, \dots, S_{i,j-1})$. The total probability of the set of players' traces P is $Pr(P \mid M) = \prod_{i \in \mathbf{P}} Pr(P_i \mid M)$.

The choice of *collapse* is left up to the designer and depends on the intended application. Prediction of player search behavior in a game with maze-like elements, for example, may only require the model to predict where the player will move to next. Or, a system designed to give mathematical hints might only require a model capable of predicting the value of the next fraction the player will produce. In Refraction, we are primarily concerned with how the player will use pieces and manipulate the laser. This gives us good, though incomplete, overall information on their playing ability, and is described in Table 1.

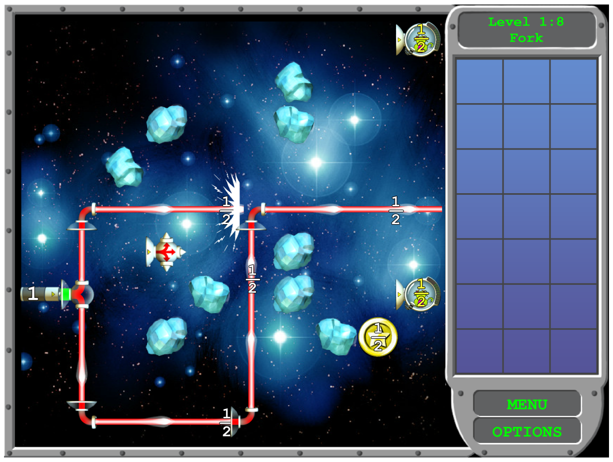


Figure 1: A game state from level 8 of Refraction, on which we will perform most of our analysis. The pieces are used to split lasers into fractional amounts and redirect them to satisfy the target spaceships. All ships must be satisfied at the same time to win.

Feature	Value
Fringe lasers	2 1/2,East
Pieces used	1 W-NS, 2 S-E, 1 N-E, 1 W-N
Ship values satisfied	(none)
Pieces blocking lasers	Benders: 1 S-E
% coins satisfied	0.0

Table 1: The *collapse* function we use in Refraction, applied to the state in Figure 1. States that share all feature values are considered the same. Pieces are listed as (input)-(outputs) in cardinal directions, such that W-NS is a splitter with a westward input and north-south outputs. Fringe lasers are those at the edge of the laser graph either entering the wrong kind of ship or not entering a piece at all.

5. METRICS

In this section we explain how we will evaluate the performance of our predictive model. Our aim is to build models that accurately reflect how populations behave in Refraction levels. We use a similar evaluation metric as [15] and [5] by measuring the information content, in bits, of the population under our predictive model. This number is easily calculated as: $I(P \mid M) = \log_2 Pr(P \mid M)$. We then compare the information content of the data under our model as compared to the information content of the data under a random model, $Compression(P \mid M) = \frac{I(P|M)}{I(P|Random)}$. In general, the goal is to find M maximizing $Pr(P \mid M)$, which corresponds to minimizing $Compression(P \mid M)$. We choose this metric as it offers some interpretability, with 0 indicating perfect ability to predict every move and 1 indicating no compression. This metric also retains the same meaning and scale regardless of the number of players. Unless otherwise stated, all evaluations of goodness of fit are done with 5-fold cross-validation on 1000 players, drawn at random from players from the website Kongregate from a period of August 2011 to September 2012.

6. HYBRID BEHAVIOR PREDICTOR

Here, we describe the three portions of our hybrid predictive model and describe the conditions under which each is used. Each individual method has different benefits and drawbacks and is suitable at a different level of data. We use a combination of them to keep all their benefits, giving us good predictive power, interpretability, and generalizability. At the end, we describe the full model in detail.

6.1 Markov

Collaborative filtering models, which search for similar players and use their data to predict the behavior of new players, are an attractive approach for our problem space because they are data-driven and model-free. There are a number of methods for determining the similarity of two players. We describe and compare two methods: a simple Markov model with no knowledge of player history and a model with full awareness of player history.

In the simple Markov model, we compute the probability of a state transition based only on the player's current state. To estimate these state transitions, we use our prior data, aggregating together any examples which start in the same initial state. To prevent the probability of a player from going to 0 when they make a transition that we have not seen before, we add a smoothing parameter r. With r probability, the player will choose between the possible successor states $succ(S_{i,j-1})$ randomly, and with the remaining $1 - r$ probability, the player will move according to the Markov model as outlined above. We empirically determine that the best performance is achieved with $r = 0.3$.

One weakness of this similarity metric is that it ignores player history. We also attempted other collaborative filtering models. For example, we could consider using only the transitions from other players with the same full history of moves on that level when issuing predictions, reverting to Markov if data is sparse. In the limit of infinite data, we would expect this model to outperform all others based only on order of visits to game states.

We found, however, that the performance of the second history-aware model is worse than the performance of the simple Markov model. The comparison is shown in Figure 2. The underlying issue is that for most observed paths, no previously observed player has followed the exact same path of results. The history-aware model can perform no better than random in these cases. After experimenting with several different collaborative filtering-based models, we settled on the pure Markov model as the most straightforward and accurate approach, achieving a base compression of 0.756.

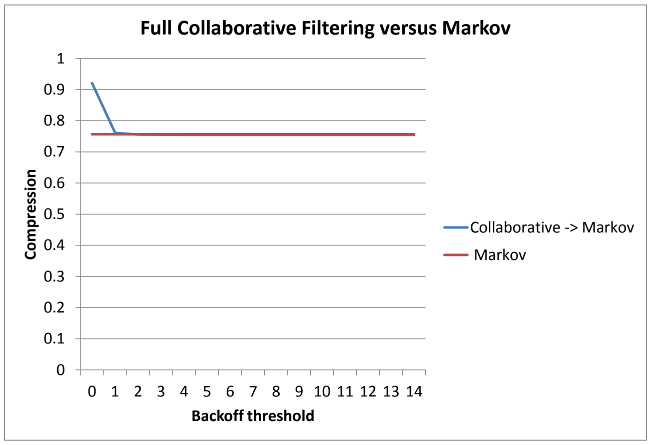


Figure 2: Effect of collaborative filtering models where players are non-Markovian. We create a hybrid model where we first attempt to find similar players under exact path matching, and if there are fewer than n of them, we consult a simple Markov model instead. The Markov model is the same or better at predicting player moves.

6.2 State Aggregation

In the limit of infinite data, we would expect the Markov model to outperform all other methods that make the same assumption that players are history-free. However, the amount of data required for good performance can be quite high. In our case, this problem is compounded by the fact that puzzle game state spaces are difficult to search by design. As a result, most states are visited infrequently, as is shown in Figure 3. It is challenging to predict how new players will behave when they reach these uncommonly-visited states.

One way to address this data sparsity problem is to aggregate data from states that are similar, and use this aggregated data to make predictions. This requires only a minor modification to the original Markov model: instead of looking at the distribution of transitions from the player's current state, we look at the distribution of transitions from all states similar to the player's current state. Here, we use *collapse* to find similar states, though the designer could substitute other functions if desired. To determine when to use this state aggregation approach, we introduce a backoff parameter b_a. When a state has been visited by fewer than b_a players, we switch from the Markov model to the aggregation model.

The aggregated states are not exactly the same as the current state because the *collapse* function throws away some amount of useful data. Thus, we would expect this approach

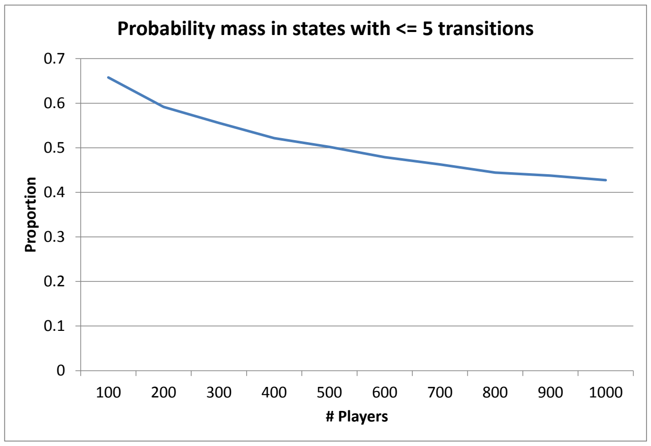


Figure 3: Many moves occur in locations where we have little data, even with 1000 players.

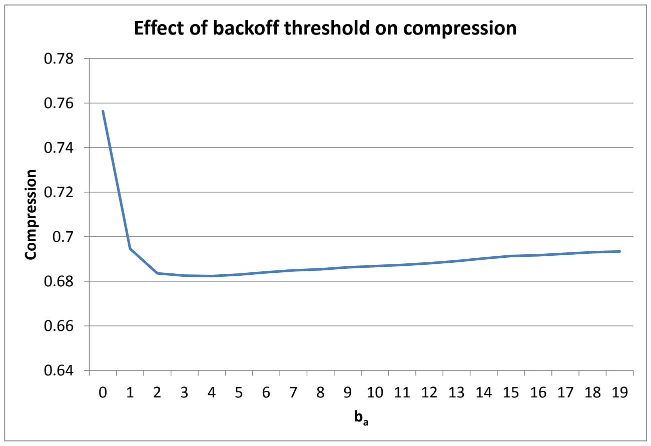


Figure 4: Selection of b_a, the backoff parameter controlling when to consult transitions from all similar states instead of exact states.

to be most beneficial when data is extremely sparse, and become progressively less beneficial as we gather more data. This is exactly the effect we observe, as seen in Figure 4. In general, the best value of b_a depends on the similarity of player's traces through the state space. For Refraction, the optimal value is $b_a = 4$. The overall compression drops from 0.756 to 0.682 when using an approach that combines state aggregation and the Markov model, compared to using the Markov model alone. Applying state aggregation blindly in all cases erases most of the improvement and results in a compression of 0.732, so it is important to tune this parameter correctly.

6.3 Player Heuristic Search

Both of the previously described models have certain advantages. They both require minimal designer input: state space and transition functions for the Markov model, and a similarity function for the aggregation model. Both models also only improve as more data is gathered. Unfortunately, these methods also have two significant drawbacks: they perform poorly when there is very little data available, and they have parameters that are difficult to interpret. An educator trying to determine whether a game level teaches a particular math strategy, for example, would have difficulty learning this from the transition probabilities of a graph with tens of thousands of states.

In order to address these shortcomings, we use a method that models how players explore the game space in cases where data is particularly sparse. We assume that players are heuristically-guided probabilistic searchers. This assumption is reasonable given that players are attempting to solve fraction puzzles which are fun precisely because the solution is not obvious. This allows us to utilize information from every game state and generalize that information to new states. In comparison, the Markov with state aggregation approach can only utilize data from similar states. We expect this heuristic search approach to be most effective when data is scarce. Since the search model is only an approximation of player behavior, this method will become worse relative to the Markov with state aggregation approach as data become more plentiful, since the Markov approach has the power to precisely fit player behavior with enough data.

We provide a brief summary of the player heuristic search algorithm here, but for a full formalization of the algorithm please refer to Jansen et al. [15]. Note that we make a few modifications to the Jansen algorithm, described below. Our search algorithm assumes that users select moves by following a heuristic function v, which determines the likelihood that a player will visit a particular collapsed state. The function v is a weighted linear sum of simple designer-specified functions $a_1, \ldots, a_n$ that operate on collapsed states $c \in \mathbf{C}$: $v(c) = \sum_{k=1}^{n} \lambda_k a_k(c)$. Players, when they make a move, apply the heuristic to each possible collapsed successor in $c \in succ(S_{i,j-1})$ and assign it probability mass $e^{v(c)}$ to prevent negative probabilities, given by (1).

$$Pr(collapse(S_{i,j}) \mid S_{i,j-1}, \lambda_1, \ldots, \lambda_n) = \frac{e^{v(collapse(S_{i,j}))}}{\sum_i e^{v(C_i)}} \tag{1}$$

We optimize the weights λ_k to maximize the log-likelihood of the data using Covariance Matrix Adaptation: Evolutionary Strategy, an optimizer designed to run on noisy functions with difficult-to-compute gradients [14]. Our algorithm differs from the original in that both the possible successor states and the state that the heuristic operates on are collapsed states, since we want to predict the general type of move players will make rather than their exact move. As before, we also introduce a backoff parameter b_h. When searching for transitions from aggregated players, if there are fewer than b_h datapoints, we switch from the Markov with aggregation model to the heuristic model. Empirically we discover that the optimal value is achieved at $b_h = 4$.

The base heuristics $a_1, \ldots, a_n$ are designer-specified, and should reflect the components of the game that players pay attention to while choosing between moves. The heuristics we use for Refraction are listed in Table 2. In practice, the selection of heuristics is closely related to the definition of the *collapse* function used to project raw game states in the prediction task, since both are chosen according to the game features that the designer views as important.

Table 2: Basic heuristics and values in Figure 1

Heuristic	Value in above state
% ships satisfied	0
% ships matching fringe laser values	1
% pieces used	5/6
% pieces blocking lasers	1/6
% coins satisfied	0

The model of player heuristic search allows us to predict subsequent moves even when a player is visiting a state that has never been seen before. Furthermore, the weights λ_k that are optimized in this model are interpretable; they tell us how the population of players values each of the game features defined in the heuristics a. This information can help designers learn about their games, as described in Section 8.

6.4 Full Hybrid Model

Tying all the components together, we now provide a description of the full hybrid prediction model for Refraction, which combines the simple Markov model, state aggregation, and player heuristic search.

1. Assume a player is currently in state s_a.
2. Consult the Markov model for all other players' transitions to collapsed states from state s_a. If there are b_a or more transitions, predict the observed distribution with the random action smoothing parameter of r.
3. Otherwise, apply the state aggregation function to all the nodes in the graph, and count all transitions from all states with collapsed value $collapse(s_a)$. If there are b_h or more transitions, take the observed distribution, remove any states impossible to reach from s_a, and predict the resulting distribution smoothed by r.
4. Otherwise, apply the heuristic with parameters learned from the training set to each of the successors using Equation (1) to get the probability of each transition.

7. EXPERIMENTS

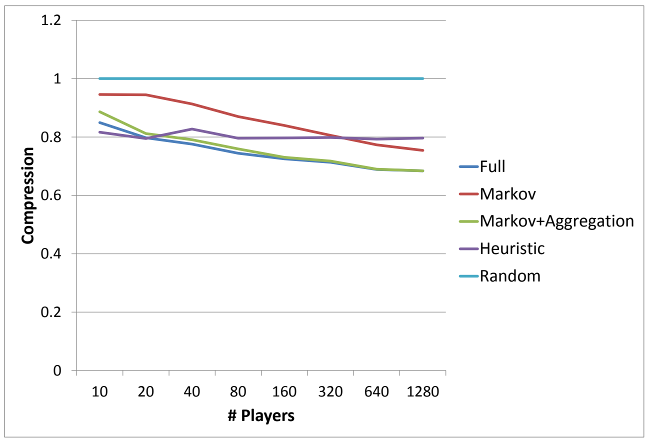
We now evaluate the performance of our predictive model. The performance of the full backoff model, from Markov to state aggregation to player heuristic search depending on the available data, is shown in Figure 5(a). Some features of the graph are worth noting.

- The full model is superior to any of the individual models at nearly all tested levels of data, with the Markov with state aggregation a close second.
- The data-driven approaches continuously improve as more data as added.
- The heuristic function is superior at the start, but its performance does not improve very much as more players are added. This is almost certainly because the model makes very strong assumptions about how players behave that allow it to take advantage of extremely sparse data; however, because the model is not completely accurate, it contains inherent bias that no amount of data will remove.
- The gap between Markov, Markov with state aggregation, and the full backoff model narrow as the amount of data increases. As we gather more players, the amount of probability mass on players in uncommonly visited states shrinks, so the Markov model is used to predict player behavior in more and more situations.

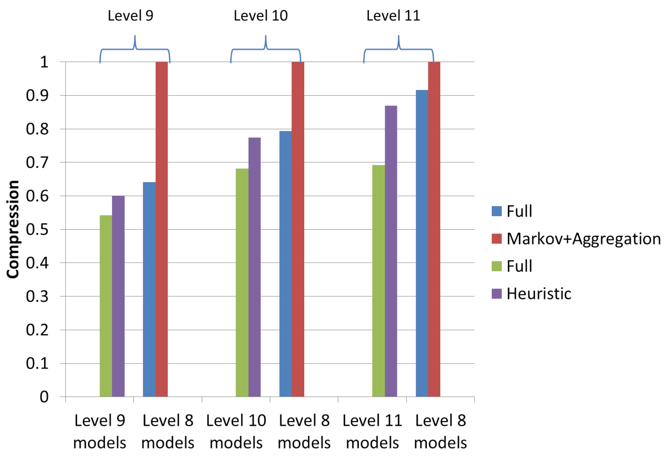
While the heuristic portion of the model seems to offer only incremental improvements, its true power can be seen when we attempt to generalize our models to future levels, as shown in Figure 5(b). Using 1000 players, we first learn heuristic parameters from level 8. We then use the learned heuristic to predict player behavior on levels 9, 10, and 11, comparing these to a Markov with state aggregation model trained on level 8. To get a sense of what "good" performance might look like, we also train player search heuristics and full models learned on the transfer levels and evaluate their compression values with 5-fold cross-validation as usual. We note some features of the graph here.

- We see immediately that the Markov with aggregation portion of the model has no generalization power at all. The state space and possible transitions via *succ* are completely different on future levels, so it's impossible to find similar players and use their transitions to predict moves later on.
- The heuristic portion of the model, on the other hand, allows it to predict what players will do in future levels. When compared to full models fit directly to player data from those levels, it is very good at predicting behavior on level 9, somewhat good at predicting behavior on level 10, and not very good at predicting behavior on level 11. Educational games are explicitly designed to teach players new and better strategies as they play, so we would expect performance to decrease over time.
- In addition, we can see that by level 11 even a heuristic trained on player data from that level is losing power. This means that the features of the state space players pay attention to is no longer being captured by the component heuristic functions $a_1, \ldots, a_n$. As the game introduces new concepts such as compound fractions, equivalent fractions, and fraction addition, players will need to pay attention to more features of the state space than are represented in our choice of $a_1, \ldots, a_n$. This speaks to the importance of choosing these component heuristics for the method's performance.

We caution that the generalization power of our model in these open-ended learning domains can only reasonably be expected to be high for the next few tasks and will be poor if those tasks have radically different state spaces from the training tasks. These caveats notwithstanding, these are promising results that suggest the learned heuristic captures something fundamental about how players navigate the search space of a Refraction level. This might allow designers to guess how players at a certain point in time will behave on levels without needing to release updated versions of the game, or allow educators to simulate and evaluate user performance on assessment levels without needing to interrupt player engagement by actually giving these tasks.



(a) Performance



(b) Generalization

Figure 5: Performance and generalization power of our model. The full model is superior to the other models at most amounts of data. In addition, the full model learned from level 8 data is able to generalize to predict behavior in future levels due to the heuristic component.

Table 3: Heuristic parameters learned on different player populations

Population	% ships	% ships matching	% correct pieces	% incorrect pieces	% coins	Compression
Kongregate	2.529	0.116	0.317	-13.391	0.937	0.782
BrainPOP	1.868	-0.149	1.584	-8.527	0.665	0.862

8. INTERPRETABILITY

One of the key drawbacks of model-free methods are that their results are extremely difficult to interpret, even for experts. The Markov model with state aggregation suffers from this problem, as it is essentially a black box that predicts distributions of transitions. The learned heuristic parameters, on the other hand, offer some glimpses into player behavior. We demonstrate this capability by analyzing some ways in which players differ between two populations. The first is Kongregate, a website whose main demographic is age 18-24 males [2], whose data we have been using up until this point. The second is BrainPOP, a website aimed at schools whose main demographics are children and middle-aged women (who are likely teachers) [1]. We learned heuristic weights on 1000 randomly selected players from each population, shown in Table 3. The goal is not to study how different populations interact with educational puzzle games, so we will not dwell on these results; we simply want to show how these parameters can lead to interesting hypotheses about player behavior.

Two interesting differences are immediately apparent based on these parameters. First, Kongregate players have a stronger negative weight on states with incorrectly used pieces as compared to BrainPOP players, suggesting they are less prone to placing pieces incorrectly. Second, BrainPOP players seem more interested in merely placing pieces down given the relatively high weight on used pieces parameter. Given that they also compress more poorly, one possible explanation is that they have less coherent plans and so place pieces more randomly. These hypotheses cannot be verified merely by looking at the heuristic values, but are sufficiently concrete that we can now run statistical tests to check their validity. For the following analyses, we use the non-parametric Wilcoxon rank-sums test due to the non-normality of our data. As we perform two tests on a dataset after learning parameters from that same dataset, we use the Bonferroni correction to avoid false positives; thus the threshold significance value is set at $\alpha = 0.025$. We report effect sizes as r values, with 0.1 considered small, 0.3 medium, and 0.6 large.

To see if Kongregate players understand piece directionality better than BrainPOP players, we assign to each player the proportion of piece placements such that a laser hits the dropped piece from an incorrect side. We discard players who place no pieces. We find a statistically significant effect of Population on Proportion Drops Incorrect (W=728148, Z=-18.06, r=0.4, p <0.0001), with Kongregate players having a median 0% incorrect drops (N=973) and BrainPOP players having a median of 12% incorrect drops (N=969).

Next, to see if BrainPOP players act with less foresight, we ask how often players make a move, only to take it back immediately. More precisely, for a player who traverses states $s_a, s_b, s_c, s_b, s_c, s_a$, we look at all the triples of moves: (s_a, s_b, s_c), (s_b, s_c, s_b), (s_c, s_b, s_c), and (s_b, s_c, s_a). We then assign to this player the proportion of triples in which the first and third state are the same, discarding players who visit fewer than three states. We find a statistically significant effect of Population on Proportion Takebacks (W=651589, Z=-23.35, r=0.53, $p < 0.0001$), with Kongregate players having a median of 13% takeback moves (N=968) and BrainPOP players having a median of 32% takeback moves (N=971).

These analyses show that the learned parameters in our hybrid model can be valuable tools for game designers, educators, and researchers for analyzing how populations use their systems. For instance, because Kongregate players are primarily adults and BrainPOP players are primarily children, we might wonder if children have more difficulty understanding piece directionality and spatial reasoning and plan their moves less carefully than adults do. A researcher might attempt to generalize these results to other strategic tasks, while a game designer might create a different version of Refraction with easier levels, fewer pieces, and clearer graphics for children. Either way, the learned parameters are a useful tool to help understand how players behave.

9. CONCLUSION

Predicting player behavior in open-ended learning environments is an interesting and complex problem. This ability could be used for a host of automatic applications to bolster engagement, learning, or transfer. In this paper, by using a combination of data-driven and model-based approaches, we presented a "best-of-all-worlds" model able to predict player behavior in an educational game. First, our hybrid model's performance is better than any individual component's. Second, the learned weights of the sub-heuristics are human-readable and can give insights into how players behave. We used these parameters to formulate hypotheses about how two populations behave differently and confirmed them with strong statistical results. Finally, we demonstrated how the heuristic portion of the model allows us to generalize and predict how players will behave on levels in which we have no data at all, opening the door to many adaptive applications involving problem ordering and choice.

There are many possible avenues for future work. On a lower level, we could use more powerful collaborative filtering models taking advantage of timestamps in order to find similar players. Automatic generation of state aggregation functions and autotuning the b_a and b_h parameters would remove the need for some expert authoring. On a higher level, trying the same method on other open-ended educational environments, not necessarily games, could tell us how well the method generalizes. Using the model for applications such as dynamic hinting systems just when we predict players will quit or make egregious errors could increase player engagement and learning. Finally, the ability to estimate behavior on future, unseen problems could be used to increase transfer by selecting tasks which specifically target incorrect strategies or concepts we believe the player has, reflected in the heuristics they use.

10. ACKNOWLEDGMENTS

This work was supported by the University of Washington Center for Game Science, DARPA grant FA8750-11-2-0102, the Bill and Melinda Gates Foundation, Google, NSF grants IIS-0811902 and IIS-1048385, and an NSF Graduate Research Fellowship under Grant No. DGE-0718124.

11. REFERENCES

[1] Alexa Ranking, BrainPOP.
[2] Alexa Ranking, Kongregate.
[3] D. W. Albrecht, I. Zukerman, and A. E. Nicholson. Bayesian models for keyhole plan recognition in an adventure game. *User Modeling and User-Adapted Interaction*, 8(1-2):5–47, Jan. 1998.
[4] V. Aleven, B. M. McLaren, J. Sewall, and K. R. Koedinger. The cognitive tutor authoring tools (CTAT): preliminary evaluation of efficiency gains. ITS'06, 2006.
[5] I. Althofer. Data compression using an intelligent generator: The storage of chess games as an example. *Artificial Intelligence*, 52(1):109 – 113, 1991.
[6] S. C. Bakkes, P. H. Spronck, and G. van Lankveld. Player behavioural modelling for video games. *Entertainment Computing*, 3(3):71 – 79, 2012.
[7] Y. Bergner, S. Droschler, G. Kortemeyer, S. Rayyan, D. Seaton, and D. Pritchard. Model-based collaborative filtering analysis of student response data: Machine-learning item response theory. In *EDM*, 2012.
[8] J. C. Campione. Assisted assessment: A taxonomy of approaches and an outline of strengths and weaknesses. *Journal of Learning Disabilities*, 22(3), 1989.
[9] M. Feng and N. T. Heffernan. Can we get better assessment from a tutoring system compared to traditional paper testing? can we have our cake (better assessment) and eat it too (student learning during the test)? In *EDM*, 2010.
[10] M. Feng, N. T. Heffernan, and K. R. Koedinger. Predicting state test scores better with intelligent tutoring systems: developing metrics to measure assistance required. ITS'06, pages 31–40, 2006.
[11] L. S. Fuchs, D. L. Compton, D. Fuchs, K. N. Hollenbeck, C. F. Craddock, and C. L. Hamlett. Dynamic assessment of algebraic learning in predicting third graders' development of mathematical problem solving. *Journal of Educational Psychology*, 100(4):829 – 850, 2008.
[12] J. P. Gee. Learning by design: Games as learning machines. *Interactive Educational Multimedia*, 8:15–23, 2004.
[13] E. L. Grigorenko and R. J. Sternberg. Dynamic testing. *Psychological Bulletin*, 124(1):75 – 111, 1998.
[14] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evol. Comput.*, 9(2):159–195, June 2001.
[15] A. R. Jansen, D. L. Dowe, and G. E. Farr. Inductive inference of chess player strategy. PRICAI'00, pages 61–71, Berlin, Heidelberg, 2000. Springer-Verlag.
[16] P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering for improved recommendations. AAAI-02, Menlo Park, CA, USA, 2002.
[17] C. Pedersen, J. Togelius, and G. N. Yannakakis. Modeling player experience in super mario bros. CIG'09, pages 132–139, Piscataway, NJ, USA, 2009. IEEE Press.
[18] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. CSCW '94, 1994.
[19] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen. Collaborative filtering recommender systems, 2007.
[20] A. M. Smith, C. Lewis, K. Hullett, G. Smith, and A. Sullivan. An inclusive taxonomy of player modeling. *University of California, Santa Cruz, Tech. Rep. UCSC-SOE-11-13*, 2011.
[21] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Adv. in Artif. Intell.*, 2009, Jan. 2009.
[22] N. Thai-Nghe, T. Horváth, and L. Schmidt-Thieme. Factorization models for forecasting student performance. In *EDM*, pages 11–20, 2011.
[23] B. G. Weber, M. Mateas, and A. Jhala. Using data mining to model player experience. In *FDG Workshop on Evaluating Player Experience in Games*, Bordeaux, France, 06/2011 2011. ACM, ACM.
[24] C.-N. Ziegler, G. Lausen, and L. Schmidt-Thieme. Taxonomy-driven computation of product recommendations. CIKM '04, 2004.

Sequences of Frustration and Confusion, and Learning

Zhongxiu Liu
Visit Pataranutaporn
Jaclyn Ocumpaugh
Worcester Polytechnic Institute
100 Institute Road
Worcester, MA 01609
{zhongxiuliu,vpatara,jocumpaugh}@wpi.edu

Ryan S.J.d. Baker
Teachers College, Columbia University
525 W. 120th St.
New York, NY 10027
+1 (212) 678-8329
baker2@exchange.tc.columbia.edu

ABSTRACT

In this paper, we use sensor-free affect detection [4] and a discovery with models approach to explore the relationship between affect occurring over varying durations and learning outcomes among students using Cognitive Tutor Algebra. Researchers have suggested that the affective state of confusion can have positive effects on learning as long as students are able to resolve their confusion [10, 22], and recent research seems to accord with this hypothesis [17]. However, there is some room for concern that some of this earlier work may have conflated frustration and confusion. We replicate these analyses using sensor-free automated detectors trained to distinguish these two affective states. Our analyses suggest that the effect may be stronger for frustration than confusion, but is strongest when these two affective states are taken together. Implications for these findings, including the role of confusion and frustration in online learning, are discussed.

Keywords

Affect, confusion, frustration, affect sequences, affect detection, learning outcomes, discovery with models, affective persistence

1. INTRODUCTION

Affect has become an area of considerable interest within research on interactive learning environments [1, 10, 11, 18, 23]. Though findings relating boredom and engaged concentration to learning have largely accorded to prior hypotheses, there have been surprising patterns of results for other affective states, with unstable effects for confusion between studies and often no effects for frustration [7, 21].

However, many of these early studies investigated overall proportions of affective states, rather than considering the potential differential impacts of affect manifesting in different ways. It may be important to consider the multiple ways a specific affective state can manifest, especially considering that there can be considerable variance in how long an affective state lasts [8], affect may be influenced by behavior and vice-versa [3, 5] and some affective states may not be unitary in nature (for example, [12] refers to "pleasurable frustration," which is presumably different in nature than the non-pleasurable frustration often discussed in the research literature).

This puzzle is of particular interest for the affective state referred to as confusion. While relationships between boredom and learning, and engaged concentration and learning, often follow hypothesized patterns [7, 21], confusion appears to manifest in unstable ways across studies. For example, while [7] and [9] find a positive relationship between confusion and learning, with an experimental intervention in the case of [9], [21] finds a negative relationship. Frustration, somewhat surprisingly, routinely does not appear to be correlated with differences in learning outcomes [7, 21].

One possibility is that these results — particularly the results for confusion — may be based on insufficient information. That is, the overall prevalence of an affective state may not accurately predict its impact; how it manifests matters. As [22] notes, students who become confused may either deliberate until they resolve their confusion or become hopelessly "stuck" in unresolved confusion; the former situation has been hypothesized to help learning while the latter undercuts student achievement [22]. As such, the duration of a student's state of confusion may be meaningful. Under this hypothesis, the longer a student remains confused, the less likely they are to resolve that confusion [22]. [10] suggests that confusion may have a dual nature when considered as an affective state: it is possible for it to trigger either persistence (engagement) or resistance to the learning process.

These hypotheses were investigated in Lee et al. [17], who analyzed students' affect over time as the students learned introductory computer programming. Lee and colleagues broke down students' compilation behaviors within this context into sequences of 8 compilations within the learning software, and used text replays [2] to label student behavior in terms of whether the student was thought to be confused. They then developed a data-mined model based on these labels, and distilled its outputs into sequences of two or three consecutive affective states (confused or not confused). They then correlated each student's proportion of these sequences with the student's mid-term exam scores. This test was given after the learning activity studied.

Lee et al. found evidence that short-term confusion that resolves seems to impact learning positively, whereas prolonged confusion affects learning negatively [17]. They found a fairly strong negative relationship between prolonged confusion (three measurements of confusion in a row) and learning (r=-0.337), while students who had brief periods of confusion followed by extended periods where the student was not confused had more positive learning (r=.233).

The results in [17] are intriguing, and show the benefits of this type of fine-grained analysis. However, there are some limitations to this study that may reduce confidence in its findings and therefore call for replication and clarification. (These limitations were pointed out by the anonymous reviewers at the time of submission of [17]). One key potential limitation was that the operational definition of confusion used in [17] differs substantially from that used in prior research on affect and learning [3, 7, 21]. In [17], clips were coded as confused based on extended student difficulty, for example when a student failed to resolve an error on several consecutive programming compilations. It is not clear that these inferences capture confusion in the same sense that is traditionally described in the affect literature. In particular, this behavior and other aspects of the operational definition of confusion in [17] may have incorporated instances of frustration as well as confusion. This potential limitation was due to the approach used to label confusion; the human coders in [17] inferred and hand-labeled affect solely from a fairly limited subset of the information available in log files, as opposed to the field observations or video observations used in other work, each of which leverage more information to discriminate affect. While the text replay method has been shown to be reliable for inferring behaviors [2, 24], its use in affect labeling is relatively more experimental and may be more open to question.

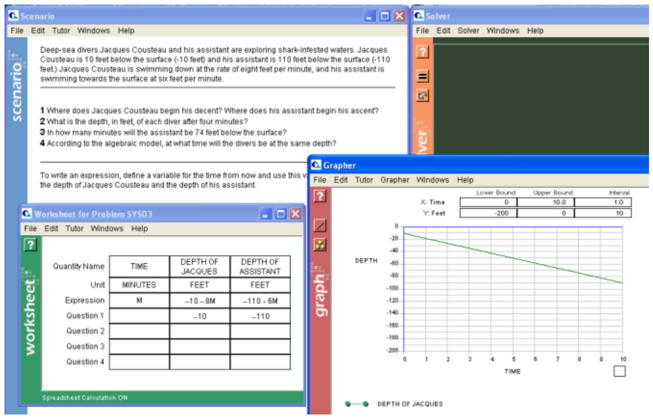
Another limitation in this early work is that the measure of learning used (a mid-term exam) was not grounded in any measure of students' knowledge prior to the learning activity. As such, this work assumes that specific affective patterns led to student success, but it is equally possible that student prior knowledge led both to those affective patterns and to high scores on the mid-term.

In this paper, we build on this work, replicating it but extending it to address these concerns by incorporating models specifically tailored to distinguish confusion and frustration and by adding a pre-test. By doing so, we can better understand the relationship between duration of affect and student learning outcomes. In these analyses, we consider confusion and frustration taken independently, as well as the union of these two affective states (which in our current view may have been what was assessed in [17]).

2. METHODS

2.1 Tutor Studied

The learning system used in this study was Cognitive Tutor Algebra I, an interactive learning environment now used by approximately 500,000 students a year in the USA. The students in this study used a lesson on systems of algebraic equations as part of their regular mathematics curriculum. In Cognitive Tutors, students solve problems with exercises and feedback chosen based on a model of which skills the student possesses. Cognitive Tutor Algebra has been shown to significantly improve student performance on standardized exams and tests of problem-solving skill [14].


Scenario
File Edit Tutor Windows Help
scenario
Solver
File Edit Solver Windows Help
solver
Worksheet for Problem SYS03
File Edit Tutor Windows Help
worksheet
Spreadsheet Calculation ON
Grapher
File Edit Tutor Grapher Windows Help
graph
DEPTH
TIME
DEPTH OF JACQUES

Deep-sea divers Jacques Cousteau and his assistant are exploring shark-infested waters. Jacques Cousteau is 10 feet below the surface (-10 feet) and his assistant is 110 feet below the surface (-110 feet.) Jacques Cousteau is swimming down at the rate of eight feet per minute, and his assistant is swimming towards the surface at six feet per minute.

1 Where does Jacques Cousteau begin his decent? Where does his assistant begin his ascent?
2 What is the depth, in feet, of each diver after four minutes?
3 In how many minutes will the assistant be 74 feet below the surface?
4 According to the algebraic model, at what time will the divers be at the same depth?

To write an expression, define a variable for the time from now and use this v
the depth of Jacques Cousteau and the depth of his assistant.

Quantity Name	TIME	DEPTH OF JACQUES	DEPTH OF ASSISTANT
Unit	MINUTES	FEET	FEET
Expression	M	–10 – 8M	–110 + 6M
Question 1		–10	–110
Question 2			
Question 3			
Question 4			

	Lower Bound	Upper Bound	Interval
X: Time	0	10.0	1.0
Y: Feet	-200	0	10

Figure 1: The Systems of Equations A lesson, from Cognitive Tutor Algebra I, used in this study.

2.2 Data Set

Data were collected from 89 students in rural Western Pennsylvania (the data presented here was also discussed in [4], where affect detectors were presented for this data; these affect detectors are in turn used in this paper, in "discovery with models" analyses). Compared with the state's average, students at this high school had a higher average on the PSSA standardized exam, were less likely to be a member of ethnic minority group, and were less likely to be eligible for free or reduced-price lunch. They were well-balanced for gender.

Each student in this study participated in a learning session using the Systems of Equations A lesson of Cognitive Tutor Algebra, which focuses on learning to graph and solve systems of equations. Each student used the tutor software for two class sessions. Tutor activities were preceded and followed by pre-test and post-test measures of learning. (Four students who did not complete all three of these activities were later excluded.) The average pre-test score was 75.2% (SD = 25.3%), and the average post-test score was 79.8% (SD = 23.5%).

During the learning session, two expert field observers coded students' affect following the protocol outlined in [19]. Within this protocol, holistic observations are conducted based on a combination of facial expression, posture, actions within the software, context, and other factors. Confusion and frustration are distinguished, with a key difference being that frustration involves negatively-valenced affect often combined with expressions of dissatisfaction or anger, whereas confusion is a less negative experience. Though the two states are relatively similar conceptually, typically they have not been challenging for observers to distinguish within this protocol; boredom and confusion have more often been the source of disagreement between coders [19]. Observations are conducted in a pre-determined order, with an approach designed to minimize observer effects and to sample evenly across students during the period of observation, both in terms of number of observations per student, and the time when observations occur.

After field observations were collected, they were synchronized with features distilled from interaction log data, and detectors were constructed and validated for several affect categories, two of which (confusion and frustration) will be used in this study. Complete detail on the automated detectors is given in [4]. In brief, the frustration detector was generated at using the REPTree algorithm, achieving a Kappa of 0.23 and an A' of 0.64, under student-level cross-validation. The confusion detector was produced using JRip, achieving a Kappa of 0.40 and an A' of 0.71, under student-level cross-validation. Note that the values of A' given here are lower than in [4]; these represent the exact same detectors, but the values of A' given in that earlier work were computed using the implementation in RapidMiner 4.6, which was afterwards discovered to be buggy. The values given here are re-computed using the Wilcoxon interpretation of A' rather than the AUC interpretation, using code at http://www.columbia.edu/~rsb2162/computeAPrime.zip.

In the study presented in the current paper, automated detectors were used in order to achieve repeated measurements of a student's affect over relatively brief periods of time, while avoiding observer effects (although the protocol in [19] is designed to be non-intrusive, and to reduce observer effects, continually observing a student over extended periods of time increases the probability that the student will notice that they are being observed and change their behavior). Labels were generated by automated detectors at the level of 20-second intervals of student behavior, termed clips. The grain-size of 20-seconds was selected because this matches the original length of the field observations used to create the detectors. Problem boundaries and other events were not considered when clips were created. While it could be argued that it is better to avoid allowing clips to extend across problem boundaries, affect may extend across these events, and avoiding these transitions may give a less representative picture of overall student affect. A total of 29,777 clips were generated across the students' use of the tutoring software.

Three applications of these detectors are studied. The first application uses only the confusion detector, labeling clips as either confused (C) or not (N), splitting students based on a 50% confidence cut-off. The second application uses only the frustration detector, labeling clips as either frustrated (F) or not (N), also splitting students based on a 50% confidence cut-off. The third applies both detectors simultaneously, and considers a clip as confused/frustrated (referred to as A for "Any" below) if either detector had confidence over 50%. This third application, in our view, may map best to the approach taken in [17].

Once clips were labeled, they were segmented into sequences of three consecutive states. These sequences were chosen to be comparable to the 3-step sequences in [17], but represent a finer level of granularity because of the shorter duration of clips in this work (20 seconds versus 8 compilations, which can take several minutes). Potential sequences for each application are included with their frequencies in Tables 1-3.

Table 1. Possible Sequences for Confusion, with Frequencies (%)

NNN	NNC	NCN	NCC	CNN	CNC	CCN	CCC
93.78	1.91	1.74	0.23	1.84	0.09	0.23	0.16

Table 2. Possible Sequences for Frustration, with Frequencies (%)

NNN	NNF	NFN	NFF	FNN	FNF	FFN	FFF
96.20	1.16	1.09	0.14	1.15	0.08	0.14	0.04

Table 3. Possible Sequences for "Any" (Unified Confusion/Frustration), with Frequencies (%)

NNN	NNA	NAN	NAA	ANN	ANA	AAN	AAA
90.25	2.94	2.70	0.41	2.86	0.20	0.40	0.24

Once detectors were applied, the relative frequency of each sequence was compared to several learning measures, including pretest scores, posttest scores, and the difference between the two. Because the number of tests introduces the potential of spurious effects, the Benjamini & Hochberg (B&H) adjustment [6] is used as a post-hoc control. This method does not guarantee each test's significance, but it does guarantee a low overall proportion of false positives, while preventing the substantial over-conservativism found in methods such as the Bonferroni correction [cf. 20].

In this study, we consider two levels of baseline statistical significance (α=0.05 or 0.1) for the Benjamini & Hochberg adjustment. The 0.05 level reflects full statistical significance, whereas 0.1 reflects marginal significance. Within the B&H adjustment, each test retains its original statistical significance, but the α value cutoff for significance changes depending on the order of the test in significance among the tests run. For understandability, adjusted significance cutoffs are given in tables below for all tests run.

3. RESULTS

3.1 Duration of Affect and Learning Gains

In this section, we compare the relative frequency of sequences of confusion and frustration to assessments of gains in student learning over time. Learning gains are computed as post-pre; the alternate metric of (post-pre)/(1-pre) is difficult to interpret when some students obtain pre-test scores of 100%, which were seen in this data set. In order to understand the importance of individual patterns, we apply separate significance tests for each pattern (with post-hoc controls as discussed below), rather than building a unitary model to predict learning gains from a student's combination set of sequences.

Results for confusion diverged considerably from what might be predicted based on previous research. As shown in Table 4, only three of eight possible sequences showed marginal significance when correlated with confusion, and all of these effects disappeared after post-hoc controls were applied. That is, contrary to theoretical predictions [10, 22], and the interpretation of the findings in [17], differences in sequences of affective state of confusion do not appear to be associated with learning gains in this data.

Table 4. Confusion vs. Learning Gains (No results remain significant after post-hoc control)

3-step - diff	r	p	p cutoff (sig)	p cutoff (marginal)
NNC	0.21	0.054	0.00625	0.0125
CNC	0.198	0.070	0.0125	0.025
NNN	-0.181	0.097	0.01875	0.0375
NCN	0.179	0.101	0.025	0.05
CNN	0.157	0.151	0.03125	0.0625
NCC	0.149	0.173	0.0375	0.075
CCN	0.131	0.231	0.04375	0.0875
CCC	-0.049	0.654	0.05	0.1

By contrast, frustration (Table 5) shows several correlations with learning gains that remain marginally statistically significant after post hoc adjustments. Interestingly, the patterns for frustration match those reported for confusion in [17]. Namely, extended (3-step) periods of no frustration (NNN) are negatively correlated with learning gains. That is, 60 seconds without frustration negatively impacts learning. Introducing one 20-second interval of frustration (as in NFN, NNF, FNN, and FNF) seems to improve learning outcomes (r=0.273, 0.25, 0.248, and 0.208, respectively), but this effect is reduced or eliminated if the sequence contains two intervals of frustration. Only one sequence with two intervals of frustration (FNF) remains marginally significant after post-hoc adjustment, but with a lower effect-size (r=0.208) than those with only one interval of frustration. These results accord with those for confusion in [17].

As such, one possible explanation is that the construct primarily being inferred in [17] was frustration. The findings seen here match well if that assumption is made; they do not match well, if the codes in [17] genuinely reflected the affective state of confusion. We will discuss this possibility further in section 3.3.

Table 5. Frustration vs. Learning Gains (Significant results are in dark gray; marginally significant results are in light gray)

3-step - diff	r	p	p cutoff (sig)	p cutoff (marginal)
NFN	0.273	0.011	0.00625	0.0125
NNN	-0.262	0.016	0.0125	0.025
NNF	0.25	0.021	0.01875	0.0375
FNN	0.248	0.022	0.025	0.05
FNF	0.208	0.056	0.03125	0.0625
FFF	0.174	0.111	0.0375	0.075
NFF	0.136	0.215	0.04375	0.0875
FFN	0.136	0.215	0.05	0.1

3.2 Duration of Affect and Pre-test/Post-test

In the previous section, we saw evidence that brief frustration is associated with positive learning gains, but that lengthier frustration is associated with poor learning gains. In this section, we break down the learning gain measure into its constituent parts, the student's pre-test score and post-test score. Results shown in Tables 6-7 show that pretest scores can predict the frequencies of both confusion and frustration during the learning session. Specifically, lower pretest scores are more likely to co-occur with sequences containing at least one instance of that particular affect (as in CNN, NCN, and NNC when only the confusion detector is applied in Table 6 or in FNN, NFN, or NNF when only the frustration detector is applied in Table 7). Similar effects are found for sequences where two instances of either affect have been detected (as in CCN and NCC, or FFN and NFF). Further, higher pretest scores correlate with higher frequencies of prolonged states of not-confused and not-frustrated (both of which are represented as NNN in Tables 6-7). All the significant r-values in Tables 6-7 remain significant or marginally significant after the post-hoc control.

Table 6. Confusion vs. Pretest Scores (Significant results are in dark gray; marginally significant results are in light gray)

3-step - pre	r	p	p cutoff (sig)	p cutoff (marginal)
NCC	-0.295	0.006	0.00625	0.0125
CCN	-0.283	0.009	0.0125	0.025
NNC	-0.26	0.016	0.01875	0.0375
NNN	0.255	0.018	0.025	0.05
CNN	-0.226	0.037	0.03125	0.0625
NCN	-0.195	0.074	0.0375	0.075
CNC	-0.161	0.141	0.04375	0.0875
CCC	-0.005	0.967	0.05	0.1

Table 7. Frustration vs. Pretest Scores (Significant results are in dark gray; marginally significant results are in light gray)

3-step - pre	r	p	p cutoff (sig)	p cutoff (marginal)
NNN	0.277	0.010	0.00625	0.0125
NNF	-0.273	0.011	0.0125	0.025
FNN	-0.27	0.012	0.01875	0.0375
NFN	-0.267	0.014	0.025	0.05
NFF	-0.231	0.033	0.03125	0.0625
FFN	-0.231	0.033	0.0375	0.075
FNF	-0.125	0.253	0.04375	0.0875
FFF	-0.02	0.854	0.05	0.1

Surprisingly, correlating the affective sequences to post-test scores shows essentially no relationships. As Tables 8-9 show, neither confusion nor frustration sequences are significantly correlated with posttest results. In other words, low pre-test results predict confusion and frustration will occur during the learning session, but presence of these affective states does not predict post-test performance. These results suggest either that the tutor was effective at bringing all students up to mastery, or that there was a ceiling effect in test performance. In other words, students who were confused or frustrated during the learning session because they began with low domain knowledge caught up to students who, because they began with high domain knowledge, experienced little confusion or frustration. However, it is notable that as was found when compared to learning gains and to pre-test results, confusion and frustration have the same pattern for post-test results.

Table 8. Confusion vs. Posttest Scores (No results remain significant after post-hoc control)

3-step - post	r	p	p cutoff (sig)	p cutoff (marginal)
CCN	-0.155	0.157	0.00625	0.0125
NCC	-0.147	0.180	0.0125	0.025
NNN	0.068	0.539	0.01875	0.0375
CNN	-0.064	0.561	0.025	0.05
CCC	-0.061	0.579	0.03125	0.0625
CNC	0.052	0.635	0.0375	0.075
NNC	-0.04	0.716	0.04375	0.0875
NCN	-0.005	0.966	0.05	0.1

Table 9. Frustration vs. Posttest Scores (No results remain significant after post-hoc control)

3-step - post	r	p	p cutoff (sig)	p cutoff (marginal)
FFF	0.177	0.106	0.00625	0.0125
FNF	0.102	0.351	0.0125	0.025
NFF	-0.093	0.396	0.01875	0.0375
FFN	-0.093	0.396	0.025	0.05
NFN	0.025	0.822	0.03125	0.0625
NNF	-0.009	0.937	0.0375	0.075
FNN	-0.008	0.946	0.04375	0.0875
NNN	0	1.000	0.05	0.1

3.3 Unifying Confusion and Frustration

Confusion and frustration have some theoretical similarities, although they are often considered separately in affective research. Both are affective states that occur when a student is struggling with difficult material and has not yet achieved understanding. As discussed earlier, one way to interpret the work in [17] is that their model of confusion may also have included instances of frustration. Hence it may be worth studying these two constructs in a unified fashion – treating them as if they are the same construct during analysis. Also, as discussed in previous sections, the relationships between confusion and learning, and frustration and learning, were qualitatively similar in our data set. They were of different magnitudes (frustration had higher correlations than confusion) but were generally pointing in the same direction. This trend also warrants a joint analysis of the two states.

In order to do so, we applied both detectors (which operate independently) to the data at the same time. Any instance that was labeled as either confused (C) or frustrated (F) in previous sections was now labeled as "any" (A), including the rare instances where a single clip was labeled by the detectors as indicating both confusion and frustration. Instances of A are contrasted with instances where neither (N) affect was detected. Table 10 shows the correlations between learning gains and 3-step any/neither (A/N) sequences.

Table 10. Correlations between 3-step "Any" sequences and Learning Gains. (Significant results are in dark gray; the marginally significant, in light gray.)

3-step - diff	r	p	p cutoff (sig)	p cutoff (marginal)
NNA	0.295	0.006	0.00625	0.0125
NAN	0.284	0.008	0.0125	0.025
NNN	-0.279	0.010	0.01875	0.0375
ANN	0.262	0.015	0.025	0.05
ANA	0.213	0.050	0.03125	0.0625
NAA	0.204	0.061	0.0375	0.075
AAN	0.19	0.081	0.04375	0.0875
AAA	0.01	0.931	0.05	0.1

Several findings from this analysis are similar to the findings presented earlier in this paper, but obtain higher correlations than are seen for confusion or frustration alone. Extended periods of "neither" (i.e., NNN) during the learning session are negatively correlated with learning gains (r = -0.279). All 3-step sequences of short term "any" (i.e., NNA, NAN, and ANN) are found to be positively correlated with learning gains, (r=0.295, 0.284, and 0.262, respectively). Moreover, ANA, NAA, and AAN are found to be positively correlated at a marginally significant level (r=0.213, 0.204, and 0.19, respectively).

Compared with the significant r-values of 3-step frustration and learning gains in Table 5, the r-values for "any" have larger magnitudes, meaning that combining confusion and frustration yields stronger correlations with learning gains than frustration does alone.

4. CONCLUSION AND DISCUSSION

In this paper, we discussed correlations between student test scores and sequences of two affective states—confusion and frustration—during learning with Cognitive Tutor Algebra. These affective states were studied both independently and in combination.

A decade ago, key theoretical models of confusion and frustration during learning and interaction hypothesized that confusion leads to frustration [16] as part of a process where students fail to learn. In line with this theory, researchers suggested that identifying and responding to frustration was essential [13, 15]. However, research looking at overall proportions of student affect (e.g., confusion or frustration) found inconsistent patterns for confusion and null results for frustration (e.g., [7, 21], leading one paper to argue that frustration is significantly less important to learning than other affective states such as boredom [3]).

Research that followed this suggested that the dynamics of affect over time might play an important role in learning outcomes. Confusion that led to frustration, for example, was hypothesized to lead to poorer learning outcomes than confusion that resolved [10, 22].

In this paper, we find a pattern that accords broadly with [17], where confusion and frustration are associated positively with learning for brief episodes and negatively for lengthy episodes. Somewhat contrary to expectations (but consistent with the work in [17]), this effect is strongest if the two affective states are considered together, and weakest if confusion is considered alone (with frustration in the middle). This finding is not inconsistent with the prior literature (differing relations between frustration and learning based on the length of frustration are quite consistent with overall null effects) but does reinterpret it somewhat.

One important limitation to the research presented here is that the length of the affective sequences differs from that found in [17], complicating comparisons between the two. It is known that different affective states often have different durations [8]. However, these durations are likely to be determined by the population and learning context as well. In other words, brief frustration in one context may be lengthy frustration in another. (This possibility may explain the similarity in results between this paper and [17]. Although the time per affective observation was different, the times used in each environment may have matched the general time for a student to make progress in the different environments, as computer programming is a more time-consuming activity than completing highly scaffolded mathematics problems.) Understanding what the "tipping point" is between brief and lengthy confusion or frustration, in different contexts, may be a valuable step for future research.

Overall, this paper's results suggest that attempting to understand overall relationships between affective states and learning is prone to conflating multiple phenomena. Affective states are not unitary; it matters at minimum how long they are, it matters what follows them [23], and probably other factors matter as well (such as culture, for instance). Researchers have also considered the possibility of multiple types of frustration (for instance, [12] speaks of "pleasurable frustration"). Our results show temporal effects for frustration that are highly similar to those hypothesized for confusion, results that deserve more careful consideration in future research. Though a student's overall degree of frustration has often been associated with null effects [e.g., 7, 21], it appears that frustration is associated with differences in learning when considered in a finer-grained fashion. It may be that the conditions that lead to both frustration and confusion (the struggle associated with learning material that is not immediately apparent) are necessary components of the learning process, and both frustration and confusion only become detrimental if a student is unable to reach resolution in an adequate time frame. It is also possible that frustration may be simply an outcome of the cognitive processes underlying these phenomenon, or even just a result of confusion being resolved or not resolved (e.g., different types or intensities or durations of confusion might trigger persistence or resistance, while varying lengths of frustration merely reflect these differences). The similar patterns between confusion and frustration raise questions about whether the best theoretical split is even between confusion and frustration, or whether we should instead move to comparing brief-confrustion, extended-confrustion, and perhaps pleasurable-confrustion (alternate terms for the affective state combining confusion and frustration are welcome). Work to understand and model these affective states in their full complexity will be an essential area of future research. These endeavors will be supported by the advent of data-mined models, such as the ones used here, that can identify affect in a fashion that is both fine-grained and scalable.

5. ACKNOWLEDGMENTS

This research was supported by grant "Toward a Decade of PSLC Research: Investigating Instructional, Social, and Learner Factors in Robust Learning through Data-Driven Analysis and Modeling," National Science Foundation award #SBE-0836012. We would like to thank the anonymous reviewers of [17] for very helpful comments and suggestions.

6. REFERENCES

[1] Arroyo, I., Cooper, D., Burleson, W., Woolf, B. 2010. Bayesian Networks and Linear Regression Models of Students' Goals, Moods, and Emotions. *Handbook of educational data mining* (Oct. 2010). Taylor and Francis Group, London, UK, 323.

[2] Baker, R.S.J.d., Corbett, A.T., Wagner, A.Z. 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 System* (Jhongli, Taiwan, June 26-30, 2006), 29-36.

[3] Baker, R.S.J.d., D'Mello, S., Rodrigo, M., Graesser, A. 2010. Better to be frustrated than bored: The incidence and persistence of affect during interactions with three different computer-based learning environments. *International Journal of Human-computer Studies*, 68, 4 (Dec. 2010). Elsevier B.V., Oxford, UK, 223-241.

[4] Baker, R.S.J.d., Gowda, S.M., Wixon, M., Kalka, J., Wagner, A.Z., Salvi, A., Aleven, V., Kusbit, G., Ocumpaugh, J., Rossi, L. 2012. Sensor-free automated detection of affect in a Cognitive Tutor for Algebra. *Proceedings of the 5th International Conference on Educational Data Mining* (Chania, Greece, June 19-21, 2012), 126-133.

[5] Baker, R.S.J.d., Moore, G., Wagner, A., Kalka, J., Karabinos, M., Ashe, C., Yaron, D. 2011. The Dynamics Between Student Affect and Behavior Occuring Outside of Educational Software. *Proceedings of the 4th bi-annual International Conference on Affective Computing and Intelligent Interaction*. (Memphis, TN, Oct 9-16, 2011).

[6] Benjamini, Y.,Hochberg, Y. 1995. Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing Author(s). *Journal of the Royal Statistical Society,* Series B, 58, 1 (1995), London, UK, 289-300.

[7] Craig, S., Graesser, A., Sullins, J., 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). Taylor & Francis, London, UK, 241-250.

[8] D'Mello, S.K., Graesser, A.C. 2011. The Half-Life of Cognitive-Affective States during Complex Learning. *Cognition and Emotion*, 25, 7 (2011). Taylor and Francis Group, London, UK, 1299-1308.

[9] D'Mello, S.K.., Lehman, B., Pekrun, R., Graesser, A.T. In Press. Confusion Can Be Beneficial For Learning. *Learning and Instruction*. Elsevier B.V., Oxford, UK.

[10] D'Mello, S.K., Person, N., Lehman, B.A. 2009. Antecedent-Consequent Relationships and Cyclical Patterns between Affective States and Problem Solving Outcomes. *Proceedings of 14th International Conference on Artificial Intelligence in Education* (Brighton, UK, July 6-10, 2009), 57-64.

[11] Forbes-Riley, K., Litman D. 2009. Adapting to Student Uncertainty Improves Tutoring Dialogues. *In Proceeding of the 2009 Conference on Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling* (Brighton, UK, July 6-10, 2009), 33-40.

[12] Gee, J.P. 2007. *Good video games+ good learning: Collected essays on video games, learning, and literacy*

(Mar. 2007). Peter Lang Pub Incorporated, Bern, Switzerland.

[13] Hone, K. 2006. Empathic Agents to Reduce User Frustration: The Effects of Varying Agent Characteristics. *Interacting with Computers,* 18, 2 (Mar. 2006). Elsevier B.V., Oxford, UK, 227-245.

[14] Koedinger, K.R., Corbett, A.T. 2006. Cognitive Tutors: Technology bringing learning science to the classroom. *The Cambridge Handbook of the Learning Sciences* (Apr. 2006). Cambridge University Press, Cambridge, UK, 61-78.

[15] Klein, J., Moon, Y., Picard, R. 2002. This computer responds to user frustration – Theory, design, and results. *Interacting with Computers*, 14, 2 (Feb. 2002). Elsevier B.V., Oxford, UK, 119-140.

[16] Kort, B., Reilly, R., Picard R. 2001. An Affective Model of Interplay between Emotions and Learning: Reengineering Educational Pedagogy—Building a Learning Companion. *Proceedings of the 1st IEEE International Conference on Advanced Learning Technology: Issues, Achievements and Challenges* (Madison, WI, Aug 06-08, 2001), 43-48.

[17] Lee, D.M., Rodrigo, M.M., Baker, R.S.J.d., Sugay, J., Coronel, A. 2011. Exploring the Relationship between Novice Programmer Confusion and Achievement. *Proceedings of the 4th bi-annual International Conference on Affective Computing and Intelligent Interaction* (Memphis, TN, Oct 9-12, 2011).

[18] McQuiggan, S.W., Robison, J.L., Lester, J.C. 2010. Affective Transitions in Narrative-centered Learning Environments. *Educational Technology & Society,* 13, 1(Jan. 2010). International Forum of Educational Technology & Society, 40-53.

[19] Ocumpaugh, J., Baker, R.S.J.d., Rodrigo, M.M.T. 2012. Baker-Rodrigo Observation Method Protocol (BROMP) 1.0. Training Manual version 1.0. Technical Report. New York, NY: EdLab. Manila, Philippines: Ateneo Laboratory for the Learning Sciences.

[20] Perneger, T.V. 1998. What's Wrong with Bonferroni Adjustments. *British Medical Journal*, 316 (Apr. 1998). BMJ Publishing Group, London, UK, 1236-1238.

[21] Rodrigo, M.M.T., Baker, R.S.J.d., Jadud, M.C., Amarra, A.C.M., Dy, T., Espejo-Lahoz, M.B.V., Lim, S.A.L., Pascua, S.A.M.S., Sugay, J.O., Tabanao, E.S. 2009. Affective and Behavioral Predictors of Novice Programmer Achievement. *Proceedings of the 14th ACM-SIGCSE Annual Conference on Innovation and Technology in Computer Science Education* (Paris, France, July 06-09, 2009), 156-160.

[22] Rodrigo, M.M.T., Baker, R.S.J.d., Nabos, J.Q. 2010. The Relationships between Sequences of Affective States and Learner Achievements. *Proceedings of the 18th International Conference on Computers in Education* (Putrajaya, Malaysia, Nov 29 - Dec 3, 2010).

[23] Sabourin, J., Rowe, J., Mott, B., Lester, J. 2011. When Off-Task is On-Task: the Affective Role of Off-Task Behavior in Narrative-Centered Learning Environments. *Artificial Intelligence in Education,* 21 (2011). Springer, Berlin/Heidelbery, Germany, 534-536.

[24] Sao Pedro, M. A., Baker, R.S.J.d., Montalvo, O., Nakama, A., Gobert, J.D. 2010. Using Text Replay Tagging to Produce Detectors of Systematic Experimentation Behavior Patterns. *Proceedings of the 3rd International Conference on Educational Data Mining*, 181-190.

Data Mining in the Classroom: Discovering Groups' Strategies at a Multi-tabletop Environment

Roberto Martinez-Maldonado, Kalina Yacef, Judy Kay
School of Information Technologies
The University of Sydney,
NSW 2006, Australia
{roberto, judy, kalina}@it.usyd.edu.au

ABSTRACT

Large amounts of data are generated while students interact with computer based learning systems. These data can be analysed through data mining techniques to find patterns or train models that can help tutoring systems or teachers to provide better support. Yet, how can we exploit students' data when they perform small-group *face-to-face* activities in the classroom? We propose a novel approach that aims to address this by discovering the strategies followed by students working in small-groups at a multi-tabletop classroom. We apply two data mining techniques, sequence and process mining, to analyse the actions that distinguish groups that needed more coaching from the ones that worked more effectively. To validate our approach we analysed data that was automatically collected from a series of *authentic* university tutorial classes. The contributions of this paper are: i) an approach to mine face-to-face collaboration data unobtrusively captured at a classroom with the use of multi-touch tabletops, and ii) the implementation of sequence mining and process modelling techniques to analyse the strategies followed by groups of students. The results of this research can be used to provide real-time or after-class indicators to students; or to help teachers effectively support group learning in the classroom.

Keywords

Collaborative Learning, Sequence Mining, Process Mining, Interactive Tabletop, Classroom

1. INTRODUCTION

Collaborative face-to-face activities can offer particular advantages compared to computer-mediated group work [17]. These include a natural channel for both verbal and non-verbal communication, improved perception of quality of group discussions, and an increased productivity in completing tasks [17, 18]. The classroom is a common environment in which the teacher can foster face-to-face collaboration skills acquisition by making use of small-group activities [8]. However, even in small-group activities, it is challenging for teachers to provide students the attention that they may require and be aware of the process followed by each group or their individual contributions [21]. Commonly, teachers try to identify the groups that work effectively to leave them work more independently and be able to devote time to groups needing their attention.

Multi-user shared devices, such as interactive tabletops, provide an enriched space where students can communicate face-to-face with each other and, at the same time, interact with a large work area that has access to digital content and allows the creation of persistent artefacts [14]. Interactive tabletops may afford new possibilities to support learning but they also introduce additional challenges for a new space of interaction. In order for these tabletops to be integrated into the classroom, as with any emerging technology, they should provide additional support to teachers compared with what they can currently do without such technology [4]. Currently, these devices are making their way into the classroom in the form of multiple interactive tabletops that have the potential of providing teachers with new ways to control groups [1, 11]; plan and enact authentic collaborative activities [10]; and monitor students' progress [5, 11].

At the same time, the increasing usage of technology for learning and instruction has made it possible to collect students' traces of activity resulting in large amounts of data gathered while they interact with computer based learning systems. These data can be analysed through data mining techniques to find patterns or train models that can help tutoring systems or teachers to provide enhanced support [3]. Although there is substantial research work on mining students' data obtained from individual or online learning systems, there is still little research on automatically exploiting the data generated when learners perform small-group *face-to-face activities in the classroom*.

A slightly hidden potential of interactive tabletops is that they can open new opportunities for capturing learners' *digital traces of activity,* offering teachers and researchers the possibility to inspect the process followed by students and recognise patterns of group behaviour [12]. This paper presents a novel approach that focuses on analysing face-to-face collaboration data to discover the strategies that distinguish groups that need more coaching from the ones that work effectively.

To validate our approach we analysed data that was automatically and unobtrusively collected from authentic tutorials that covered part of the regular curricula of a university subject in the area of Management. The teacher designed a small-group collaborative activity, based on the concept mapping learning technique, using our multi-tabletop classroom environment called MTClassroom [11]. This allows multiple small-groups of students to work around a number of interactive tabletops, perform a series of tasks, discuss a topic and provide a solution to a case proposed by the teacher. The system automatically logs identified students' actions on the shared device and all the steps that groups performed to build a collaborative artefact. We describe the application of two data mining techniques. First, we used a sequential pattern mining technique to look for patterns that can help find differences between groups according to the teacher assessment. Then, we used the Fuzzy Miner tool [6] to discover the processes most often followed by both high and low achieving groups. The main contributions of this paper are: i) an approach to mine face-to-face collaboration data unobtrusively captured at a classroom with the use of multi-touch tabletops, and ii) the implementation of sequence mining and fuzzy modelling techniques to analyse and discover strategies followed by groups of students.

The paper is structured as follows. The next section describes the state of research on the areas of interactive tabletops in the classroom and data mining for collaborative learning. Then, we present details of the multi-tabletop tutorials and our technical infrastructure. Section 4 presents the motivation and design of study. Section 5 describes the data pre-processing and the methods. Section 6 presents a discussion of the results. Section 7 states the conclusions and the avenues for future research.

2. RELATED WORK

There is a steady growth of the usage of tabletops in education. More specifically, there are a number of research projects that have used *multiple tabletops or shared devices in the classroom*. One of these is Synergynet [1], a multi-tabletop setting that has served to study the ways school kids collaborate and interact to achieve group goals. This project also included the design of tools for the teacher to *control* the classroom activities. Another approach was proposed by Do Lenh [5], who developed a setting for training on logistics, that consisted of four tangible horizontal devices that could be *orchestrated* by the teacher using paper-based commands or through a remote computer. This project also offered minimalist indicators of progress of each small group presented at a wall display. Even though these two previous projects included real students and teachers, they were mostly designed and deployed as experimental scenarios. A different approach was followed by Martinez-Maldonado et al. [10], who presented a multi-tabletop system that permitted teachers to *assess the design* and *enactment* of their planned classroom activities through the use of analytics tools. This is the only previous work that has focused on exploiting the collected data from a multi-shared device environment to describe the activities that occur in an authentic classroom.

In the case of *data mining applied to collaborative settings*, the closest study to ours was presented by Martinez-Maldonado et al. [12]. It consisted in extracting and clustering frequent sequential patterns to then link them with high level group actions at a pen-based tabletop learning application called Mysteries. One important study, even though not related to tabletops, was performed by Perera et al. [20] who explored the usage of sequence mining alphabets and clustering to find trends of interaction associated with effective group-work behaviours in the context of a software development tool. Moreover, Anaya et al. [2] analysed a computer-mediated learning tool to classify and cluster learners according to their level of collaboration.

The work reported in this paper is the first effort we are aware of that proposes an integrated solution, inspired by authentic needs of the teacher in the classroom, to exploit the students' data that can be captured by multiple tabletops though the application of a data mining technique and a process modelling tool.

3. MULTI-TABLETOP TUTORIALS

This section describes our technical infrastructure that consists of: the multi-tabletop classroom, a teacher's dashboard, the system for capturing identified learners' actions and a learning tool for building concept maps. We also describe the teacher's design of the tutorials.

3.1 Technical Infrastructure

Our multi-tabletop classroom is called MTClassroom [11]. This has a number of interconnected multi-touch interactive tabletops (four in this study). Figure 1 shows an instance of MTClassroom for a demo tutorial. Each tabletop consists of a 26 inch PQlabs overlay placed over a high-definition display that is enriched with Collaid [9]. Collaid is a system that provides an ordinary interactive tabletop the capability of automatically and unobtrusively identifying which person is touching where, based on an over-head depth sensor (www.xbox.com/kinect). Using this system, each tabletop can identify actions performed by each student according to their seating position.

The logging system of each tabletop records the activity logs to a central synchronised repository that can be accessed in real time by other services. One of these is a teacher's dashboard called MTDashboard [11]. This dashboard provides functions for the teacher to orchestrate the tabletops (e.g., blocking the touch input of all tables or moving the class to the next phase) and to see key live-indicators of work progress of each small-group. Figure 2 shows the teacher holding the dashboard, displayed on a tablet device, while she provides feedback to a group. The classroom activity consisted in elaborating collaborative concept maps about a case proposed by the teacher. *Concept mapping* is a technique that promotes learning by allowing students to visually represent their understanding in the form of *concepts* associated by *linking words* that creates statements [16]. We used a minimalist version of a tabletop concept mapping application called Cmate [9]. Cmate provides students with a list of concepts and linking words suggested by the teacher, and also allows them to type their own words, in order to build a concept map that represents their solutions. Prior to the tutorials, the teacher creates a *master concept map* with the crucial concepts and links that learners are expected to include in their maps.

3.2 Tutorials Design

Eight tutorial sessions were organised in the School of Business of the University of Sydney during week 6 of semester 2, 2012 for the course: *Management and organisational ethics*. The teacher designed a case resolution activity to cover the topic of the curricula corresponding to that week. A total of 140 students attended these tutorials (from 15 to 20 students per session) that were organised in groups of 4, 5 or 6 students.

The teacher designed the tutorial script as follows: 1) *Introduction* (10 minutes): the teacher forms groups, explains to students how to use the concept mapping application and

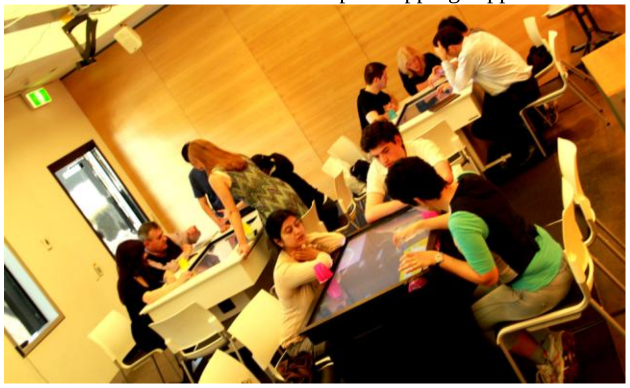


Figure 1. MTClassroom: a multi-tabletop classroom with capabilities for capturing differentiated students' activity.

introduces the first activity. 2) *Activity 1* (10 min.): using the MTDashboard, the teacher cleans up the four tabletops for all groups to start at the same time. Students are instructed to create a concept map that represents how the main actors of the case are associated. 3) *Reflection 1* (5 min.): the teacher blocks the tabletops, leads a short class discussion about partial solutions and introduces Activity 2. 4) *Activity 2* (15 min.): this is for the teacher "*the most important activity of the tutorial from the learning perspective*". The teacher unblocks the tabletops, and students discuss and focus on representing a final solution to the case in their concept map. 5) *Class sharing and reflection* (10 min.): the teacher asks each group to share their solution with the class. After each group has explained their map, the teacher summarises the outcomes of the tutorial, finishes the session and assesses each group in private. The class time was fixed to 50 minutes. Details of these tutorials can be found in [11].

4. STUDY DESIGN AND DATASET DESCRIPTION

The teacher in the classroom can face a number of challenges related with control, awareness and resources management [22] which depend on a number of factors that may fall out of the scope of what tabletop systems can capture. The tabletop systems are not totally aware of the classroom situation, for example, if a group of students is talking, if they work on-task or if someone needs to leave the class. The teacher can have a better idea of the productivity of students' discussions within each group, however, one of the main conclusions after finishing the tutorials was that for the teacher it is not easy to know aspects of the final artefacts that students built or their individual contributions [11].

In a post-tutorial interview the teacher expressed her view as follows: *"I don't want to see a lot of information in the dashboard, this can be distracting. But more information can be provided after the tutorials for assessment, like who did what, when, and the quality of the work".* These are indeed the aspects of group work that tabletops are aware of in detail. Our system can capture: 1) differentiated students' action on the tabletop; 2) the sequential actions performed to build the group artefact.

Inspired by the above teacher needs, but framed on what tabletops can actually capture in an authentic classroom, we propose an approach to distinguish strategies followed by groups that either needed more coaching or worked effectively. We analyse three sources of contextual information i) identified individual actions on the tabletop that can occur in parallel, in turn, or on other students' objects, ii) the quality of students' actions according to the teachers' artefact, and iii) the impact of students' actions on the group artefact. In this paper we focus on the students' actions performed in Activity 1. This is important because a certain degree of success in Activity 1 is required for Activity 2. This also allows the approach to be applicable in real-time, to provide feedback to teachers before the tutorial is over, so they can target their support during Activity 2.

Table 1. Possible actions on the concept mapping tabletop system

High impact actions (content and structure)	Low impact actions (layout)	No impact actions
Add a concept/link	*Move* a concept/link	*Open* or *close* menus
*Del*ete a concept/link	*Merge* two links	*Move*/*scroll* menu-concepts
Edit a concept/link		

Figure 2. A teacher attending a group while holding the MTDashboard

The teacher assessed groups at the end of each tutorial, using one of three possible values: low, medium or high achievement. The teacher specified that the assessment criteria mostly considered the *quality of each group solution* presented at the end of the tutorial and the *quality of their discussions* during the tutorial. We considered the activity data of all the 32 groups divided in two sets: 20 groups that were high achieving and 12 groups that were medium or low achieving.

The *initial raw data* of each group consists of a long sequence of actions in which each element is defined as: {*Resource, ActionType, Author, Owner, Time, Relevance*}, where *Resource* can be: *Conc* (concept), *Link* (proposition) or *Menu*. *ActionType* can be: *Add* (create a concept or link), *Del* (delete), *Mov* (move) links, *Chg* (edit), *Scroll*, *Open* or *Close* (a menu). *Author* is the learner who performed the action, *Owner* is the learner who created an object or owns a menu, *Time* is the timestamp when the action occurred and *Relevance* indicates if the concept or link belongs to the crucial elements of the teacher's map. Table 1 lists all the possible actions in the dataset grouped by their impact on the group concept map. Some examples of actions are: {ConceptA, Add, 3, 3, 17:30:02, Crucial}, when a learner adds a crucial concept to the map; {LinkY, Move, 2, 6, 17:30:04, Irr}, when s/he moves a link created by another learner; and {MenuConcepts, Open, 2, 2, 17:30:07,-} when s/he opens the list of suggested concepts. The original sequence obtained for each group contained from 74 to 377 *physical actions.*

We address four research questions regarding the strategies and characteristics that can differentiate groups according to their extent of achievement. The formulation of these is based on the triangulation of the nature of the available data (differentiated students' actions and their impact on their artefact), the teacher's needs (awareness on students' participation and quality of their work), and open issues in the study of multi-tabletop classrooms [10]. Our research questions are the following. 1) *Can we distinguish groups by inspecting patterns of parallelism and turn-taking*? As the teacher is interested in the participation of all students in the construction of the group solution [10], we analyse whether it is possible to find differences among groups where students worked at the same time (in parallel or taking turns) or not. 2) *Can we distinguish groups by inspecting students' interactions on others' objects*? Other studies inspired this question; these have suggested that interacting with what others' have done may trigger further discussion that is beneficial for tabletop collaboration [11, 13]. 3) *Can we distinguish groups* *by inspecting students' map quality*? This and the next question are directly inspired by teachers' needs, as noted above, and the data captured by our system about the groups' artefacts and the process followed to build them. 4) *Can we distinguish groups by inspecting the process followed by students' actions and their impact on the group artefact?*

5. METHOD

Sequential mining and process mining are techniques that have been used to identify patterns in educational datasets by considering the order of students' actions [7, 12, 19]. We used a sequential pattern mining technique called *differential sequence mining* [7] to distinguish strategies followed by groups that were either high or low achievers and address each of our first three research questions. For these, we analysed two of the sources of contextual information listed in the previous section: i) identified actions on the tabletop and ii) the quality of students' artefact. In order to address the fourth question, and analyse the strategies that distinguish groups according to iii) the impact of students' actions on the group map, we used the *Fuzzy Miner* tool [6]. Next subsections present the motivation for using these tools, the data pre-processing and the implementation of each technique.

5.1 Sequence mining

One of the data mining techniques that has been succesfully applied to identify patterns that differentiate high from low achieving students is differential sequence mining (DSM) [7]. In general, a sequential pattern is a consecutive or non-consecutive ordered sub-set of a sequence of *events* that is considered frequent when it meets a minimum *support* threshold. In educational contexts, the *events* commonly correspond to individual or grouped students' actions logged by the learning system. The DSM algorithm extracts frequent consecutive ordered sequences of actions from 2 datasets and performs an analysis of significance to obtain the patterns that differentiate them. The actions can also contain contextual information as defined by an *alphabet*. Alphabets can be used to *encode* each action to a set of concatenated keywords. In our study, each action was encoded to the format *{Resource-ActionType-Context}*. We implemented a DSM solution to investigate the differential patterns in terms of degree of parallelism, actions of students on others' objects and relevance of the links and concepts students use according to the teacher;s map. Table 2 presents the keywords of each of our three alphabets. The encoded actions encoded using any alphabet should contain at least one keyword for the *Resource* column and one for the ActionType column. We add one keyword of the corresponding contextual information (three rightmost columns in Table 2) according to the Resource type. Alphabet 1 aims to model the differentiated individual actions performed on the tabletop that occur in parallel (with other students' actions, keyword: *Parallel*), in turns (when the previous action was performed by a different student, keyword: *Other*), or as a series of actions by the same student (*Same*). Alphabet 2 models the actions that students perform on their own objects (*Own*) or on other students' objects (*NoOwn*). Finally, Alphabet 3 indicates whether the concept or link involved in the action belongs to the crucial objects defined by the teacher (*Cruc* or *NoCruc*).

In a previous study, we found that it is very important to consider the periods of significative inactivity registered by the tabletop [11]. During these periods of inactivity students can be having productive discussions, off-task talking or not working collaboratively at all. In our study, even when we do not perform speech detection, it is important to at least consider the occurrence of inactivity. To define a period of inactivity, we explored the time gap between each action performed on the tabletop. We found that time gaps between actions below one standard deviation from the mean ($<\mu+1\sigma$) account for the 92% of the set. (μ= 4.30 seconds, σ= 8.62, $\mu+1\sigma$=13 seconds). This means that a period above 13 seconds without logged actions can be considered as a block of inactivity. We defined these blocks as *short* when the gap was between 13 ($\mu+1\sigma$) and 22 ($\mu+2\sigma$) seconds, and *long*, for gaps longer than 22 seconds ($\mu+2\sigma$). We detected from 6 to 19 periods of inactivity in each group.

The output of the DSM algorithm, using the three alphabets, consists of three sets of frequent sequential patterns that differentiate high from low achieving groups according to the teacher's assessment. In this study, we set a minimum support of 0.5 to consider a pattern as frequent and a maximum error of one to allow matching sequences with up to 1 different action, similarly to previous work on educational data exploration [7].

5.2 Process mining

The sequence mining approach presented above can extract patterns of activity that distinguishes groups; however, it does not give insights of the higher level view of the *processes* followed. The Fuzzy miner [6] is a process discovery tool that can generate a meaningful abstraction of a general process, from multiple instances by distinguishing the activities that are important. It is especially suitable to mine *unstructured* processes, like the concept mapping construction in this study. The input of this algorithm is a series of consecutive actions, or group of actions. The result is a directed graph in which each node represents an action, or group of actions, and the edges represent the transitions between these. The nodes and edges that appear in the graph should meet a *conformance threshold* based on the instances that were used to build the model.

The objective of this second analysis is to discover the meaning of the higher level steps that high and low achieving groups performed to build the concept map and the impact of such actions. For this, we performed the following data preparation before using the Fuzzy miner tool.

Table 2. Keywords included in the alphabets for the sequential pattern mining.

Resource	Action type		**Alphabet 1** *Parallelism –turn taking*	**Alphabet 2** *Actions on others' objects*	**Alphabet 3** *Master map distance*
Concept (Conc)-C	*Add -C,L*	*Delete (Del)-C,L*	*Parallel*	*Own*	*Cruc (C,L)*
Link -L	*Edit (Chg) -C,L*	*Merge (Move)-L*	*Other*	*NoOwn*	*NoCruc (C,L)*
Menu -M	*Move -C,L,M*	*Open -M*	*Same*		
		Close -M			
Inactivity block (Inact)-B	*Short(Shrt) -B*	*Long –B*			

1) Data grouping. We grouped the actions into periods of activity in order to generalise similar actions according to their impact on the concept map. First, we explored the number of actions contained in each period of activity between periods of inactivity. Figure 3 illustrates the frequencies of the number of actions within blocks of activity in the dataset (μ= 12.85 actions, σ= 17.68). The distribution shows a high frequency of periods with a small number of continuous actions, and a long tail of longer sets of actions. In fact, the 71% of the periods of continuous activity were below the mean size (13 actions) and the 87% of them were below one standard deviation from the mean (30 actions). We considered the mean (13 actions) as a practical threshold for the maximum size of a block of activity.

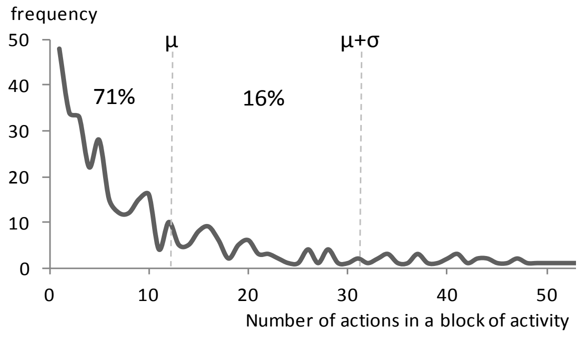


Figure 3. Distribution of the length of the sets of activity in terms of number of actions.

2) Actions categorisation. Based on the definition and previous research on concept mapping [15, 16], we categorise students' actions according to their impact on the group map. Actions that make a change in the structure or content of the concept map are categorised as *High-impact actions.* These include actions that modify the quantity or content of concepts and links (Table 1). The second category is *Low-impact actions,* which includes actions that modify the layout of the map, which is important for the activity, but not crucial. These actions include moving concepts and links, or merging links. Finally, actions performed on the menus of the application belong to *No-impact actions*.

3) Blocks categorisation. Each block was categorised according to the actions that occurred within that period following the next rules: *HighOnly* for blocks that contained only high-impact actions and some no-impact actions; *HighLow,* if the block contained at least one high-impact and one low-impact actions; *LowOnly*, for blocks that contained only low-impact actions and no-impact actions; and *NoImpact* if the block contained just no-impact actions. Periods of inactivity were categorised as either *InactShort* or *InactLong*, as explained earlier.

4) Addition of contextual information. According to our research aim, we highlighted the importance of distinguish the learners who work on their own or on other students' objects. For this, we added the information about who touched which object with the keywords *NoOwn* if most of the actions were performed on others' objects and *Own* if the actions were performed on the same learners' objects.

After performing the data preparation we divided the dataset into two sets, one for high and one for low achieving groups, as we did for the sequential mining. We generated two corresponding fuzzy models using the plugin implemented in the ProM framework (www.processmining.org). Then, we performed two model analyses: analysis of the number of active learners, and a validation of the models to discriminate groups.

Analysis of number of active learners. We explore whether there is a difference in the number of learners that were actively involved in each of the significant activities that appear in each fuzzy model (the nodes of the model). For the latter, the explored values corresponded to blocks of activity in which only one learner (*1u*), two (*2u*), or more than 2 learners (*+u*) were involved in the actions within a block of activity. This takes into account that all groups had from 4 to 6 group members. No correlation was found between the group size and the level of achievement of each group (r = 0.2).

Validation of the models. We performed a cross validation of the two models to evaluate if they can be used to effectively differentiate high from low achieving groups. To do this, we calculate, for each group process, the level of conformance of both fuzzy models and validate that the model that fit the most corresponds to the level of achievement of the group.

6. RESULTS AND DISCUSSION

6.1 Sequence mining results

After applying the DSM algorithm on the encoded datasets according to our three alphabets, we selected the patterns whose *instance* support (number of times the pattern is repeated within a group log) differed between the high and low achieving groups ($p<=0.10$) and that were composed of at least 2 actions. Table 3 presents the top-4 most frequent sequences for each of the three alphabets explored in this part of the study.

Alphabet 1: focused on parallelism and turn-taking. We obtained a total of 23 *differential patterns* for groups that were either high or low achieving after analysing the first encoded dataset. The top sequences in Table 3 indicate the presence of actions in parallel for *move* events (sequence A) and actions that contain the keyword *Other,* when adding and moving elements of the concept map (sequences B, C and D). These provide evidence that in high achieving groups more than 1 student quite often interacted with the tabletop at the same time. In fact, the keywords *Parallel* and *Other* appeared in 13% and 66% in the frequent patterns of high groups, while in the low achieving groups there were no patterns with the keyword *Parallel* and the keyword *Other* only appeared in the 30% of them.

Alphabet 2: focused on actions on others students' objects. In this case, we obtained a total of 29 *differential patterns*. Table 3 shows that in high achieving groups, students tended to interact with objects created by other students, such as moving and adding links using others' concepts, either followed or preceded by periods of inactivity (keywords *NoOwn* and *Inact* in sequences I, J, and L). The keyword *NoOwn* appeared two times more often in the frequent sequences of the high groups than in the achieving groups (in 42% and 22% of the sequences respectively). The presence of actions on students' own objects (*Own*) was similar in all groups.

Alphabet 3: focused on Master map distance. We obtained 28 *differential patterns* by analysing the encoded dataset. This includes contextual information of the concepts and links that belong to the crucial elements defined by the teacher. The patterns in Table 3 show that in high achieving groups, students tended to work with more crucial elements than low achieving groups. However, an analysis of all patterns found showed that there was not a large difference in actions performed on crucial elements (keyword *Cruc* was present in 87% and 84% of the patterns of high and low achieving groups respectively). However, the key difference was that high achieving groups interacted with less non-crucial concepts and links (keyword *NoCruc* was in 19% and 73% of the patterns of high and low achieving groups respectively).

The sequences of events extracted using this technique, provides some insights about the strategies followed by groups. Low achieving groups tend to have long periods of inactivity on the tabletop before or after creating links or performing a chain of actions that affect the layout of their concept map (e.g. action *Inact-Long* in patterns G, H, N, O and X). High achieving groups also had periods of inactivity, but these were shorter. Long periods of inactivity appeared two times more in the low achieving groups, followed or preceded by other actions (*Inact-Long* appeared in 48% and 22% of the sequences of high and low achieving groups respectively). There was no difference in the appearance of short periods of inactivity.

These findings suggest that, to discover the strategies followed by groups, this approach offers a limited view of the meaning of the actions. The frequent sequences that were found can be used to build a model or benchmark to 'detect' if students' actions are similar to either high or low achieving groups. However, the patterns themselves do not provide information about the process that groups followed during the activity that would be easily associated with groups' behaviours.

6.2 Process mining results

Figure 4 shows the resulting fuzzy models after applying the second approach to mine the process of both, high and low achieving groups where the conformance with their corresponding datasets was above 80%. Nodes of the graph represent categories of action blocks of activity and the edges the transitions between these. Each node contains: the name of the block category, the conformance of the block with the dataset, and the rates of active students that were involved in the activities (*1u*, *2u* and *+u*). Nodes with conformance rates below to 0.1 were not considered in the models to include the majority of the block categories but disregarding the actions that rarely appeared in the data and that would make the graph unnecessarily complex. The numbers next to the edge lines are indicators of conformance of the transitions with the datasets.

By visually comparing both graphs we can highlight that they share the same core blocks of activity. These include: the blocks *Inact-Short* and *Inact-Long* (marked with an orange small square in the top left of the node). We confirmed the results obtained with the sequence mining, where low achieving groups showed more long periods of inactivity compared with high groups (conformance of 0.68 and 0.98 respectively). Both models also have in common the categories *HighLow-NoOwner* and *HighLow-Owner* (blue markers) that represent activity that combined high and low impact actions on the group map (conformance of 1 and around 0.4 respectively). The last similarity, in terms of nodes, corresponds to blocks of low impact actions where students interacted with other students' objects (*LowOnly-NoOwner*, red markers).

The nodes marked with a yellow star correspond to activity blocks that appear in one model but not in the other. High achieving groups, contrary to the expected, presented more blocks of actions with no impact on the concept map (*NoImpact-Owner/NoOwner*). However, both nodes had the least conformance with the model (0.11 and 0.2 respectively). In contrast, low achieving groups presented blocks of activity with only high impact actions (*HighOnly- Owner/NoOwner*). The conformance of these blocks was not low (conformance of 0.37 and 0.74 respectively).

However, the main difference between the models is in the structure of the transitions. For the model of high achieving groups, there is only one transition between *different* blocks of activity. This was, in addition, not very frequent (0.08 conformance, between *NoImpact-Owner* and *HighLow-Owner*). By contrast, the model of low achieving groups contains 5 transitions between activity nodes with a conformance of up to 0.17 (between *HighLow-Owner* and *LowOnly-NoOwner*). Additionally, we did not find any observable difference in the actions performed on other students objects (*NoOwner*) and students' own objects (*Owner*).

Table 3. Top-4 most frequent sequences after applying differential sequence mining on each encoded dataset.

Alphabet 1	**High achieving groups**	**Low achieving groups**
	A- {Menu-Mov-Same}>{Menu-Mov-Same}>{Menu-Mov-**Parallel**}	E- {Link-Add-Same}>{Link-Rem-Same}>{Con-Mov-Same}
	B- {Con-Mov-**Other**}>{Link-Add-Same}>{Con-Mov-Same}> {Link-Add-Same}	F- {Link-Rem-Same}>{Con-Mov-Same}>{Link-Add-Same}
	C- {Inact-Shrt}>{Con-Mov-**Other**}>{Link-Add-Same}	G- {Link-Add-Same}>{Link-Chg-Same}>{Inact-Long}
	D- {Con-Mov-**Other**}>{Link-Add-Same}>{Con-Mov-Same}	H- {Inact-Long}>{Inact-Shrt}>{Con-Mov-Same}
Alphabet 2	**High achieving groups**	**Low achieving groups**
	I- {Con-Mov-**NoOwn**}>{Con-Mov-**NoOwn**}>{Link-Add-Own}> {Inact-Shrt}	M- {Inact-Shrt}>{Con-Mov- **NoOwn** }>{Link-Add-Own}> {Link-Chg-Own}
	J- {Inact-Shrt}>{Con-Mov- **NoOwn** }>{Con-Mov- **NoOwn** }> {Link-Add-Own}	N- {Link-Add-Own}>{Link-Chg-Own}>{Inact-Long}
	K- {Link-Mov- **NoOwn** }>{Link-Mov- **NoOwn** }>{Con-Mov- **NoOwn** }	O- {Link-Chg-Own}>{Inact-Long}
	L- {Inact-Shrt}>{Con-Mov- **NoOwn** }>{Con-Mov- **NoOwn** }	P- {Inact-Long}>{Inact-Shrt}>{Con-Mov- **NoOwn** }
Alphabet 3	**High achieving groups**	**Low achieving groups**
	Q- {Con-Mov-**Cruc**}>{Link-Add-**Cruc**}>{Con-Mov-**Cruc**}> {Link-Add-**Cruc**}	U- {Link-Rem-**NoCruc**}>{Con-Mov-**Cruc**}>{Link-Add-**Cruc**}> {Link-Chg-**NoCruc**}
	R- {Inact-Shrt}>{Con-Mov-**Cruc**}>{Con-Mov-**Cruc**}>{Link-Add-**Cruc**}	V- {Link-Chg-**NoCruc**}>{Link-Chg-**NoCruc**}>{Inact-Shrt}
	S- {Link-Add-**Cruc**}>{Link-Mov-**Cruc**}>{Con-Mov-**Cruc**}	W- {Inact-Shrt}>{Link-Add-**Cruc**}>{Link-Chg-**NoCruc**}
	T- {Link-Chg-Irr}>{Con-Mov-**Cruc**}>{Link-Add-**Cruc**}	X- {Con-Mov-**Cruc**}>{Link-Add-**Cruc**}>{Link-Chg-**NoCruc**}>{Inact-Long}

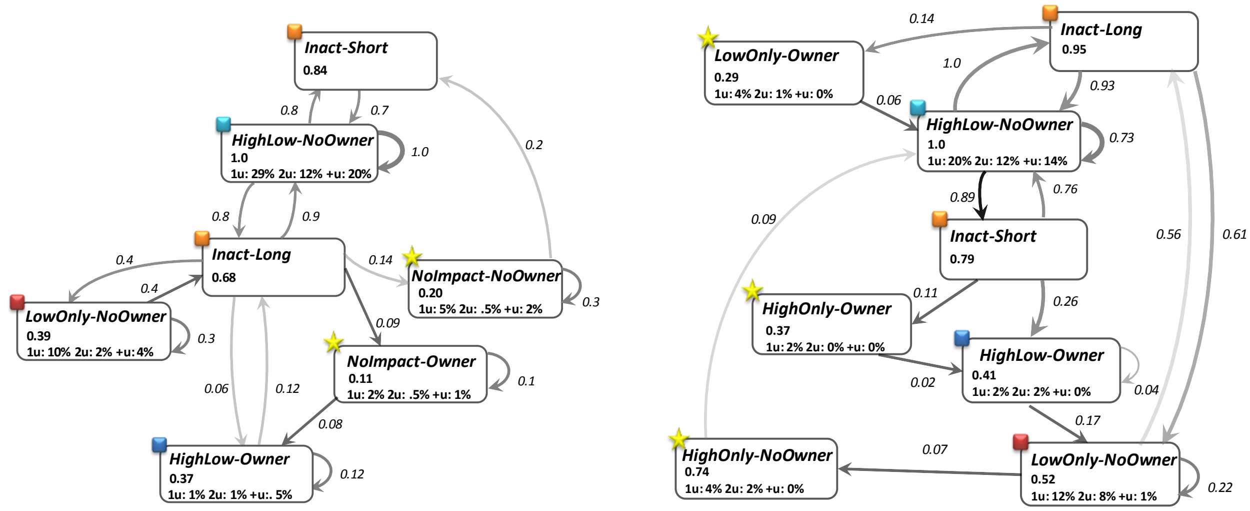

Inact-Short 0.84
HighLow-NoOwner 1.0 1u: 29% 2u: 12% +u: 20%
Inact-Long 0.68
NoImpact-NoOwner 0.20 1u: 5% 2u: .5% +u: 2%
LowOnly-NoOwner 0.39 1u: 10% 2u: 2% +u: 4%
NoImpact-Owner 0.11 1u: 2% 2u: .5% +u: 1%
HighLow-Owner 0.37 1u: 1% 2u: 1% +u:. 5%
0.8
0.7
1.0
0.2
0.8
0.9
0.4
0.4
0.14
0.3
0.3
0.09
0.1
0.06
0.12
0.08
0.12
Inact-Long 0.95
LowOnly-Owner 0.29 1u: 4% 2u: 1% +u: 0%
HighLow-NoOwner 1.0 1u: 20% 2u: 12% +u: 14%
Inact-Short 0.79
HighOnly-Owner 0.37 1u: 2% 2u: 0% +u: 0%
HighLow-Owner 0.41 1u: 2% 2u: 2% +u: 0%
HighOnly-NoOwner 0.74 1u: 4% 2u: 2% +u: 0%
LowOnly-NoOwner 0.52 1u: 12% 2u: 8% +u: 1%
0.14
1.0
0.93
0.06
0.73
0.89
0.76
0.09
0.56
0.61
0.11
0.26
0.02
0.04
0.17
0.07
0.22

Figure 4. Fuzzy model generated from groups' activity. Left: Fuzzy model of high achieving groups (Conformance: 86%, Cuttoff: 0.1). Right: Fuzzy model of low achieving groups (Conformance: 81%, Cuttoff: 0.1).

Next, we present the analysis of the number of students involved in the activities and the validation to determine if the observable differences can distinguish high from low achieving groups.

Active learners. Table 4 shows the results of the cumulated distribution of the number of learners involved in the periods of activity for both high and low achieving groups (partial rates displayed in the third line of text inside each node of Figure 4). Both high and low achieving groups presented more than the half of the blocks of activity performed by a single student (54/55%). The main difference found was that high achieving groups presented blocks of activity in which more than two learners were involved in comparison with low achieving groups (+u, 27% and 19% respectively). In low achieving groups most of the blocks of activity were performed by either one or two learners.

Table 4. Distribution of the number of active learners in blocks of activity

Achievement	One learner (1u)	Two learners (2u)	More learners (+u)
High	55%	**18%**	**27%**
Low	54%	**27%**	**19%**

Validation. In order to validate that the two models generated by the fuzzy miner are different and can be used to distinguish the process followed by either high or low achieving groups, we estimated how accurately each model will conform to each group's activity. We performed a cross-validation to compare the level of fit of both models to the data blocks of each group by measuring whether the conformance of the model that corresponded to the level of achievement of the group was higher. Table 5 shows the confusion matrix which layouts the results of this analysis. This indicates that the fuzzy model for low achievement could distinguish the 100% of the low achieving cases, however, three high achieving groups presented a superior conformance to this model. The conformance of the model of high achievement was higher for the high achieving groups in 17 of the 20 cases. The difference between the levels of fit of each model was statistical significant for high achieving groups (paired t(23) = 2.46, p = 0.0219) and very close to statistical significance (p<=.05) for the model of low achievement (paired t(7) = 2.16, p = 0.061).

Table 5. Validation of the fuzzy models

		Predicted class	
		High	**Low**
Actual class	*High*	17	3
	Low	0	12

7. CONCLUSIONS AND FUTURE WORK

This paper described the technological infrastructure and the data mining and process mining techniques used to analyse the strategies that distinguish high from low achieving groups in the classroom. We presented a novel approach to mine traces of collaboration of students working face-to-face on an activity linked with the regular curricula and supported by a number of teacher-orchestrated interactive tabletops. Our goal was to exploit students' data that was unobtrusively captured in an authentic classroom in contrast to a controlled experimental setting. This can make our approach immediately applicable in a real classroom context equipped with the technology required.

Sequential frequent mining was applied to find patterns of activity that differentiate groups. Results revealed interesting patterns that indicated students in high achieving groups worked more often in parallel, interacted with other students' objects and mostly focused on the crucial elements of the problem to solve. The fuzzy miner tool was used to model the process that groups followed by grouping and categorising students' actions. This modelling proved effective in helping distinguish part of the process followed by groups. High achieving groups tended to build their concept map interweaving periods of focused activity with periods of tabletop inactivity. Low achieving groups, by contrast, presented more transitions between different categories of blocks of activity including periods with only actions that caused high impact on the map. We also found that important strategies can be mined from early data. Our analysis was only performed on the data captured from the first activity of the classroom sessions. This gives time for the results of the analysis to be used by facilitators or group members in the classroom.

The knowledge generated by the sequence patterns and the fuzzy models can be used in several valuable ways. Firstly, derived groups' indicators can be displayed in a processed form on the teachers' dashboard to help them adapt in real-time the support to groups that might need closer attention. Secondly, the findings can be used to generate indicators of group learning to be shown to the teacher for after-class reflection or re-design of the activity or to reflect on students' performance or assessment. Thirdly, this information can be the basis to build student models that can be shown to learners to encourage reflection and self-assessment.

We acknowledge some current limitations of our approach. The first is that the technology to capture students' actions is not yet developed to automatically record verbal interactions in the classroom, which is crucial in collaborative work. However, our approach proved that even modest interaction data can provide insights about their strategies. Regarding the configuration of the data mining method, especially for the Fuzzy process mining, changing some thresholds can produce different results. For example, the size of blocks of activity was set to the mean number of actions between two periods of inactivity (13 actions). We explored the generation of fuzzy models using two more heuristics for the maximum block size: $\mu/2$ and $\mu+\sigma$. We obtained conformance rates as low as 60% for the block size heuristic of $\mu/2$, and very similar fuzzy models and conformance rates for the heuristic $\mu+\sigma$ compared to the one we used in the study. Even when these rates are lower than the ones we obtained using the μ heuristic, a deeper analysis of the configuration of the approach is part of the work in progress.

Our current work includes the exploration of ways to present the results of our approach to the teacher, in real time and for after class analysis. We also aim to connect the students' data that can be captured when they work at the tabletop with other activities that they perform, for example, through online learning systems.

8. REFERENCES

[1] AlAgha, I., Hatch, A., Ma, L. and Burd, L. Towards a teacher-centric approach for multi-touch surfaces in classrooms. *International Conference on Interactive Tabletops and Surfaces 2010 (ITS 2010)* ACM (2010), 187-196.

[2] Anaya, A. and Boticario, J. Application of machine learning techniques to analyse student interactions and improve the collaboration process. *Expert Systems with Applications*, 38, 2 (2011), 1171-1181.

[3] Baker, R. S. and Yacef, K. The State of Educational Data Mining in 2009: A Review and Future Visions. *Journal of Educational Data Mining (JEDM)*, 1, 1 (2009), 3-17.

[4] Cuban, L., Kirkpatrick, H. and Peck, C. High access and low use of technologies in high school classrooms: Explaining an apparent paradox. *American Educational Research Journal*, 38, 4 (2001), 813-834.

[5] Do-Lenh, S. *Supporting Reflection and Classroom Orchestration with Tangible Tabletops*. *PhD dissertation* École Polytechnique Fédérale de Lausanne, 2012.

[6] Günther, C. and Aalst, W. P. Fuzzy Mining – Adaptive Process Simplification Based on Multi-perspective Metrics. *Business Process Management* Springer Berlin Heidelberg (2007), 328-343.

[7] Kinnebrew, J. S. and Biswas, G. Identifying Learning Behaviors by Contextualizing Differential Sequence Mining with Action Features and Performance Evolution. *Educational Data Mining* (2012), 57-64.

[8] Leonard, S., Mary Elizabeth, S. and Samuel, S. D. Effects of Small-Group Learning on Undergraduates in Science, Mathematics, Engineering, and Technology. *Review of educational research*, 69, 1 (1997), 21-51.

[9] Martinez-Maldonado, R., Collins, A., Kay, J. and Yacef, K. Who did what? who said that? Collaid: an environment for capturing traces of collaborative learning at the tabletop. *International Conference on Interactive Tabletops and Surfaces (ITS2011)* ACM (2011), 172-181.

[10] Martinez-Maldonado, R., Dimitriadis, Y., Kay, J., Yacef, K. and Edbauer, M.-T. Orchestrating a multi-tabletop classroom: from activity design to enactment and reflection. *International Conference on Interactive Tabletops and Surfaces (ITS 2012)* ACM (2012), 119-128.

[11] Martinez-Maldonado, R., Dimitriadis, Y., Kay, J., Yacef, K. and Edbauer, M.-T. MTClassroom and MTDashboard: supporting analysis of teacher attention in an orchestrated multi-tabletop classroom. *Submitted to International Conference on Computer Supported Collaborative Learning (CSCL2013)* (2013).

[12] 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.

[13] Martinez-Maldonado, R., Yacef, K., Kay, J. and Schwendimann, B. An interactive teacher's dashboard for monitoring multiple groups in a multi-tabletop learning environment. *International Conference on Intelligent Tutoring Systems (ITS 2012)* (2012), 482-492.

[14] Müller-Tomfelde, C. and Fjeld, M. Tabletops: Interactive Horizontal Displays for Ubiquitous Computing. *Computer*, february (2012), 78-81.

[15] Nousiainen, M. and Koponen, I. T. Concept maps representing knowledge of physics: Connecting structure and content in the context of electricity and magnetism. *Nordic Studies in Science Education*, 6(2010), 155-172.

[16] Novak, J. Concept mapping to facilitate teaching and learning. *Prospects*, 25, 1 (1995), 79-86.

[17] Ocker, R. and Yaverbaum, G. Asynchronous Computer-mediated Communication versus Face-to-face Collaboration: Results on Student Learning, Quality and Satisfaction. *Group Decision and Negotiation*, 8, 5 (1999), 427-440.

[18] Olson, J. S., Teasley, S., Covi, L. and Olson, G. The (currently) unique advantages of collocated work. *Distributed work*(2002), 113-135.

[19] Pechenizkiy, M., Trcka, N., Vasilyeva, E., Aalst, W. v. d. and Bra, P. D. Process Mining Online Assessment Data. *2nd International Conference On Educational Data Mining* (2009), 279-288

[20] Perera, D., Kay, J., Koprinska, I., Yacef, K. and Zaiane, O. Clustering and Sequential Pattern Mining of Online Collaborative Learning Data. *IEEE Transactions on Knowledge and Data Engineering* 21, 6 (2009), 759-772.

[21] Race, P. *A briefing on self, peer & group assessment*. Learning and Teaching Support Network, York 2001.

[22] Zhang, D., Zhao, J. L., Zhou, L. and Jay F. Nunamaker, J. Can e-learning replace classroom learning? *Communications ACM*, 47, 5 (2004), 75-79.

Meta-Reasoning Algorithm for Improving Analysis of Student Interactions with Learning Objects using Supervised Learning

L. Dee Miller and Leen-Kiat Soh
Department of Computer Science and Engineering
University of Nebraska, Lincoln
{lmille, lksoh}@cse.unl.edu

ABSTRACT

Supervised learning (SL) systems have been used to automatically learn models for analysis of learning object (LO) data. However, SL systems have trouble accommodating data from multiple distributions and "troublesome" data that contains irrelevant features or noise—all of which are relatively common in highly diverse LO data. The solution is to break up the available data into separate areas and then take steps to improve models on areas containing troublesome data. Unfortunately, finding these areas in the first place is a far from trivial task that balances finding a single distribution with having sufficient data to support meaningful analysis. Therefore, we propose a BoU meta-reasoning (MR) algorithm that first uses semi-supervised clustering to find compact clusters with multiple labels that each support meaningful analyses. After clustering, our BoU MR algorithm learns a separate model on each such cluster. Finally, our BoU MR algorithm uses feature selection (FS) and noise correction (NC) algorithms to improve models on clusters containing troublesome data. Our experiments, using three datasets containing over 5000 sessions of student interactions with LOs, show that multiple models from BoU MR achieve more accurate analyses than a single model. Further, FS and NC algorithms are more effective at improving multiple models than a single model.

Keywords

Learning Object Analysis; Supervised Learning; Clustering; Meta-Reasoning

1. INTRODUCTION

Learning objects (LOs) are independent and self-standing units of learning content that are predisposed to reuse in multiple instructional contexts [2]. An example of an LO is a self-contained lesson on recursion with a tutorial, interactive exercises, and assessment questions. In general, the analysis of student interactions with LOs is important for many groups including students, instructors, researchers, and content developers [16]. First, for students, such analyses can improve student study strategies and allow for more self-regulated learning [1]. Second, instructors can use such analyses to choose appropriate LOs for their students [8]. Third, such analyses can help researchers and content developers investigate which student interactions are associated with the different learning outcomes [6].

One previously used approach for the analysis of student interactions with LOs is supervised learning (SL) systems [16]. SL systems learn a model from previously recorded sessions of student interactions (features) and learning outcomes (labels) that can predict the learning outcome for a specific session of student interactions with a high degree of accuracy.

SL systems have one main advantage over other approaches (e.g., statistical analysis): they learn the model automatically without the need for direct human intervention. First, learning the model can help students and instructors. Such a model predicts the learning outcome for a student in real-time based on the observed student interactions [15]. Such predictions can allow the LO to adjust the content presented to a student while he or she is taking the LO and provide real-time updates to instructors on student mastery of LO content. Second, a model learned automatically without human intervention provides independent, high-level guidelines on which types of student interactions are associated with the learning outcomes [4]. Such guidelines can serve as a useful starting point for further investigation by researchers and inform content developers on which parts of the LO may need to be revised.

However, SL systems have some potential problems which can limit the effectiveness of their models for analysis:

First, SL systems assume that the training data from previously recorded sessions comes from a single underlying distribution. Unfortunately, such training data is likely to come from multiple distributions and be highly diverse due to a wide variety of factors including students with different backgrounds, LOs with different content, instructors providing varying amounts of support for the LO content, etc. These factors make it ***difficult to learn a single model which can "fit" all this highly diverse training data*** and still achieve high accuracy.

Second, SL systems assume that the training data available is relatively "clean" being free of student interactions unrelated to the learning outcomes (i.e., irrelevant features) and errors in the student interactions and learning outcomes provided (i.e., noise). Unfortunately, such training data is all too likely to contain both irrelevant features and noise. Irrelevant features are relatively common when researchers are uncertain which student interactions are relevant and, thus, record as many student interactions as possible since they cannot retroactively record additional interactions. Noise is relatively common when developers fail to create assessment questions appropriate for all students and when students are motivated to "game the system" to achieve a certain learning outcome (e.g., a good grade on the LO). These factors make it ***difficult for a single model to achieve high accuracy on areas containing large amounts of "troublesome" training data*** with irrelevant features and/or noise.

Intuitively, we could address these problems and improve the effectiveness of SL systems for analysis by using an approach which first breaks up the training data into areas—each containing similar student interactions—and learns a separate model on each area. We could then identify which areas consistently contain "troublesome" training data and take steps to improve the models in those areas.

However, breaking up the training data and finding areas with "troublesome" training data are far from trivial tasks. As previously mentioned, the training data collected is likely to be highly diverse. Such diversity makes it difficult for an approach to find suitable areas balancing two factors. First, each area should contain similar data from a ***single distribution***. Second, each area should contain sufficient training data to support ***meaningful analysis***. Further, the model learned on an area is likely to fit the irrelevant features and/or noise in that area which can make it difficult to identify the areas containing ***troublesome training data***.

Therefore, we propose new meta-reasoning algorithm called the Boundary of Use (BoU MR) to improve the effectiveness of SL models for analysis of student interactions with LOs. Our algorithm first uses an iterative process, based on semi-supervised clustering, which breaks up the training data in different ways until suitable areas are found. These suitable areas, which we dub BoU clusters, include training data with similar student *interactions* from a ***single distribution*** which, nevertheless, have multiple learning outcomes to support ***meaningful analysis***. After clustering, the BoU MR learns a separate model for each of these clusters. Then, our algorithm evaluates each of these BoU clusters using a localized estimate to detect ***troublesome training data*** (e.g., feature selection for irrelevant features). Finally, our algorithm takes steps to selectively improve the models for the difficult BoU clusters containing troublesome training data; for example, removing irrelevant features and relearning the model on the "refined" training data.

In the following, we will investigate the BoU MR using two objectives. Objective 1 is to investigate the impact of breaking up the training data on LO datasets using three types of SL systems to learn the models: decision trees, support vector machines, and artificial neural networks. We compare the accuracy for a single model with that for multiple models from the BoU MR. Objective 2 is to investigate the effectiveness of BoU MR for improving its models on difficult BoU clusters. For this objective, we consider both feature selection and noise correction algorithms. We compare the accuracy to the BoU MR that uses these algorithms to help selectively relearn the models for difficult BoU clusters to a single model relearned after the same algorithm is applied to all the training data.

The rest of the paper is organized as follows. Section 2 provides background on SL systems and model improvement algorithms used in our study. Section 3 describes our BoU meta-reasoning algorithm in more detail. Section 4 discusses the experimental setup and results. Finally, we conclude and outline future work.

2. BACKGROUND

Here we discuss background on the SL systems and model improvement algorithms. Discussion of the LO datasets is deferred until Section 4.

2.1 Supervised Learning (SL) Systems

We consider three types of SL systems in the experiments below. To help demonstrate the effectiveness of our meta-reasoning algorithm for SL systems in general, we chose three widely used SL systems with very different properties. First, *artificial neural networks* (ANNs) learn a vector of weights on features in the dataset to choose the labels for new data [19]. ANNs consist of multiple nodes connected to threshold functions or to additional layers of nodes. ANNs are updated iteratively (e.g., using gradient descent) until they correctly predict the labels for the training data. Second, *decision trees* (for classification) learn a tree data structure to generate the labels for new data [19]. The decision tree first selects one feature as the root node and adds an edge for every label value. The decision tree continues to add nodes and edges recursively until all the training data has been sorted into groups with similar labels. The leaves are then set to the common label. Third, *support vector machines* (SVMs) learn a hyperplane to separate the training data such that data on the same side mostly have the same label [19]. SVMs first use a kernel function to transform all values for the dataset into higher dimensional space where they are linearly separable. Then, the SVM attempts to maximize the distance (i.e., margin) between the training data with different labels.

2.2 Model Improvement Algorithms

We consider two types of algorithms for improving the models in the experiments below: feature selection and noise correction.

First, feature selection (FS) algorithms find the subset of relevant features for the dataset using an evaluation criterion based on *filters* or *wrappers*. Filters evaluate the relevant features using only the intrinsic properties of the data whereas wrappers use the accuracy of the SL system model [10]. To avoid overfitting common to wrapper-based feature selection, we use a state-of-the-art filter-based FS algorithm called Lasso in the experiments below. Lasso FS uses a shrinkage method for FS which maintains a coefficient for each of feature (Hastie et al., 2011). Lasso computes these coefficients by using a pairwise coordinate descent approach to minimize the sum of squares subject to a constraint on the coefficients. Features whose coefficients have shrunk to zero are considered to be irrelevant and removed entirely from the dataset.

Second, noise correction (NC) algorithms are designed to identify noisy labels and then remove or replace them. There are two general types of noise correction algorithms [13]: (1) noise *tolerant* correction modifies existing SL systems to better accommodate noisy labels (e.g., rule-post pruning for decision trees) and (2) noise *filtering* detects noisy labels in the training data before the model is learned. We use both types of noise correction algorithms in the experiments below. We use decision trees rule post-processing and SVMs with soft margins both designed to accommodate noisy labels [19] for the former and a state-of-the-art algorithm called LSVM [18] for the latter. LSVM uses a hybrid approach for NC that starts by using a *k*-Nearest Neighbor algorithm to select the neighborhood of similar data around a given instance. LSVM then learns a local SVM on that local neighborhood (hence the name). Based on the maximal margin principle, if the LSVM incorrectly predicts the label for that instance, then the label is deemed noisy. Furthermore, to avoid accidently injecting noise into the training data, in our experiments, noisy labels are removed rather than being replaced.

3. METHODOLOGY

The central idea for the BoU meta-reasoning (MR) algorithm is to first break up the training data into special BoU clusters containing sessions with similar student interactions (i.e., instances) from a single distribution. At the same time, the cluster should sufficient data with multiple learning outcomes (i.e., labels) to support a meaningful analysis in the form of a SL model. In a very real sense, BoU clusters allow us to "zoom in" and get a more detailed analysis on highly diverse LO data than could be obtained using all the data together.

Next, our algorithm learns a separate model based on the training data in each BoU cluster. By using only the data in a single cluster, BoU MR guarantees that each model is more detailed and expressive on member data than a single model trained on all data together. After learning the separate models, our algorithm uses a localized estimate of the accuracy for each model by comparing the predicted and actual labels for the cluster members: ***correct*** *when the predicted matches the actual; otherwise,* ***incorrect***. Based on the predominant correct/incorrect label, BoU MR thus assigns each BoU cluster a type: correct BoU clusters where the model is doing well and incorrect clusters where the model is struggling on troublesome data.

Figure 1 provides an example of four such BoU clusters. In this figure, the clear circles are correct training data instances and grey circles are incorrect instances. Clusters 1 and 2 contain only correct instances and are thus flagged as correct. Note that Cluster 1 contains only a single label while Cluster 2 contains both labels—both are considered correct based on the localized estimate. Clusters 3 and 4 are flagged as incorrect since they contain a mix of both correct and incorrect instances.

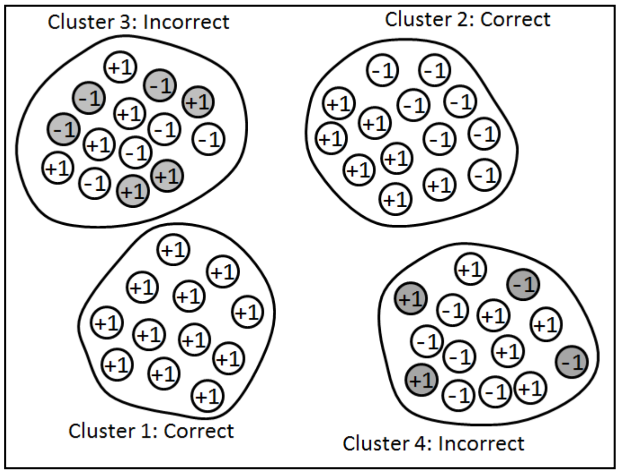


Figure 1. Example BoU Clusters. The grey data instances are those on which the model failed to predict the correct label.

After identifying the aforementioned clusters, BoU MR takes steps to improve the models for incorrect clusters. This is done by using either a feature selection or noise correction algorithm *selectively on only the incorrect clusters*. In this way, the models are left alone on correct clusters where they are already doing well. The BoU MR then relearns each model for a previously incorrect cluster using the refined data in that cluster.

Taken together, the combination of multiple, expressive models learned on highly diverse LO data and selective improvement on clusters containing troublesome data allows our BoU to improve the overall effectiveness of SL models for analysis of student interactions with LOs.

3.1 BoU Essential Components

Here we discuss the basic process and equations used for creating the clusters and deciding whether they are correct or incorrect. First, to make use of the BoU notion of clusters to find clusters with a single distribution that support meaningful analysis, we use a semi-supervised clustering (SSC) algorithm [9] to cluster the training data. Briefly, SSC algorithms create clusters based on both similarity in the training data and additional information available on how the session instances should be clustered (e.g., constraint that two instance must/cannot be clustered together). For our purpose—to find BoU clusters, the additional information that we incorporate for each session instance is whether or not the model predicts the label correctly or incorrectly. The actual SSC algorithm used is based on the *k*-Means variant discussed in Kulis et al. [9]. The modified objective function for BoU-style clusters π can be expressed as:

$$\sum_{c=1}^{k} \sum_{x_i \in \pi_c} \|x_i - m_c\|^2 + \sum_{\substack{x_i, x_j \in C \\ s.t. l_i \neq l_j}} w_{ij} \quad (1)$$

where C is the set of cannot-link constraints, w_{ij} is the penalty cost for violating a constraint involving points x_i and x_j and l_i refers to the model prediction for x_j s.t. $l_i \in \{\text{correct}, \text{incorrect}\}$. The first term in Eq. 1 is the *k*-Means objective function that chooses the closest centroid while the second term is a penalty function for assigning a data instance deemed correct to a cluster with incorrect instances (or vice-versa). The training data is assigned to the cluster that *minimizes* this objective function. This predisposes the SSC algorithm to find high similarity clusters that have either predominately correct or incorrect instances. Note that clusters with predominately incorrect labels are guaranteed to contain *multiple labels* since a single-labeled cluster would be trivial for the model.

Second, the BoU needs to be able to evaluate each cluster to decide whether the model for that cluster needs improvements. Here we propose using a localized estimate of model performance to decide whether the model needs improvement to accommodate troublesome data. This estimate can be expressed as:

$$est(\pi_c) = \sum_{x_i \in \pi_c} [l_i = \text{correct}] \,/\, |\pi_c| \quad (2)$$

where π_c is the cluster under consideration, x_i is the cluster member, and l_i refers to the model prediction.

Third, we decide whether each BoU cluster is *correct* or *incorrect*. The decision making strategy here is to make use of a specified confidence interval to identify correct clusters where improvement is not needed:

$$type(\pi_c) = \begin{cases} \text{correct } if\, est(\pi_c) \geq 1 - \delta \\ \text{incorrect } otherwise \end{cases} \quad (3)$$

where est is the localized estimate (Eq. 2) and δ is the purity threshold parameter for the confidence interval. Eq. 3 is based on work in Dasgupta & Hsu [5] where clusters are evaluated using confidence intervals on the correctly-labeled member data to decide whether to request further labels for the member data.

Finally, we also create a hierarchy of BoU clusters. The BoU uses a hierarchical, top-down approach that iteratively (1) splits the data into clusters and identifies the correct and incorrect clusters, and (2) selectively improves the models on incorrect clusters. Specifically, at each layer of the cluster dendrogram, the SSC algorithm previously described splits the data into BoU clusters (Eq. 1). Next, each cluster from the split is assigned a type (Eq. 3) based on the localized estimate (Eq. 2). If the cluster is deemed incorrect, the specified improvement algorithm (cf., Section 2.2) runs using only the member data in that cluster and the model is relearned on that refined data. Correct clusters skip both steps and instead inherit the model from their parent cluster. After BoU algorithm stops splitting, the clusters and relearned models at the leaves of the dendrogram are used to make up the ensemble that predicts the labels for new data. The leaves are used so that each training instance belongs to only a single cluster thus avoiding confusion on cluster membership. Ultimately, this ensemble predicts the labels for a new instance by selecting the cluster containing the most similar instances and then using the model associated with that cluster to predict the label.

An example of this top-down approach is given in Figure 2 with the type, improvement, and model for each cluster. In this example, we use FS as the model improvement algorithm that gives a set of relevant features (f). This figure shows how the clusters are split and the models are inherited or relearned from one layer to the next. The original cluster starts with the model learned on all the data. After the improvement ($f1$) and relearn steps ($m1$), this cluster is split into correct and incorrect clusters. *The correct cluster uses the model from the parent*, while the incorrect cluster goes through model improvement and relearning before being split again. As shown in the last split, one child can retain the parent model ($m3$) while the other goes through the improvement and relearn steps.

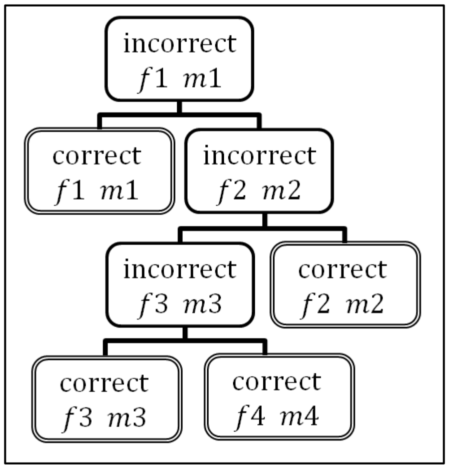


Figure 2. Cluster Splits from our BoU MR along with Type, FS Improvement (f), and Model (m). The double width borders denote the final set of clusters used.

3.2 BoU MR Algorithm

Here we present the complete BoU MR algorithm, as shown in Figure 3. Before we begin, there are two general guidelines we adopt to decide when to stop splitting the clusters. First, to avoid breaking up data in a single distribution, we stop splitting when a correct cluster is found with a high purity in terms of correctly-labeled instances (i.e., *purity* stop). Second, to support meaningful analysis, we stop splitting when clusters lack sufficient coverage (based on the percentage of the training data they contain) to learn a detailed model (i.e., *coverage* stop).

Our algorithm starts with a single cluster with all the training data and a model trained on that data. The algorithm runs recursively to create the dendrogram of BoU clusters. First, this algorithm uses Eq. 3 to compute the type for the BoU cluster (line 1). If the type is incorrect, the specified improvement algorithm is used on that BoU cluster's member data (cf., Section 2.2) and its model is relearned (lines 2-3). Subsequently, the cluster's type is updated based on the relearned model (line 4). If a cluster's type is now correct, then there is a purity stop and the algorithm returns the BoU cluster and its relearned model. Otherwise, the algorithm splits the training data into two new BoU clusters using the SSC algorithm previously discussed (Eq. 1). If both the new clusters meet the minimum coverage requirement (line 6), containing a percentage of the training data above the threshold φ, then the algorithm runs recursively on the two new BoU clusters with the parent's model. Otherwise, there is a coverage stop due to insufficient instances and the parent cluster/model is returned.

The BoU MR algorithm runs in polynomial time based on the number of recorded sessions and, as such, runs fast even as the number of LOs increases. The actual time complexity is dependent on the clustering algorithm: *O(IDF)* where *I* is the max iterations, *D* is the number of session instances, and *F* is the number of features. Further, the actual BoU clusters and the SL models can be computed offline to accommodate thousands of LOs. The real-time analysis only consists of mapping the new session to the BoU cluster based on current student interactions with the LO. This can be done very quickly as the number of clusters is much less than the total number of recorded sessions.

Some readers may argue that, by looking at the way the clusters are identified hierarchically, we are actually introducing overfitting on the data when creating the BoU clusters. But recall that the BoU MR algorithm is designed to prevent the labels from having too much influence on the BoU clusters: its clusters are created based on both feature similarity and labels, and not just labels alone. Additionally, the coverage stop also helps in this regard by acting as a regularizer and rejecting small clusters.

```
$C = Single\ cluster\ with\ all\ the\ training\ data$
$m = Model\ trained\ on\ C$
$I = Model\ improvement\ algorithm$
$S = Supervised\ learning\ system$
$\boldsymbol{function}\ BoUClustering(C, m)\ \boldsymbol{returns}\ C'\ and\ m'$
(1)  $\boldsymbol{if}\ type(C) == incorrect$ // check purity stop
(2)      $C' \leftarrow improve(C, I)$
(3)      $m' \leftarrow relearn(C', S)$
(4)      $\boldsymbol{if}\ type(C') == incorrect$ // check purity stop
(5)          $C_1, C_2 \leftarrow SSC(C', 2)$ // split the cluster
(6)          $\boldsymbol{if}\ coverage(C_1) > \varphi\ \boldsymbol{and}$
                 $coverage(C_2) > \varphi$ // check coverage stop
(7)              $BoUClustering(C_1, m')$
(8)              $BoUClustering(C_2, m')$
(9)          $\boldsymbol{end\ if}$
(10)     $\boldsymbol{end\ if}$
(11) $\boldsymbol{else}$
(12)     $C' \leftarrow C$
(13)     $m' \leftarrow m$
```

Figure 3. BoU MR Algorithm.

4. IMPLEMENTATION AND RESULTS

Here we start by describing the experimental setup including the learning object (LO) datasets. In Section 4.1, we provide results demonstrating the effectiveness of using the BoU to learn multiple models on the LO datasets (Objective 1). In Section 4.2, we provide results for using the BoU to improve existing models with feature selection and noise correction algorithms (Objective 2).

First, we use three widely studied SL systems in the experiments below: artificial neural networks (ANNs), support vector machines (SVMs), and decision trees (DTs). We use the Java implementations for all three from the Weka library with the parameters values suggested in Witten et al. [19]. The BoU MR algorithm uses a Java implementation for the semi-supervised clustering algorithm based on Kulis et al. [9]. We use of 0.1 for the purity threshold (δ) and a 0.1 for the coverage threshold (φ) both fine-tuned based on empirical results. Additionally, this clustering algorithm normalizes the student interactions features to the same range before creating the clusters. For the model improvement algorithms, we use a Java implementation of Lasso feature selection based on the R glmnet package [7]. We use the C++ implementation of LSVM noise correction from the FaLKM-lib package [17]. We use the values suggested in Segata [17] for the numerous parameters for LSVM.

Second, the intelligent learning object guide (iLOG) LO datasets used in the experiments are based on a three year deployment of 16 learning objects to introductory CS courses at the University of Nebraska, Lincoln [11]. During this deployment, there were over 5000 separate sessions between students and LOs. Large amounts of data were collected including (1) student interactions with the LOs during the tutorial, exercise, and assessment components (e.g., time spent on a page), (2) student demographic data (e.g., gender), (3) scores on the CS placement test [12], and (4) survey responses to both MSLQ and evaluation Likert surveys. The iLOG datasets distill the data collected into the instances, features, and labels necessary for supervised learning. Each instance represents a student-LO session with the features summarized in Table 1. The label for an instance is whether or not students passed the LO assessment component (i.e., if a student achieves $\geq$ 70% then she passes, otherwise she fails). As shown in Table 1, there are relatively few changes in the features collected from one year to the next. The increase in instances is the result of deployment to a larger number of courses.

We use the iLOG datasets in the experiments because they exemplify the aforementioned problems with SL systems in the following manner. First, the LOs were deployed to students in introductory CS courses with highly diverse backgrounds (e.g., CS majors, nonmajors, etc.) resulting in *multiple distributions* in the resulting datasets. Second, the LOs were deployed online using the Moodle Learning Management System and students were required to take the LOs and part of their course grades (5%). The difficulty of writing LO content suitable for all students and the online deployment, taken together, resulted in both *irrelevant features and noise* in the data tracked. For example, students trying to achieve high assessment scores without spending time on the tutorial content. Thus, these datasets are prime candidates for using FS and NC.

Table 1. Summary of the iLOG datasets in the experiments.

Features	**iLOG 2008**	**iLOG 2009**	**iLOG 2010**
Metadata	5	5	5
Tutorial	10	10	10
Exercises	20	20	16
Assessment	10	10	10
Student Demo	9	9	9
Placement	16	16	16
MSLQ	47	45	50
Evaluation	10	9	9
Total	127	124	125
Instances	**iLOG 2008**	**iLOG 2009**	**iLOG 2010**
Fail	426	738	1228
Pass	604	1131	2215
Total	1030	1869	3443

Finally, the experiments below compare the single model with the multiple models from the BoU. For all experiments, we provide both the test and F1 accuracy results based on ten-fold cross validation. In Section 4.1, we compare a single model to the BoU models learned using three SL systems (ANN, SVM, and DTs). In Section 4.2, we compare a single model to the BoU models after refining the data using FS and NC. This results in six (FS or NC × ANN, SVM, or DT) configurations.

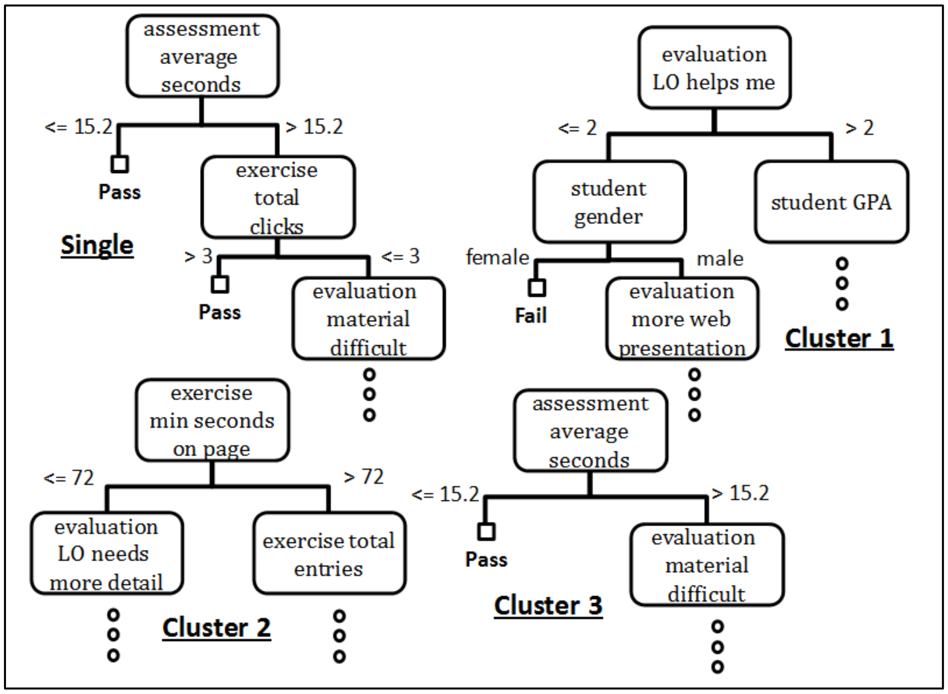
4.1 Multiple Model Investigation

Table 2 provides the test and F1 accuracy, on the iLOG datasets, for single model and the BoU multiple models. The BoU multiple models provide higher test and F1 accuracy than a single model on all three iLOG datasets. These results are reasonable given that the BoU can break up the iLOG dataset to better accommodate data from multiple distributions, for example, LOs deployed to different courses, students with different majors, etc.

To probe further into how the BoU multiple models accommodate the iLOG data, Figure 4 provides the actual decision trees on the iLOG 2008 dataset created using a single model and multiple models based on three clusters. (The trees for the iLOG 2009 and 2010 datasets are similar.) As shown in Figure 4, the trees learned on Clusters 1-2 have a very different idea of what features are the most important for predicting the labels (i.e., at the root) than the a single tree learned on all the data. By using these diverse trees—i.e., models, the BoU MR can better model the separate distributions in the iLOG datasets than forcing a single model to accommodate all the data. At the same time, the BoU clusters contain sufficient data to allow for fully expressive trees on the iLOG dataset. The proof of this is that Cluster 3 actually learns the same tree as the single model. Taken together, the capability to find diverse trees while retaining the same tree as the single model helps explain the BoU MR benefits to the test and F1 accuracy results. Additionally, BoU models learned on the local data provide more specific and detailed analyses than using a single model on all the data. These analyses can uncover very interesting connections in the data that would otherwise be hopelessly buried in the single model such as that between evaluation survey questions and gender in Cluster 1. Overall, these results help establish the effectiveness of BoU multiple models for LO analysis.

Table 2. Test and F1 accuracy for a single model and BoU multiple models. Grey cells indicate higher test accuracy while (*) indicates significantly higher accuracy (*t*-test, $p <=$ 0.05). The average number of BoU clusters is also given.

Dataset	Test Accuracy		F1 Accuracy		Ave. #
iLOG 2008	Single	BoU	Single	BoU	Clusters
ANN	0.69	0.74*	0.63	0.67	1.90
SVM	0.68	0.73*	0.64	0.69*	2.10
DT	0.72	0.73	0.68	0.68	4.60
iLOG 2009	Single	BoU	Single	BoU	Clusters
ANN	0.69	0.74*	0.60	0.67*	1.90
SVM	0.67	0.69*	0.58	0.61*	2.00
DT	0.62	0.65	0.52	0.56	2.00
iLOG 2010	Single	BoU	Single	BoU	Clusters
ANN	0.70	0.72*	0.56	0.61	3.20
SVM	0.69	0.71	0.57	0.62*	3.60
DT	0.65	0.67*	0.50	0.51	2.30



assessment average seconds
<= 15.2
> 15.2
Pass
exercise total clicks
Single
> 3
<= 3
Pass
evaluation material difficult
evaluation LO helps me
<= 2
> 2
student gender
student GPA
female
male
Fail
evaluation more web presentation
Cluster 1
exercise min seconds on page
<= 72
> 72
evaluation LO needs more detail
exercise total entries
Cluster 2
assessment average seconds
<= 15.2
> 15.2
Pass
evaluation material difficult
Cluster 3

Figure 4. Decision trees on the iLOG 2008 dataset created using a single model and multiple models based on BoU clusters.

On the other hand, as shown in Table 2, the accuracy (even for BoU multiple models) is relatively low on all three iLOG datasets (e.g., test accuracy in the 60s for DTs). As alluded to earlier, these datasets contain both irrelevant features and noise both of which are problematic for SL systems in general. In the next section, we show how the BoU MR can use feature selection and noise correction algorithms to break through this ceiling and improve both test and F1 accuracy.

4.2 Model Improvement Investigation

4.2.1 Lasso Feature Selection

Table 3 provides the test and F1 accuracy for the single model and the BoU multiple models both improved using the Lasso feature selection on the iLOG datasets.

Table 3. Test and F1 accuracy for a single model and BoU multiple models both using Lasso feature Selection (FS). Grey cells indicate higher test accuracy while (*) indicates significantly higher accuracy (*t*-test, *p* <= 0.05). The average number of BoU clusters is also given.

Dataset	Test Accuracy		F1 Accuracy		Ave. #
iLOG 2008	Single	BoU	Single	BoU	Clusters
ANN+FS	0.72	0.75*	0.65	0.69*	2.00
SVM+FS	0.72	0.74*	0.66	0.70*	2.00
DT+FS	0.74	0.75	0.70	0.71	2.43
iLOG 2009	Single	BoU	Single	BoU	Clusters
ANN+FS	0.71	0.76*	0.63	0.68*	2.00
SVM+FS	0.71	0.70	0.63	0.62	2.00
DT+FS	0.66	0.69*	0.55	0.60*	2.00
iLOG 2010	Single	BoU	Single	BoU	Clusters
ANN+FS	0.73	0.74	0.58	0.60	2.80
SVM+FS	0.72	0.72	0.61	0.61	3.30
DT+FS	0.67	0.70*	0.52	0.55	2.30

First, we observe that using Lasso on the training data allows nearly across-the-board increases in test and F1 accuracy for both single and BoU multiple models. Additionally, the increases in accuracy reported between Tables 2 and 3 are generally statistically significant (*t*-test, *p* <= 0.05).

To explain, the Lasso feature selection identifies and removes irrelevant features from the training data. Since these features are unimportant to the actual label, had they been incorporated into the model by the SL system, they would tend to confuse and distort the model lowering predictive accuracy and making the model less useful for analysis of student learning outcomes.

Second, using Lasso, we observe that the BoU multiple models still provide generally higher test and F1 accuracy than does a single model on all three iLOG datasets. As previously discussed the BoU retains the capability to break up the iLOG datasets and learn a separate model on each distribution. The BoU also has the capability to further improve and "fine-tune" these models using Lasso selectively only on the clusters that are deemed to contain troublesome data.

To probe further into how the BoU uses Lasso selectively to improve the models, Table 4 provides an example on the iLOG 2009 dataset of the number of relevant features used for the single model and separately for the models in BoU clusters. (The results for the iLOG 2008 and 2010 datasets are similar). As shown in Table 4, for a single model, Lasso removes almost half the features belonging to the exercise and MSLQ categories while mostly retaining features in the other categories. Lasso on BoU clusters gives more diverse results on the features removed. Lasso on cluster 1 removes additional features from the tutorial and assessment categories compared to that for the single model. Next, Lasso on cluster 2 removes a similar number of features as Lasso for the single model. Lastly, the BoU MR does not use Lasso at all on cluster 3 as this cluster is deemed free of troublesome data. Our algorithm prevents Lasso from removing locally relevant features just because they are irrelevant on the rest of the training data. This, in turn, prevents the model from being distorted by removing relevant features necessary for predicting the learning outcome for this cluster. Taken together, *the* *capability to use Lasso selectively*—thus allowing the relevant features to be customized for each cluster—helps explain the BoU benefits to the test and F1 accuracy results. Additionally, Lasso used separately for models provides additional insights that do not show up when Lasso is used for a single model; for example, suggesting that the students with sessions in cluster 1 are getting less out of the LO tutorial and assessment. Overall, these results help establish that BoU can further improve the multiple models for LO analysis using Lasso.

Note that the capability to use Lasso feature selection to improve models is important for educational data mining (EDM) in general since EDM datasets often contain numerous irrelevant features. However, these datasets are also often highly diverse forcing Lasso to make difficult decisions to retain features as relevant that are irrelevant in many areas or remove features because they are only relevant to a minority of the training data. The advantage of using BoU is that its clusters allow for multiple, expressive models. Recall the discussion for Table 4 where Lasso found very different feature vectors when used on different clusters. Combined with the identical decision trees previously discussed (cf., Figure 4), this supports our claim that BoU clusters contain sufficient data to allow for more effective utilization of Lasso separately on the data in each cluster.

Table 4. Number of relevant features selected by Lasso feature run on all the training data (Single) and run separately on the three BoU clusters (C1-C3). The total number of features is also included for reference (2009).

Features	**2009**	**Single**	**C1**	**C2**	**C3**
LO Data	5	5	5	4	5
Tutorial	10	7	3	8	10
Exercises	20	11	9	11	20
Assessment	10	7	6	9	10
Student Demo	9	9	6	6	9
Placement	16	13	9	10	16
MSLQ	45	22	19	24	45
Evaluation	9	9	6	6	9
Total	124	83	63	78	124

4.2.2 LSVM Noise Correction

Table 5 provides the test and F1 accuracy results, on the iLOG datasets, for the single model and the BoU multiple models both using the LSVM noise correction.

Table 5. Test and F1 accuracy for a single model and BoU multiple models both using LSVM noise correction (NC). Grey cells indicate higher test accuracy while (*) indicates significantly higher accuracy (*t*-test, *p* <= 0.05). The average number of BoU clusters is also given.

Dataset	Test Accuracy		F1 Accuracy		Ave. #
iLOG 2008	Single	BoU	Single	BoU	Clusters
ANN+NC	0.73	0.75*	0.65	0.69	2.00
SVM+NC	0.72	0.74	0.68	0.70*	4.10
DT+NC	0.73	0.75*	0.68	0.71	4.00
iLOG 2009	Single	BoU	Single	BoU	Clusters
ANN+NC	0.72	0.75*	0.56	0.67*	2.00
SVM+NC	0.70	0.70	0.50	0.62*	2.00
DT+NC	0.64	0.69*	0.36	0.59*	2.00
iLOG 2010	Single	BoU	Single	BoU	Clusters
ANN+NC	0.71	0.75*	0.52	0.61*	4.10
SVM+NC	0.72	0.70	0.49	0.60	3.50
DT+NC	0.67	0.70*	0.36	0.54*	2.50

First, as with Lasso, using LSVM noise correction allows nearly across-the-board improvements in accuracy. To explain, LSVM improves the model by flagging potentially noisy instances whose labels (i.e., pass/fail) do not match those in nearby instances with similar features. These noisy instances would otherwise distort the model since they provide contradictory labels and, thus, lower predictive accuracy.

Second, using LSVM, on all three datasets the BoU multiple models provide generally higher test and F1 accuracy than a single model. Again, the BoU can "fine-tune" these models using LSVM selectively only on the clusters with troublesome data. Now, as a whole, the total number of instances flagged as noisy by LSVM was comparable when used on the all training data and selectively on the BoU clusters. However, LSVM for multiple models had two advantages. First, by breaking up the training data, the BoU MR simplified the task of distinguishing between instances that actually have noisy labels and those in close proximity to instances with truly different labels. Second, by using LSVM selectively, *our algorithm was able to use LSVM aggressively on clusters where the model was struggling*, such as clusters containing sessions where students tried to game the system, *without worrying about damaging clusters where the model was already doing well by removing training data mistakenly deemed noisy*. Overall, these results help establish that BoU MR can further improve the multiple models for LO analysis using LSVM.

Once again, using LSVM to improve models is important on EDM datasets that contain relatively large amounts of noise. However, LSVM used only once on a highly diverse datasets may struggle to find concentrations of label noise and could end up—in its pursuit of noise to correct—flagging data near the decision boundary as noisy. Again, BoU MR helps in this regard by allowing for more effective LSVM focusing on the clusters containing large amounts of noise.

5. CONCLUSIONS AND FUTURE WORK

Supervised learning (SL) systems have been used for the analysis of student interactions with learning objects (LOs). SL systems learn a model from previously collected training data that can accurately predict the labels for new data assuming that the training data comes from a single distribution and is relatively clean. Unfortunately, LO data is highly diverse from multiple distributions and likely to contain noise which can limit the effectiveness of single models. Learning multiple models and improving models on troublesome training data is intuitively a good solution. However, identifying areas that capture only data from a single distribution without fitting the noise is far from trivial. We propose a new BoU meta-reasoning (MR) algorithm that starts by breaking up the data into clusters designed to separate troublesome data (incorrect cluster) from data where the model is doing well (correct cluster). Our algorithm then uses a separate model for each cluster. On the incorrect clusters, our algorithm takes steps to improve the model by using feature selection or noise correction on the training data before relearning the model. We have shown empirically on three LO datasets that the BoU multiple models allow for higher predictive accuracy than single models. Furthermore, we have shown that the BoU MR can further improve the models for LO analysis using feature selection and noise correction algorithms. Our results also suggest that BoU MR may also be able improve feature selection and noise correction algorithms—both important for educational data mining. Feature selection used separately on the BoU clusters makes it easier to identify features that are relevant only on certain areas. Further, noise correction focused on the clusters makes it easier to remove noise instead of accidently disrupting the decision boundary.

In this paper, we have established effectiveness of using the BoU multiple models. In the future, we intend to advance our BoU MR down two different lines of research. First, we intend to further investigate how the BoU multiple models can provide independent, high-level guidelines on which type of student interactions are associated with the learning outcomes. We have already taken the first step by showing that multiple models allow for diverse decision trees. The next step is the analysis of the ANN and SVM models using rules extraction, sensitivity analysis, or inverse classification techniques [3]. This research has strong pedagogical implications for students. Such analysis could, in real-time, inform students about the probable success/failure of their study strategies, for example, warning a student during the LO that she may not be spending enough time on the tutorial section. Second, we intend to expand the use of BoU models to other educational data mining (EDM) areas such as intelligent tutoring systems (ITS) and virtual learning platforms (VLP). These areas share some of the same properties that BoU is designed to address (e.g., noise) but have additional properties not found in LOs (e.g., "bags" of instances in VLP). To this end, we intend to evaluate how the BoU models stack up against previous work on learning accurate models in these areas (e.g., Rajibussalim [14] for ITS; Zafra & Ventura [20] for VLP).

6. REFERENCES

[1] Alharbi, A., et al. 2011. An Investigation into the Learning Styles and Self-Regulated Strategies for Computer Science Students. *ASCILITE*, 36-46.

[2] Balatsouka, P., Morris, A., O'Brien, A. 2008. Learning Objects Update: Review and Critical Approach to Content Aggregation. *Educational Technology & Society*, *11* (2), 119-130.

[3] Barbella, D., et al. 2009. Understanding Support Vector Machine Classifications via a Recommender System-Like Approach. *DMIN*, 305-311.

[4] Bernardini, A., Conati, C. 2010. Discovering and Recognizing Student Interaction patterns in Exploratory Learning Environments. *ITS*, 125-134.

[5] Dasgupta, S., Hsu, D. 2008. Hierarchical Sampling for Active Learning. *ICML*, 208-215, 2008.

[6] de Raadt, M., Simon. 2011. My Students Don't Learn the Way I Do. *ACE*, *114*. 105-112.

[7] Freidman, J., Hastie, T., Tibshirani, R. 2010. Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*, *33* (1), 1-22.

[8] Kiu, C-C. Lee, C-S. 2007. Learning Object Reusability and Retrieval through Ontological Sharing: A Hybrid Unsupervised Data Mining Approach. *ICALT*, 548-550.

[9] Kulis, B., Basu, S., Dhillon, I., Mooney, R. 2009. Semi-Supervised Graph Clustering: A Kernel Approach. *Machine Learning*, *74* (1), 1-22.

[10] Liu, H., Yu, L. 2005. Towards Integrating Feature Selection Algorithms for Classification and Clustering. *IEEE Trans. On Knowledge and Data Engineering*, *17* (4), 491-502.

[11] Miller, L. et al. 2011. Evaluating the Use of Learning Objects in CS1. *SIGCSE*, 57-62.

[12] Nugent, G., Soh, L-K., Samal, A., Lang, J. 2006. A Placement Test for Computer Science: Design, Implementation, and Analysis. *Computer Science Education*, *16* (1), 19-36.

[13] Pechenizkiy, M., et al. 2006. Class Noise and Supervised Learning in Medical Domains: The effect of feature extraction, *IEEE CBMS*, 708-113.

[14] Rajibussalim. 2010. Mining Students' Interaction Data from a System that Support Learning by Reflection. *EDM*, 249-256.

[15] Romero, C., Ventura, S., Espejo, P., Hervas, C. 2008. Data Mining Algorithms to Classify Students. *EDM*, 187-191.

[16] Romero, C., Ventura, S. 2010. Educational Data Mining; A Review of the State of the Art. *IEEE Trans. On Systems, Man, and Cybernetics*, *40* (6), 601-618.

[17] Segata, N. 2009. FaLKM-lib v1.0: A Library for Fast Local Kernel Machines. University of Trento, Italy.

[18] Segata, N., Blanzieri, E. Fast and Scalable Local Kernel Machines. 2010. *JMLR*, 1883-1926.

[19] Witten, I., Frank, E., Hall, M. 2011. *Data Mining: Practical Machine Learning Tools and Techniques*. Elsevier.

[20] Zafra, A., Ventura, S. 2009. Predicting Student Grades in Learning Management Systems with Multiple Instance Genetic Programming. *EDM*, 309-318.

Adapting Bayesian Knowledge Tracing to a Massive Open Online Course in edX

Zachary A. Pardos, Yoav Bergner, Daniel T. Seaton, David E. Pritchard
Massachusetts Institute of Technology
77 Massachusetts Ave.
Cambridge, MA 02139
{pardos, bergner, dseaton, dpritch}@mit.edu

ABSTRACT

Massive Open Online Courses (MOOCs) are an increasingly pervasive newcomer to the virtual landscape of higher-education, delivering a wide variety of topics in science, engineering, and the humanities. However, while technological innovation is enabling unprecedented open access to high quality educational material, these systems generally inherit similar homework, exams, and instructional resources to that of their classroom counterparts and currently lack an underlying model with which to talk about learning. In this paper we will show how existing learner modeling techniques based on Bayesian Knowledge Tracing can be adapted to the inaugural course, 6.002x: circuit design, on the edX MOOC platform. We identify three distinct challenges to modeling MOOC data and provide predictive evaluations of the respective modeling approach to each challenge. The challenges identified are; lack of an explicit knowledge component model, allowance for unpenalized multiple problem attempts, and multiple pathways through the system that allow for learning influences outside of the current assessment.

Keywords

Probabilistic Graphical Models, Bayesian Knowledge Tracing, MOOC, Resource model, edX

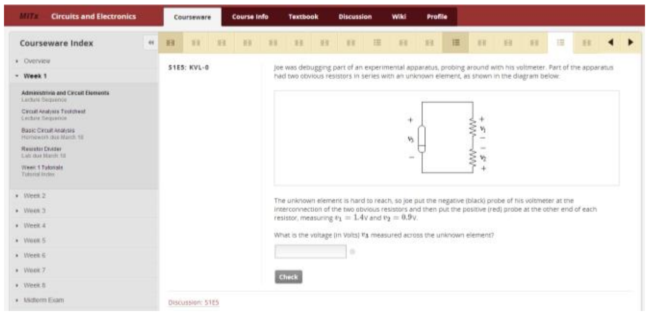
1. INTRODUCTION

Massive Open Online Courses (MOOCs) are a quickly emerging modality of learning in higher-education. They consist of various learning resources, often lecture videos, etexts, online office hours, assessments which include homework and exams, and have a specific time in which they begin and end, often corresponding closely to that of their residentially offered counter-parts. While the efficacy of MOOCs compared to their residential offerings is an open question; from the viewpoint of educational research, MOOCs provide several substantial advantages, most notably the detailed digital trail left by students in the form of log data and the size of the student cohorts, which are often several orders of magnitude larger than typical on-campus-only offerings.

Unlike Intelligent Tutoring Systems (ITS), MOOCs do not currently provide tutorial help on demand at the points of need; instead, the knowledge is self-sought and supplied by a redundancy of information across various types of resources resulting in a variety of student selected resources and pathways through the system. This rich data provided by MOOCs presents an opportunity to investigate the efficacy of student behavior under varying conditions; however, MOOCs currently lack a model of learning with which to instrument this exploration. In this paper we will show how existing learner modeling techniques based on Bayesian Knowledge Tracing can be adapted to the inaugural course, 6.002x: circuit design, on the edX MOOC platform. We identify three distinct challenges to modeling MOOC data in section 2, followed by a description of our evaluation methodologies in section 3, and finally results of the predictive evaluations of the respective modeling approach to each challenge in section 4.

1.1 Anatomy of the MOOC

The inaugural course on the edX platform, 6.002x (Spring 2012), was a 14 week-long online course featuring video lectures in weekly sequences interspersed with lecture problems, an online textbook, a discussion forum, and a course wiki. The web interface for the course is shown in Figure 1. While the sequence of videos and problems is suggested in the form of a timeline at the top of the interface, the student can take any path through the material they choose including skipping or revisiting content.



Joe was debugging part of an experimental apparatus, probing around with his voltmeter. Part of the apparatus had two obvious resistors in series with an unknown element, as shown in the diagram below:

The unknown element is hard to reach, so Joe put the negative (black) probe of his voltmeter at the interconnection of the two obvious resistors and then put the positive (red) probe at the other end of each resistor, measuring $v_1 = 1.4$V and $v_2 = 0.9$V.

What is the voltage (in Volts) v_3 measured across the unknown element?

Figure 1. The interface for the 6.002x MOOC on edX. This screenshot shows a student answering a problem that is part of the Week 1 lecture sequence.

Student grades were based on 12 homework assignments and 12 virtual labs (weighted 15% for each category, with unlimited answer attempts allowed), a midterm and a final exam (30% and 40% respectively, with 3 attempts allowed). Although lecture problems did not count towards the grade, they were still marked correct and incorrect, with instant feedback as given on the homeworks. There were 289 scored elements (i.e. counting problem subparts) in 104 lecture sequence problems, 197 in 37 homework problems, 26 in 5 midterm problems and 47 in 10 final exam problems. The homework interface and scoring mechanism had some nuances that deserve elaboration.

Weekly homework assignments consisted of several problems which were all displayed on a single web page. A typical problem consisted of a figure plus several answer field "subparts" that prompted the user for input. Correctness feedback would be shown to the right of the answer fields in the form of a red "X" for incorrect (or blank answers) and a green checkmark for correct answers. This feedback was displayed after the student clicked the problem's "check" button, which simultaneously checked all answer fields within the problem. Students could answer the subparts in any order they chose however several problems' subparts required the incorporation of answers from a previous subpart. If a student answered all the subparts before their first "check", the order in which she answered the subparts was not known, however many students elected to click the check button after each consecutive answer. Unlike most ITSs, homework was scored based on the last answer entered by the user instead of the first.

1.2 Dataset

The course drew 154,000 registrants, however; only 108,000 entered the course with around 10,000 completing the course through the final. Among those, 7,158 received a certificate for having earned at least a 60% weighted average. Our dataset consisted of 2,000 randomly chosen students from the certificate earners. A further reduction of the dataset was made by randomly selecting ten problems (and their subparts) from each of the three types of assessments; homework, lecture sequence, and exam problems.

The data for this course originated from JSON log files produced on the Amazon EC2 cloud, where the edX platform is hosted. The original log files were separated out into individual user files and the JSON records were parsed into a human readable time series description of user interaction with components of the MOOC. The final data preparation step compiled an event log by problem, consisting of one line per student event relevant to that problem. This included time spent on the event, correctness of each subpart, when the student entered or changed an answer, the attempt count of that answer, and resources accessed by the student before and between responses. An example of this data format is shown in Table 1.

Table 1. Example of the event log format of our distilled dataset

User	Res	Time	Resp1	Resp2	Count1	Count2
9	video	2m 30s	- -	- -	- -	- -
9	answer	10m 5s	correct	correct	1	1
10	book	4m 41s	- -	- -	- -	- -
10	book	40s	- -	- -	- -	- -
10	answer	20s	incorr.	- -	1	- -
10	answer	15s	incorr.	- -	2	- -
10	answer	1m 8s	incorr.	incorr.	3	1
10	answer	28s	- -	correct	- -	2
10	video	2m 10s	- -	- -	- -	- -
10	answer	6s	correct	- -	4	- -

1.3 Bayesian Knowledge Tracing

Knowledge Tracing (KT) [1] comes from the motivation to implement mastery learning [19], where every student is allowed to learn skills at his or her own pace and does not continue on to more complex material until mastery of pre-requisites has been achieved. It is based on a simplification of the ACT-R theory of skill acquisition [2] and is tasked with making this inference of mastery in the Cognitive Tutors, among other ITS. To achieve this end, simpler mastery criterion exist such as N-correct in a row to master, which is used by the ASSISTments Platform in their skill builder problem sets [3] and in the Khan Academy tutor where the term proficiency is used instead of mastery [4]. In a Cognitive Tutor, acquirable knowledge, whether declarative or procedural, is defined by fine-grained atomic pieces called Knowledge Components (KCs), typically defined by a subject matter expert. Answer steps in the tutor are tagged with these KCs and a student's past history of responses indicates his or her level of mastery of the KC. In this context, mastery is inferred to have occurred when there is a high probability (usually >= 0.95) that the KC is known by the student.

The initial KT model was not introduced as a Bayesian model; however, its formulas were found [6] to be perfectly represented by a Dynamic Bayesian Network [20], which has become the standard representation referred to as Bayesian Knowledge Tracing (BKT). The standard BKT model is defined by four parameters; prior knowledge $p(L_o)$[1], probability of learning $p(T)$, probability of guessing $p(G)$, and probability of slipping $p(S)$. Based on these parameters, inference is made about the student's probability of knowledge at time opportunity n, $p(L_n)$. The parameters and inferred probability of knowledge can also be used to predict the correctness of a student response with:

$$p(Correct_n) = p(L_n) \cdot p(\neg S) + p(\neg L_n) \cdot p(G)$$

KCs vary in difficulty and amount of practice needed to master on average, so values for these parameters are KC dependent and can be fit to training data such as log data from a previous cohort of students. Parameter fitting is often accomplished using Expectation Maximization (EM) or a grid-search of the parameters that maximizes a loss function such as sum of squared residuals of the predicted probability of a correct answer and the observed correctness. Neither fitting procedure has proved consistently superior to the other [5, 21], however; grid-search, while faster at fitting the basic BKT model, grows exponentially with the number of parameters which is a concern for extensions to BKT with higher parameterization. With both methods of parameter fitting, the objective is to define parameters that result in a projection of performance that best matches the observed data, which is the students' temporal sequence of correct and incorrect responses to questions of a particular KC.

The use of Knowledge Tracing has two stages, the stage in which the four parameters are learned, and the stage where an individual student's knowledge is being inferred from their responses. During the inference stage, the probability of knowledge at time n, given an observation, is calculated from a student's response with the following when a correct response is observed:

$$p(L_n|Correct_n) = \frac{p(L_n) \cdot p(\neg S)}{p(L_n) \cdot p(\neg S) + p(\neg L_n) \cdot p(G)}$$

And with the following when an incorrect response is observed:

$$p(L_n|Incorrect_n) = \frac{p(L_n) \cdot p(S)}{p(L_n) \cdot p(S) + p(\neg L_n) \cdot p(\neg G)}$$

The $p(L_n)$ on the right side of the formula is the prior probability of knowledge at that time, while $p(L_n|Evidence_n)$ is the posterior probability of knowledge calculated after taking an observation at that time into account. Both formulas are applications of Bayes Theorem and calculate the likelihood that the explanation for the observed response is that the student knows the KC. Since the student will be presented with feedback, there is a chance to learn. The probability the student will learn the KC from the opportunity is captured by this formula which calculates the new prior after adding in the probability of learning:

$$p(L_n) = p(L_{n-1}|Evidence_{n-1}) + p(\neg L_{n-1}|Evidence_{n-1}) \\ \cdot p(T)$$

These formulas are used in the task of determining mastery, however; this model of knowledge has been extended to serve as a

[1] The name "P(L_o)" was used to denote the prior parameter in [1]. In a BKT model, this is symbolically equivalent to $p(L_1)$.

platform to study learning phenomenon [7, 8, 9]. It is this capacity for discovery that we aim to enable in MOOCs by adapting BKT approaches.

2. Model Adaptation Challenges

In order to build a foundation for measuring learning phenomena in the MOOC, several differences between MOOCs and Intelligent Tutoring Systems need to be addressed. The first is the lack of a subject matter expert mapping of the KCs associated with questions in the system. The second challenge is the attempt-until-correct scoring of the homework and lecture sequence problems. Lastly, we will address the open interface of the virtual learning environment which allows for users to take different pathways through the course which influences learning rates within a KC differently depending on path.

2.1 Lack of a KC model

The term "learning" can have broad meanings, however; in mastery contexts it is referred to with respect to a particular skill, or knowledge component being acquired. The mapping of these skills to questions, commonly referred to as a Q-matrix [10], as well as the enumeration of the skills, often comes from a subject matter expert. These skills have been referred to as cognitive operations in the psychometrics literature [11] and the processes of identification of skills is commonly referred to as cognitive task analysis in the context of ITS [12] and expert systems. Learning curves analysis [13], a KC mapping evaluation technique, asserts that evidence of a good skill mapping is a monotonically decreasing error rate across opportunities to answer questions within a skill. Similarly, fluency is expected to increase (decreasing time to solve) across correct answers to a particular skill. A unidimensional view of questions within a MOOC or a subject such as Geometry, for instance, would result in a noisy performance and fluency plot since error rates and response times would jump as soon as new topic material was introduced in the curriculum.

While subject matter expert defined knowledge components or learning objectives are planned for select future MOOC offerings, they are not common and do not exist in the 6.002x course data used in this paper. Therefore, our goal was to utilize elements of the course structure to inform a mapping of KCs to questions. We chose to leverage the problem and subpart structure of assignments, where the problem itself would serve as the KC and its subparts would be the questions belonging to the KC. The rationale for this choice was that the professor of the course often has a particular concept in mind that they wish to tap with each problem. Performance on the subparts is evidence of the student grasping this concept. The benefit to this type of mapping is that it is domain agnostic and can be used as a baseline KC model for any MOOC. The drawback is that it does not allow for longitudinal assessment of learning over more than one week since answers to a given KC will only occur within a problem in a particular week's assignment. Reduced model fit is another drawback as Corbett & Conrad [14] evaluated a similar superficial mapping of questions to course problem structure and found that this indeed sacrificed achieving more systematic, smother learning curves. Nevertheless, we believe this mapping is a reasonable start which allows for phenomenon to be studied within a problem (which we coin "problem analytics") and the methods and models described here can be applied with a different KC model swapped in, derived by a subject matter expert, inferred from the data, or a hybridization of the two [15].

2.1.1 Basic model definition

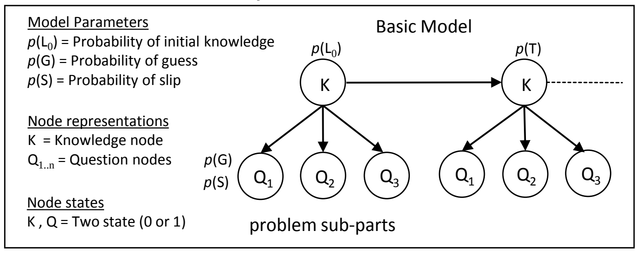

Model Parameters
$p(L_0)$ = Probability of initial knowledge
$p(G)$ = Probability of guess
$p(S)$ = Probability of slip
Node representations
K = Knowledge node
$Q_{1..n}$ = Question nodes
Node states
K , Q = Two state (0 or 1)
Basic Model
$p(L_0)$
$p(T)$
K
K
$p(G)$
$p(S)$
Q_1
Q_2
Q_3
Q_1
Q_2
Q_3
problem sub-parts

Figure 2. The basic model – a retrofit BKT model to capture answers to multiple questions in a single time slice and using homework problem as the KC. The number of parameters in this model is: **4**

Our most basic retrofitting of the BKT model to the MOOC is shown in Figure 2. In this model, which we will refer to as the "basic" model, the homework problem is the latent knowledge, K, and the observed questions are the subparts of the homework problem. When student knowledge is in the learned state this means the student has the knowledge required to answer all of the subparts. Whereas traditional application of BKT has only a single observed random variable causally linked to from the latent variable, in this model we had to accommodate for observation of multiple subpart observations at once. For example, these are the calculation steps for inferring the probability of knowledge at the second time slice when a student answers subparts one and two incorrectly on the first click of the problem check button and the third subpart correctly on the second click of the problem check button (leaving parts one and two unchanged).

First, the posterior is calculated given an incorrect answer to the first subpart: $p(L_1|Q_1 = Incorrect) = \frac{p(L_0)\cdot p(S)}{p(L_0)\cdot p(S) + p(\neg L_0)\cdot p(\neg G)}$

Next, the posterior is updated again given an incorrect answer to the second subpart:

$$p\big(L_1 \,\big|\, Q_1 = Incorrect \,\&\, Q_2 = Incorrect\,\big) = \frac{p(L_1 \,|\, Q_1{=}Incorrect\,)\cdot p(S)}{p(L_1 \,|\, Q_1{=}Incorrect\,)\cdot p(S) + p(\neg L_1 \,|\, Q_1{=}Incorrect\,)\cdot p(\neg G)}$$

Steps one and two are interchangeable, including when correct and incorrect responses are observed.

The prior for knowledge at the second time slice is then calculated by applying the probability of learning to the posterior:

$$p(L_2\,) = p\big(L_1 \,\big|\, Q_1 = Incorrect \,\&\, Q_2 = Incorrect\big) + p\big(\neg L_1 \,\big|\, Q_1 = Incorrect \,\&\, Q_2 = Incorrect\big) \cdot p(T)$$

Finally, the posterior probability of knowledge at the second time slice is calculated given the observation of a correct answer on the third question: $p(L_2|Q_3 = Correct) = \frac{p(L_2)\cdot p(\neg S)}{p(L_n)\cdot p(\neg S) + p(\neg L_2)\cdot p(G)}$

2.2 Multiple unpenalized answer attempts

The Cognitive Tutors allow for multiple answer attempts, as does the ASSISTments Platform, however; the scoring policy for those systems is to score only the first response to each question and students are aware of this policy. The assumption is therefore that the most informative response is the first response and in a standard application of BKT, only the first responses to questions are used to train and update the model. In the MOOC, three responses are allowed on the exam problems and unlimited responses on the homework and lecture sequence problems. The scoring policy for all problems is to score the last response. Since students are aware of this policy, it cannot be assumed that the most informative response is the first. For example, some students may decide to employ a quick heuristic on their first attempt instead of thinking through the problem as was observed among male users in an intro physics course [16]. Using only the last response is also problematic as these responses tend to have a very high percent correct, at least in homework and lecture problems, and a large amount of information would be lost in trying to model learning with only these responses. It is therefore an open and empirical research question as to where the most information exists in student answer attempts and so we define a model that allows the data to give us the answer. Past approaches have used regression to set BKT guess and slip parameters based on a host of contextual features [22], however these models, by admission, have not considered multiple attempts within a question.

Studies on test data where students are allowed multiple unpenalized attempts suggest that more information is contained in later responses (higher IRT discrimination) [17]. In addition to evaluating if a BKT model with attempt count information outperforms the basic BKT model in predictive accuracy, we also inspect the parameters of the model for each attempt count to observe if the trends seen in past studies reemerge in our data.

2.2.1 Count model definition

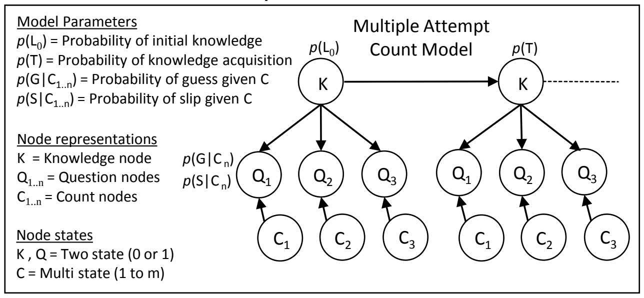

Model Parameters
$p(L_0)$ = Probability of initial knowledge
$p(T)$ = Probability of knowledge acquisition
$p(G|C_{1..n})$ = Probability of guess given C
$p(S|C_{1..n})$ = Probability of slip given C
Node representations
K = Knowledge node
$Q_{1..n}$ = Question nodes
$C_{1..n}$ = Count nodes
Node states
K , Q = Two state (0 or 1)
C = Multi state (1 to m)
Multiple Attempt Count Model
$p(L_0)$
$p(T)$
K
K
$p(G|C_n)$
$p(S|C_n)$
Q_1
Q_2
Q_3
C_1
C_2
C_3

Figure 3 The count mode – conditioning question guess and slip on answer attempt count to allow information gained from responses to vary. The number of parameters in this model is: $\mathbf{2+2\cdot(\#\ of\ attempt\ counts\ modeled,}\ \boldsymbol{m})$

The guess and slip parameters of the model dictate the amount of information gained about the latent variable from a correct or incorrect response; a guess and slip of zero in the Bayesian update calculation would mean that the value of the responses was 100% reflective of the binary state of the latent variable, while a guess and slip of 0.50 represents the maximum uncertainty regarding a response. Allowing for a different guess and slip parameter depending on attempt count therefore allows the model to capture a differing amount of information gained at each attempt. This is our modeling approach to multiple unpenalized attempts which we will refer to as the "count" model.

In the model, shown in Figure 3, count nodes, which are observable random variables, are added for every subpart since users can be on different attempt counts for different subparts. The size of the count nodes correspond to the number of attempt counts chosen to model. Inspection of the dataset showed that only ~4% of attempts were 5th attempts, therefore the size of the attempt count node was set to 6 which was also the count used for any attempt count over 6. This setting was fairly ad-hoc and could be improved upon by setting based on empirical evaluation. While the attempt count node contains a prior parameter, this was not counted as a free parameter but was instead fixed to the observed distribution of count attempts in the training data.

2.2.2 Allowing for difficulty/information gain to differ among subparts

Recent work has extended BKT to allow for different guess and slip parameters to be modeled per item in a model coined KT-IDEM (Item difficulty effect model) [3]. In ASSISTments, each problem template within a skill builder problem set was allowed to fit different guess and slip parameters, and in the Cognitive Tutor this was done at the level of the problem, where all steps of a given KC shared a guess/slip with one another within a problem but steps of the same KC that appeared in a different problem could fit different guess and slip parameter values. In both systems, prediction accuracy was improved by ~15% when there was ample data to fit each set of parameter (6 or more data points per parameter). This can be seen as allowing for variation in question difficulty among questions in a KC, or in the case of the Cognitive Tutor, allowing for variation in KC performance depending on problem context. It can also be interpreted as modulating the information gained about the latent variable depending on the question in much the same way as the count nodes in the count model modulate the information gained about the latent variable from responses depending on attempt count.

This item parameterization technique was applied to our MOOC data in the basic and count model, creating two additional models referred to as the "idem" and "idem count" model. In the idem model, the number of parameters increases to: $2+2\cdot(\#\ of\ subparts)$ and in the idem count model, increases to: $2+2\cdot(\#\ of\ subparts)\cdot(\#\ of\ attempt\ counts\ modeled)$.

2.3 Multiple pathways through the system

In many virtual learning environments, particularly in K-12, students complete one set of problems at a time and their path through the system is either fixed or the interface inhibits switching between problem sets. In the 6.002x MOOC, multiple problems are displayed on a page and students are frequently observed returning to a problem after answering another [18]. Besides learning from other problems, the redundancy of information found among the book pages, videos, wiki, and discussion board also allow the student to self-select his or her own path to acquiring the knowledge needed to complete the assignments. This means that influences on learning can come from a variety of sources in the learning environment, unlike most ITS where the learning can be assumed to come from feedback and tutorial help provided within the problem at hand. While this poses a challenge, in terms of capturing the variance in student learning, it also provides a rich trail of information and variety of pathways through the system that can be data mined and modeled.

Table 1 in section 1.2 illustrates how students can weave in and out of resources while answering assignment questions. It shows how one student answered a question correct after viewing a video and another student answered the same question incorrect until encountering a video, after which the hypothetical student answered the question correct. The consideration of resource influence on learning can be posed as a credit/blame inference problem, where, depending on problem answers and resource access in the aggregate, resources can be credited with learning or blamed for being ineffective. This model is at the very early stages of research and considers only resource type, which can be one of the following seven types: book, video, wiki, discussion, tutorial, answer to another problem, and answer to the problem at hand. Up to the last five resource access events before a response are used, earlier events are discarded.

2.3.1 Resource model definition

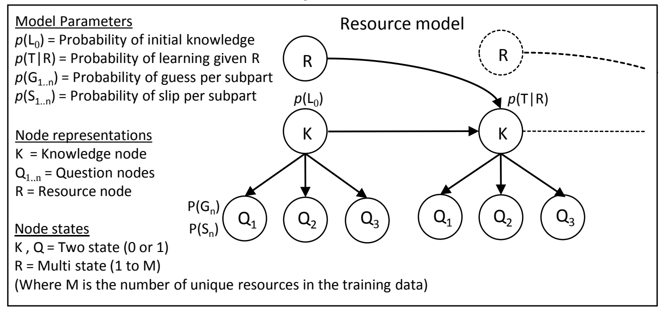

Model Parameters
$p(L_0)$ = Probability of initial knowledge
$p(T|R)$ = Probability of learning given R
$p(G_{1..n})$ = Probability of guess per subpart
$p(S_{1..n})$ = Probability of slip per subpart
Node representations
K = Knowledge node
$Q_{1..n}$ = Question nodes
R = Resource node
Node states
K , Q = Two state (0 or 1)
R = Multi state (1 to M)
(Where M is the number of unique resources in the training data)
Resource model
R
R
$p(L_0)$
$p(T|R)$
K
K
$P(G_n)$
$P(S_n)$
Q_1
Q_2
Q_3
Q_1
Q_2
Q_3

Figure 4. The resource model – based off of the idem model with resource access information added and hypothesized to influence learning. The number of parameters in this model is: $\mathbf{1+2}\cdot(\#\,\boldsymbol{of\ subparts})+(\#\,\boldsymbol{of\ resource\ types\ defined})$

The resource model was built from the idem model without attempt count taken into consideration. The model is the same except for the addition of the observable resource node, R, which conditions the learning parameter, $p(T|R)$. At each time slice, the observable, R, is given the value corresponding to the current resource type being accessed. This model generalizes the idem model and can be made mathematically identical by removing all resources types except for "answer to the problem at hand", which represents the standard learning parameter capturing the benefit of feedback. When a non-problem resource is accessed, the R node gets the value of that resource type and a time slice with no question answer input is used.

3. Training and Evaluation Methodology

All five models (basic, count, idem, idem count, and resource model) were evaluated with a 5-fold student level cross-validation where the 2,000 students and their respective data were randomly assigned to one of five bins. Models were trained on the data in four bins and predictions were made on the data of the students in the fifth bin. This training/testing procedure was repeated five times, such that each bin was used once as the testing set. This evaluation procedure was run for all models on the 10 lecture, homework, and exam problems sampled in the dataset with the exception of the resource model which was only run on the homework problems. The premise of a cross-validation is to investigate if the variance captured by the models generalizes to held out data. If it does, indicated by improved predictive performance over a simpler model, the assumption is that the variance captured by the more complex model is real and reproducible. Ideally, training can be done on a previous course cohort and tested on the data of a cohort from a subsequent offering of the course. In the absence of this kind of training/test data, student level cross-validation serves as a strong substitute.

3.1 Model training details

The models used Expectation Maximization (EM) to fit parameters to the training sets with the same set of ad-hoc initial parameter values used for all models: $p(L_o)$ = 0.20, $p(T)$ = 0.10, $p(G)$ = 0.10, $p(S)$ = 0.15. Due to the data being restricted to a limited number of computing resources at the time of evaluation, a low maximum EM iteration count of 5 was set to make the cross-fold evaluations tractable in the time period allotted. Each cross-validation fold for homework took on average 12.8 hours of compute time per model running on an Intel i5 2.6Ghz machine. The lowest compute time model was the basic model with 10.7 hours per fold and the highest was the resource model with 15.1 hours per fold. Lecture and exam problems took 1/10th the time to evaluate suggesting that more answer events occurred in the homework. For future runs, more tractable compute times could be sought with a more aggressive filtering of homework students with excessively long attempt counts or by cutting off response sequence at a particular count.

3.2 Model prediction detail

After the parameters of the model are trained, each student answer in the test set is predicted one student at a time and one time slice at a time for that student. This prediction procedure is identical to previous literature evaluating KT with the difference of accommodating for multiple responses per time slice. Walking through the prediction procedure; response data for the first student in the test set is loaded. On the first time slice, observable evidence, other than the response, is entered such as attempt counts and resource type being accessed. If there is an answer recorded for one or multiple subparts in the first time slice, the model is told which subpart or subparts were answered and makes a prediction of the student's response(s) based on the parameters learned from the training set. There will always be at least one response in each time slice except for in the case of the resource model where a time slice can represent a resource access. This prediction is logged along with the actual response. After prediction, the model is told what the student's real responses were and the model applies the Bayesian update formula to calculate a posterior and then applies the learning transition formula to calculate the new prior for the next time slice. This processes is repeated until the end of the student's response sequence and the next student is evaluated. Past answers of a student in the test set are used to predict their responses in the next time slice, however; student responses in the test set are not used to aid in prediction of other students in the test set. This form of testing, where data is utilized temporally within an instance, is not typical among classifier evaluations, however it is a principled way of evaluating student models since a real-world implementation of the model would have the benefit of a student's past responses in order to predict future performance.

3.3 Accuracy metric used

The metric chosen to evaluate the goodness of model prediction performance is Area Under the Curve (AUC) also known as Area under the Receiver Operator Curve (ROC). The metric is also equivalent to A' (A-prime). It measures a classifier's ability to discriminate between binary classes, in our case - between incorrect and correct responses. It is an accuracy metric which ranges from 0 to 1, where 1 represents perfect discrimination between responses, 0 represents perfectly inaccurate discrimination (always the opposite of the real value), and 0.50 represents classifier performance that is no better than chance. Approximations are often used to calculate AUC, such as approximate integration under the true positive vs. false positive plot of classifier performance, however the exact calculation provides a much improved intuition for the metric. To calculate AUC exactly: enumerate every possible pairing of positive and negative examples (#positive examples ***** #negative examples). For each pair, check if the classifier's prediction of the positive example is higher or equal to the negative example. The AUC is the percentage of the pairings in which this is true. The AUC metric is therefore a type of ranking metric. So long as all positive class predictions are higher than negative class prediction, a perfect AUC score will be achieved. Deviations from this perfect ranking result in lower AUC. This evaluation makes the AUC metric favorable for detecting differences between two models' ability to discriminate between a correct and incorrect answer.

We calculated AUC for each problem by comparing all predicted responses to the actual responses to subparts of the problem. Since the models in our studies are primarily being used to study learning and performance phenomenon in the aggregate, we used this evaluation instead of the equally employed evaluation of averaging over student AUCs per problem [5]. The per student evaluation is more appropriate when a model's performance at the individual student mastery prediction task is of primary concern.

4. Results

Summarized cross-validated prediction results of the four models; basic, count, idem, and idem count, are presented in this section for the three problem types; homework, lecture, and exam. In addition to predicting our 2,000 sampled students, the models are also evaluated on a smaller 200 student sample to test the reliability of the results with less data. These results are summarized in the next subsection. An analysis of the count model parameters is presented in section 4.2 followed by a deeper analysis of the IDEM model in section 4.3. A two-tailed paired t-test over problems was used to test if the difference in AUC scores between models was statistically reliably different.

4.1 All results and training using less data

A review of the models, their salient features, and number of parameters is shown in Table 2. Results of predicting the 3 problem types with the four models with sample of 200 and 2,000 students are shown in Figure 4. We will first discuss the results of evaluating the 2,000 student sample.

Table 2. Model name, parameters, and description addressing how the challenges described in section 2 were addressed.

Model	Description
basic	Lack of KC model addressed by defining problems as KCs and their subparts as questions of the KC. Retrofit BKT model to allow responses to multiple questions in a single time slice. *parameters = 4*
count	Basic model extended to capture possible variation in information gained from responses due to unpenalized multiple answer attempts. *parameters = 2+2·(#counts)*
idem	Basic model extended to account for variation between questions within a KC (subparts within a problem). *parameters = 2+2·(#subparts)*
idem count	IDEM model with multiple attempt extension. *parameters = 2+2·(#subparts) ·(#counts)*

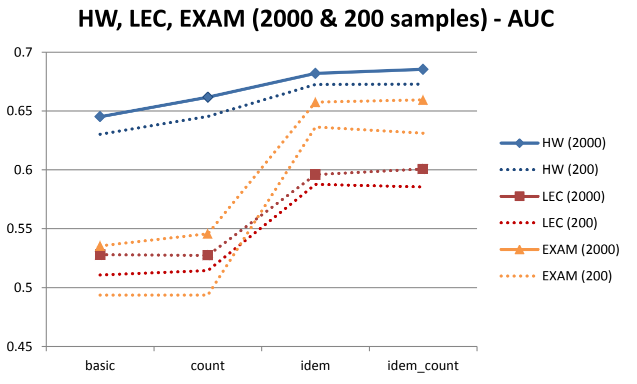


Figure 5. Cross-validated AUC results of the four models by problem type and amount of student data used.

The basic model scored an AUC of 0.6451 on homework, 0.5279 on lecture problems, and 0.5355 on exams. The homework score rivals scores achieved applying BKT to Cognitive Tutor (0.6457) and ASSISTments (0.6690) data [3]; however, the lecture and exam scores are not far above the performance of random chance. A potential upper bound benchmark for the model results presented is 0.7693, achieved in ASSISTments using a blended combination of classifiers [5].

Accounting for different information gain depending on attempt count (count model) resulted in a small but statistically significant gain in the homework (+0.0167 AUC, p = 0.008) but no statistically significant change in lecture or exam prediction performance.

Allowing for individual item guess and slip parameters (idem model) to account for differences among questions in our problem-subpart KC model resulted in the largest and most significant improvement among models. Performance improved in homework (+0.0368 AUC, p = 0.002), lecture problems (+0.0681 AUC, p = 0.013), and most prominently in exams (+0.1220 AUC, p < 0.001).

Adding the count extension on top of the idem model did not result in any statistically significant performance improvement.

Evaluation of all models using 1/10th the number of users resulted in the same relative model performance trends as with the complete sample. This gives us confidence in the reproducibility of the results. Overall predictive performance with the smaller sample decreased most in exam prediction, followed by lecture and homework.

The early stage resource model failed to show gains. In fact, while better than the basic model, it was statistically significantly lower performing than the idem model it was extended from (-0.009 AUC, p = 0.008).

4.2 Count model

There was no effect in modeling attempt count in exams, perhaps because exam attempts were limited to three. There was also no effect on lecture sequence problems, possible because lecture problems are ungraded and students make fewer attempts. However, a small improvement was found on homework, where the most multiple attempt behavior is observed.

We look at the guess and slip parameters of the count model for each answer attempt count to observe at which attempt the most information is being gained. We averaged the learned guess and slip values over the 5 training folds and found that slip stayed almost stationary (+/- 0.02) around its initial parameter value while the guess value precipitously declined with attempt count. The average guess values for each homework problem are plotted in Figure 6.

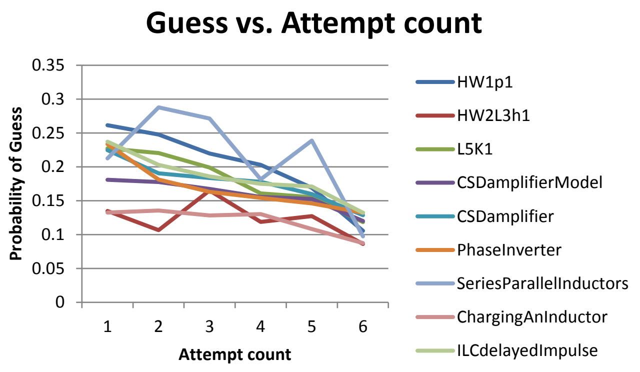


Figure 6. Learned guess parameters at each attempt count for problems in the homework.

A lower guess value means that the model can gain more confidence that the student knows the KC after observing a correct answer. Higher average guess values on the first attempt than the 6th attempt could suggest that students who struggle for longer before answering correctly are more likely to know the KC. Alternatively, lower guess values with attempt counts could simply be returning the student to the same probability of knowledge as when he began the sequence (which had been lowered due to consecutive incorrect answers).

4.3 Item difficulty model (IDEM)

The IDEM model accounted for a large amount of additional variance on top of the basic model, particularly on exam questions and least so on homework. Assuming that the 15 exam problems (midterm + final) were drafted to cover the same space of material as the 37 homework problems, it is possible that there was a higher within-problem variance among exam problem than homework problems, explaining the more dramatic improvement in modeling individual questions in exams over homework. Individual exam, lecture, and homework problem performance is shown in Figures 7, 8, and 9 respectively.

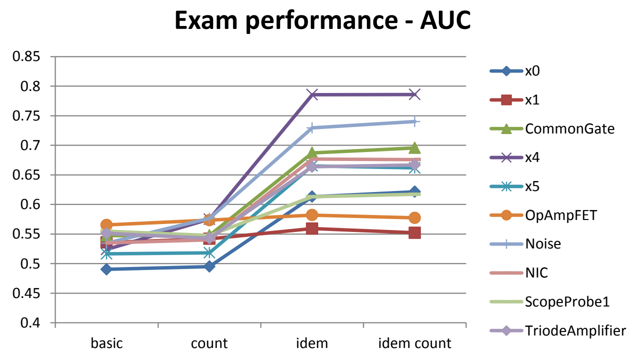


Figure 7. Individual model performance on each exam problem.

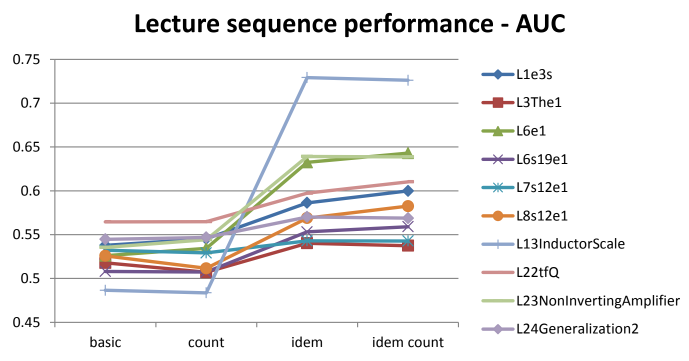


Figure 8. Individual model performance on each lecture problem

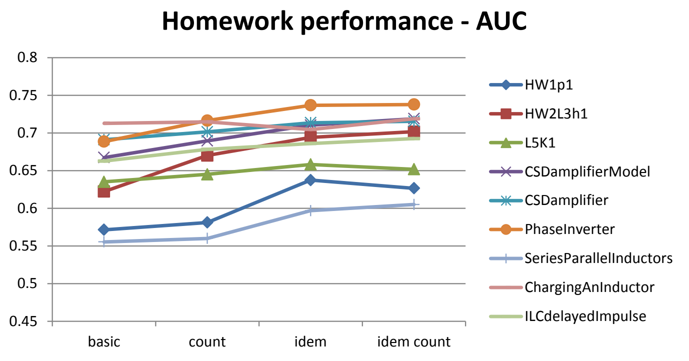


Figure 9. Individual model performance on each homework.

5. Contribution

We have presented a first foray into applying a model of learning to a MOOC. We identified three challenges to model adaptation and found that modeling variation in question difficulty resulted in the largest performance gain given our definition of KC. While our KC definition as problem with subparts as members is not ideal for measuring learning throughout the course, it nevertheless resulted in AUC performance accuracy rivaling that of prediction within systems with subject matter expert defined KC models. While we elucidated the potential for knowledge discovery given the unique variation in resource access in MOOC data, much work is left to demonstrate that this information can be seized on to produce more accurate results. This raises the question of how the efficacy of resources generalizes and the contexts and background information that needs to be considered to identify what works and for whom. Our solutions to the first two challenges, of lack of a KC model and multiple unpenalized attempt counts, will serve as an initial foundation for an efficacy assessment framework for MOOCs.

REFERENCES

[1] 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.

[2] Anderson, J. (1993). *Rules of the mind*. Hillsdale, NJ: Lawrence Erlbaum Associates

[3] Pardos, Z. & Heffernan, N. (2011) KT-IDEM: Introducing Item Difficulty to the Knowledge Tracing Model. In Konstant et al (Eds.) Proceedings of the 20th International Conference on User Modeling, Adaptation and Personalization (UMAP 2011). pp. 243-254.

[4] Gonul, F., & Solano, R. (2012). Innovative Teaching: An Empirical Study of Computer Aided Instruction in Quantitative Business Courses. Available at SSRN 2057992.

[5] Pardos, Z.A., Gowda, S. M., Baker, R. S.J.D., Heffernan, N. T. (2012) The Sum is Greater than the Parts: Ensembling Models of Student Knowledge in Educational Software. ACM SIGKDD Explorations, 13(2)

[6] Reye, J. (2004). Student modelling based on belief networks. International Journal of Artificial Intelligence in Education: Vol. 14, 63-96.

[7] Beck, J., Chang, K. M., Mostow, J., & Corbett, A. (2008). Does help help? Introducing the Bayesian Evaluation and Assessment methodology. In Intelligent Tutoring Systems (pp. 383-394). Springer Berlin/Heidelberg.

[8] Pardos, Z.A., Dailey, M. & Heffernan, N. (2011) Learning what works in ITS from non-traditional randomized controlled trial data. The International Journal of Artificial Intelligence in Education, 21(1-2):45-63.

[9] Rau, M., Pardos, Z.A. (2012) Interleaved Practice with Multiple Representations: Analyses with Knowledge Tracing Based Techniques. In Proceedings of the 5th annual International Conference on Educational Data Mining. Crete, Greece. Pages 168-171

[10] Tatsuoka, K.K. (1983) Rule space: An approach for dealing with misconceptions based on item response theory. Journal of Educational Measurement, 20, 345-354.

[11] Spada, H., & McGaw, B. (1985). The assessment of learning effects with linear logistic test models. Test design: New directions in psychology and psychometrics, 169-193.

[12] Lovett, M. C. (1998). Cognitive task analysis in service of intelligent tutoring system design: A case study in statistics. In Proceedings of the Fourth Conference on Intelligent Tutoring Systems. pp. 234-243. Springer-Verlag.

[13] Martin, B., Mitrovic, T., Mathan, S., & Koedinger, K.R. (2011). Evaluating and improving adaptive educational systems with learning curves. User Model User-Adap Inter (2011) 21:249–283.

[14] Anderson, J. R., F. G. Conrad, and A. T. Corbett (1989) Skill acquisition and the LISP Tutor. Cognitive Science, 13, 467-505.

[15] Koedinger, K.R., McLaughlin, E.A., & Stamper, J.C. (2012). Automated student model improvement. In Proceedings of the Fifth International Conference on Educational Data Mining. pp. 17-24.

[16] Kortemeyer, G. (2009). Gender differences in the use of an online homework system in an introductory physics course. Physical Review Special Topics-Physics Education Research, 5(1), 010107.

[17] Attali, Yigal. "Immediate Feedback and Opportunity to Revise Answers Application of a Graded Response IRT Model." Applied Psychological Measurement 35.6 (2011): 472-479.

[18] Bergner, Y., Pardos, Z.A., Seaton, D., Pritchard, D.E. (under review) Two steps forward one step back: analysis of out of sequence problem checking behavior in the edX system. Submitted to the 16th International Conference on Artificial Intelligence in Education.

[19] Bloom, B. S. (1968) Learning for mastery. In Evaluation Comment, 1. Los Angeles: UCLA Center for the Study of Evaluation of Instructional Programs.

[20] Murphy, Kevin Patrick. (200) Dynamic bayesian networks: representation, inference and learning. Diss. University of California.

[21] Gong, Y., Beck, J.E., Heffernan, N.T., 2010. Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting Procedures. In Proceedings of the 10th International Conference on Intelligent Tutoring Systems, 35-44.

[22] Baker, R.S.J.d., Corbett, A.T., Aleven, V. (2008) Improving Contextual Models of Guessing and Slipping with a Truncated Training Set.Proceedings of the 1st International Conference on Educational Data Mining, 67-76.

Modeling and Optimizing Forgetting and Spacing Effects during Musical Interval Training

Philip I. Pavlik, Jr.
University of Memphis
Psychology Dept.
Memphis, TN 38152
1-901-678-2326
ppavlik
@memphis.edu

Henry Hua
University of Memphis
Psychology Dept.
Memphis, TN 38152
1-901-678-5590
hyhua
@memphis.edu

Jamal Williams
University of Memphis
Psychology Dept.
Memphis, TN 38152
1-901-678-2364
jawllm10
@memphis.edu

Gavin M. Bidelman
University of Memphis,
Sch. Comm. Sci. & Disorders
Memphis, TN 38152
1-901-678-5826
gmbdlman
@memphis.edu

ABSTRACT

From novice to expert, almost every musician must recognize musical intervals, the perceived pitch difference between two notes, but there have not been many empirical attempts to discover an optimal teaching technique. The current study created a method for teaching identification of consonant and dissonant tone pairs. At posttest, participants increased their ability to discern tritones from octaves, and performance was better for those who received an interleaving order of the practice trials. Data mining of the results used a novel method to capture curvilinear forgetting and spacing effects in the data and allowed a deeper analysis of the pedagogical implications of our task that revealed richer information than would have been revealed by the pretest-to-posttest comparison alone. Implications for musical education, generalization learning, and future research are discussed.

Keywords

Model-based discovery, forgetting, interleaving, spacing effect, computer adaptive training, musical consonance

1. INTRODUCTION

Music is a rich multi-modal experience which taps a range of both perceptual and cognitive mechanisms. As in other important facets of human cognition (e.g., speech/language), music consists of constituent elements (i.e., scale tones) that can be arranged in a combinatorial manner to yield high-order units with specific categorical labels (e.g., intervals, chords). In Western tonal music, the octave is divided into 12 pitch classes (i.e., semitones). When combined, these pitch relationships can be used to construct twelve different chromatic intervals (each one semitone above or below another) that are labeled according to the relationship between fundamental frequencies of their tones. For example, two tones can form an octave (2:1 ratio), a perfect fifth (3:2 ratio), or a variety of other tonal combinations.

Perceptually, musical intervals are typically described as either consonant, associated with pleasantness and smoothness (e.g., octave), or as dissonant, associated with unpleasantness and roughness (tritone, ratio: 11:8). Helmholtz (1895) defined this “roughness” as the fluctuations in amplitude perceived by a listener which occurs when the distance between partials is small enough for them to interact (i.e., “beat”) within the auditory periphery. Consonance, on the other hand, occurs in the absence of such beating, when low-order harmonics are spaced sufficiently far apart to not interact (e.g., octave, perfect fifth). Behavioral studies demonstrate that listeners treat the various intervals hierarchically and tend to prefer consonant over dissonant pitch relationships [2, 7, 10]. It is this hierarchical arrangement of pitch which largely contributes to the sense of a musical key and harmonic structure in tonal music [8].

The ability to discriminate consonant and dissonant intervals is far better than chance [20]. Indeed, this ability emerges early in life as both newborns and infants show a robust preference for consonant over dissonant tone pairs, well before being exposed to the stylistic norms of culturally specific music [5]. As such, it is posited that the perceptual distinction between consonance and dissonance might be rooted in innate auditory processing [1, 2]. While the perceptual *discrimination* of music intervals is fairly well studied [1, 2, 7, 9] the ability to *identify* intervals remains poorly understood. While the capacity to distinguish aspects of musical structure might be present at birth, the orientation towards culture-specific music and its definitions must progress during childhood. This developmental perspective of musical learning marks the vital periods for receptivity of musical structures. However, musical training is not a standard thus resulting in a continuum of musical ability amongst the general population.

Here, we ask how individuals learn to *identify* (i.e., categorically label) the pitch combinations of music. Interestingly, musically naïve individuals could potentially benefit from musical training. Research suggests that late musical training might compensate for some gaps in musical knowledge such as the enhancement of automatic encoding of interval structures [4]. Additionally, music training is said to have both short-term and long-term benefits in regards to transfer [19]. Indeed, the benefits of prolonged musical training could have long-term effects such as improved executive functioning and perceptual organization [5]. Specifically, studies suggest that training greatly influences the developing brain, making music training a promising model for examining learning [12, 13]. If music training is a potential model for investigating learning then the implementation of different training regimens should help elucidate the general learning process.

Specifically, by training individuals to identify different *harmonic intervals*, a more optimal training regimen could be derived and contribute to our understanding of pedagogical strategies. (A harmonic interval is when two tones are played simultaneously, as opposed to a melodic interval where two tones are sequential.) As previously stated, studies have shown that people already have some prior knowledge when it comes to interval discrimination, regardless of enculturation effects or innate capacities. Indeed, listeners appear to be better at distinguishing some intervals over others. For example, intervals of small integer ratios, such as perfect fourths (4:3) and perfect fifths (3:2), are typically more difficult to discriminate and are often confused [6, 18]. Furthermore, the order in which people learn certain intervals has been seen to have an effect on their ability to make accurate distinctions, e.g. [6]. If factors such as these could be understood more deeply, it may lead to more effective ways of configuring musical training.

2. METHOD

2.1 Participants

After screening our data for participants (Amazon Turk workers) who provided a full set of responses, without omissions, we had 220 participants. The average participant age was 29.88 years (*SD* = 10.32). Participants had a mean of 4.30 years of musical experience (*SD* = 5.70). Parsed into different types of musical training, participants had on average 1.34 years (*SD* = 2.31) of training with a private tutor, 2.27 years (*SD* = 4.78) studying music on their own, and 2.86 years (*SD* = 3.38) of training in a formal school setting. Prior ear training experience averaged 0.47 years (*SD* = 1.61), 0.24 years (*SD* = 0.81) of which focused specifically on harmonic interval training.

2.2 Procedure

Participants self-selected this study from a list of various available research participation opportunities on Amazon Mechanical Turk, an online data-collection service. Participants were paid $3. Participants began the study with a survey. Items included demographic information such as age and sex. This survey also asked for various predictors, including years of different types of musical training (overall, private tutoring, school), and types of musical training (ear training, harmony, reading music).

Upon finishing the survey, participants completed the interval identification task. Prior to the pretest, to ensure that people understood the task, the following instructions were given for participants to view (participants clicked a button after reading):

> Hint: The octave interval is sometimes described as smooth, pleasing or pure. The tritone interval is sometimes described as harsh, diabolic or impure.
>
> Task: Please listen to each 2 second interval, then type 'o' for octave or 't' for tritone. After each incorrect response, you are provided review to help you learn.
>
> Goal: Practice the sound identification task, attempting to learn the interval (octave or tritone) between two notes played at the same time. This pretest portion will get an initial measure of your skill in the task, and will be followed by 96 training practices, and finally a posttest of 32 practices.

This task contained three stages: practice, learning, and posttest. The practice section presented 32 intervals for the participant to label as either a tritone or octave. In the learning section, 96 intervals were presented and participants were asked to similarly label the intervals as tritone or octave. The orders of the learning intervals were presented in various sequences (see section 2.3, Conditions). Sequence type was thus the primary independent variable of the study. The posttest section, which is the primary dependent variable of the study, once again presented 32 intervals to be labeled as tritone or octave.

All trials presented the interval sound file which lasted 2 seconds and then they typed ‘t’ or ‘o’ to indicate their response (trials timed-out after 2 minutes, but learners typically responded in less than 2 seconds). After responding, a checkmark appeared for .5 seconds to indicate the selected answer was correct; if incorrect, the correct answer was given with a replay of the now labeled interval sound just responded to incorrectly, these “study” opportunities lasted 5 seconds (so there was 3 seconds of silence after each replay).

2.3 Conditions

This experiment varied the presentation sequencing to test the effectiveness of various presentation orders on the task of identifying tritone and octave harmonies. Practice trials were presented in combinations of progressive and interleaving orders organized into four blocks, each containing 24 harmonic intervals. Conditions were randomized between subjects.

A progressive order presented the harmonic intervals in consecutive blocks, each block containing the same two intervals but presented at a higher pitch register than the previous block. Block 1 contained intervals in a low register (155.6Hz/311.1Hz for octave and 185Hz/261.6Hz for tritone) block 2, intervals of a medium-low register (277.2Hz/554.4Hz for octave and 329.6Hz/,466.2Hz for tritone), block 3, intervals from a medium-high register (493.9Hz/987.8Hz for octave and 587.3Hz/830.6Hz for tritone) and block 4, intervals from a high register (880Hz/1760hz for octave and 1046.5Hz/1480Hz for tritone). Sounds were synthesized by MIDI using instrument 1 (Piano) for a 2-second duration. An antiprogressive order presented harmonic intervals in a way that made each block maximally different from the previous block. block 1 consisted of low register tones, block 2, high register tones, block 3 medium-low tones, and block 4, medium-high tones.

An interleaving order introduced a new register for each of blocks 2-4 according to the antiprogressive or progressive order, with tones already heard from the previous blocks interleaved with the new material. In other words, new registers were taught while practicing the old ones, with an equal distribution for each of the presented tone levels within a block of 24. Conditions lacking an interleaving order did not repeat tones from previous blocks.

Therefore, the 4 experimental conditions contained all 4 combinations of progressive and interleaving orders: progressive and no interleaving, antiprogressive and no interleaving, progressive with interleaving, and antiprogressive with interleaving. As a control group, there was one condition that presented 96 learning in 4 blocks that were fully mixed, (just like the pretest and posttest). For all conditions, although each block contained a predetermined set of tones, tones were randomized within each block. Finally, practice during learning “blocks” were not marked by a brief pause with an introduction screen like the pretest, learning, and post-test were marked. In other words, transitions between sets of different items during practice were not signaled to subjects.

3. RESULTS

An analysis was conducted on the number of participants in each of the five conditions (*M* = 44.00, *SD* = 5.10), to assess whether dropout was more prevalent in certain conditions. The control condition had the fewest observations, with only 37 participants, whereas the largest group, the interleaving–progressive condition, had 51 participants. Since the control condition was most difficult, we thought that might be causing this disparity. However, attrition did not differ statistically between conditions, $\chi 2(68) = 78.15$, $p = .2$.

Means and standard deviations by condition for the post-test were as follows: Condition 1 (progressive only): Mean=.78, SD=.15; Condition 2 (progressive + interleaving): Mean=.82, SD=.16; Condition 3 (no progressive or interleaving): Mean=.79, SD=.14; and Condition 4 (interleave only): Mean=.86, SD=.16. There was statistically significant improvements, averaged across conditions, from pretest to posttest, $t(219) = 9.75$, $p < .01$. Upon finding an overall positive effect from pre to posttest, analyses focused on systematic differences between interleaving and progressive tone presentations in the practice trials. Figure 1 shows average performance across the entire 160 trials of the experiment, averaged in blocks of 8 trials. The four experimental conditions (progressive without interleaving, antiprogressive with interleaving, progressive–interleaving, and antiprogressive without interleaving) were analyzed according to two dichotomous criteria: those with or without progressive presentations, and those with or without interleaving presentations. The control condition, which presented intervals in a random order, was not used in this 2 way (interleaving/no interleaving) x 2 (progressive/antiprogressive) analysis of variance (though the results in that condition are in the same direction as below, since the highly interleaved control also performed well for learning gain relative to the blocked conditions). We used pretest score as a covariate.

The ANCOVA revealed an effect of interleaving during practice trials on posttest scores, even when controlling for pretest scores. Interleaving trial presentation order had a significant positive effect on posttest scores, $F(1, 178) = 5.81$, $p = .02$, $d = .34$. Progressive and antiprogressive ordering had no reliable effect on posttest scores, [$F(1, 178) = .48$, $p = .5$]. The interaction of progressive and interleaving orders was also non-significant, $F(1, 178) = .33$, $p = .6$.

The last series of analyses looked for any relationships between individual differences in musical training or skill with posttest scores and the overall magnitude of the pretest–posttest improvement. A list of correlations is listed in Table 1. There were modest relationships between various musical training-related and skill-related predictors and posttest scores, and some suggestion of a negative relationship between previous learning and improvement.

Overall, results suggest that the teaching paradigm created for this experiment caused improvement from pretest to posttest. This corroborates previous work [6], which presented the practice trials in an interleaved rather than sequential order, also showing a reliable advantage. Certain qualities of musical skill and training had modest relationships with posttest performance, but were negatively related to improvement. This negative correlation suggests that proficient musicians had less to gain than poorly skilled musical learners from our practice procedure.

Table 1. Correlations between participant factors and scores.

Participant factor	Posttest	Improvement
Years overall musical training	.28*	-.10
Years of musical tutoring	.25*	-.10
Ability to read music	.26*	-.16*
Understanding of musical harmonies	.30*	-.15*
Ability to hear musical harmonies	.30*	-.10
Years of self-directed musical training	.26*	-.08
Years of musical training in school	.31*	-.07
Years of ear training	.13	.03
Years of ear training on harmonies	.18*	.04

Note. N = 220, *$p < .05$

4. MODEL BASED DISCOVERY

While this paper mines data from a novel musical educational task, the computational model of the data we created is based on many of the common principles of educational data mining. To begin, the model builds upon a simple additive factors model (AFM) [3, 21]. The AFM is a logistic regression model that is based on the logic of counting prior practice events, so that the prior practice is an "additive factor" predicting future performance. In this early stage of research to understand how people learn musical intervals, we began with the simple assumption that each of our 8 stimuli was a knowledge component that could be learned independently in some sense. In a later model in this paper, we will also demonstrate how we can add a generalization component specific to each interval, but our current experimental design was not appropriate for more deep analysis of generalization primarily because the conditions only used 2 intervals and did not vary the spacing of each interval, rather, in all conditions, there was always a 50/50 chance of either interval for each trial.

However, while we fit a model that is built upon AFM principles of counting prior instances of practice of particular types, we found that the simple AFM model could not account for some of the effects in Figure 1. The first effect that could not be captured by AFM was simple interference based forgetting as a function of the number of trials since the prior repetition. We can see this forgetting effect (probably driven by multiple processes, including interference) manifesting in the practice block differences and transitions between practice blocks. For example, note how fast learning is in each block when items are blocked, but then observe the huge decrement when items are mixed in the final posttest. Retention is apparently quite poor when there is other practice between repetitions of the same type.

We can see similar differences comparing performance between blocks also, noting that as items are added in the interleaved conditions performance steps down with each additional register added to the set of stimuli blocked together. The mechanism behind this forgetting effect is not entirely clear. However, previous work demonstrates that the perceptual distinction between musical pitch relationships is continually strengthened with exposure and training [11]. Thus, it is possible that periodic lapses in performance across the practice blocks might be due to the fact that our non-musician listeners' internal templates for the intervals are not yet robust, rendering the mapping between interval sound and label unstable, and ultimately hindering behavioral identification.

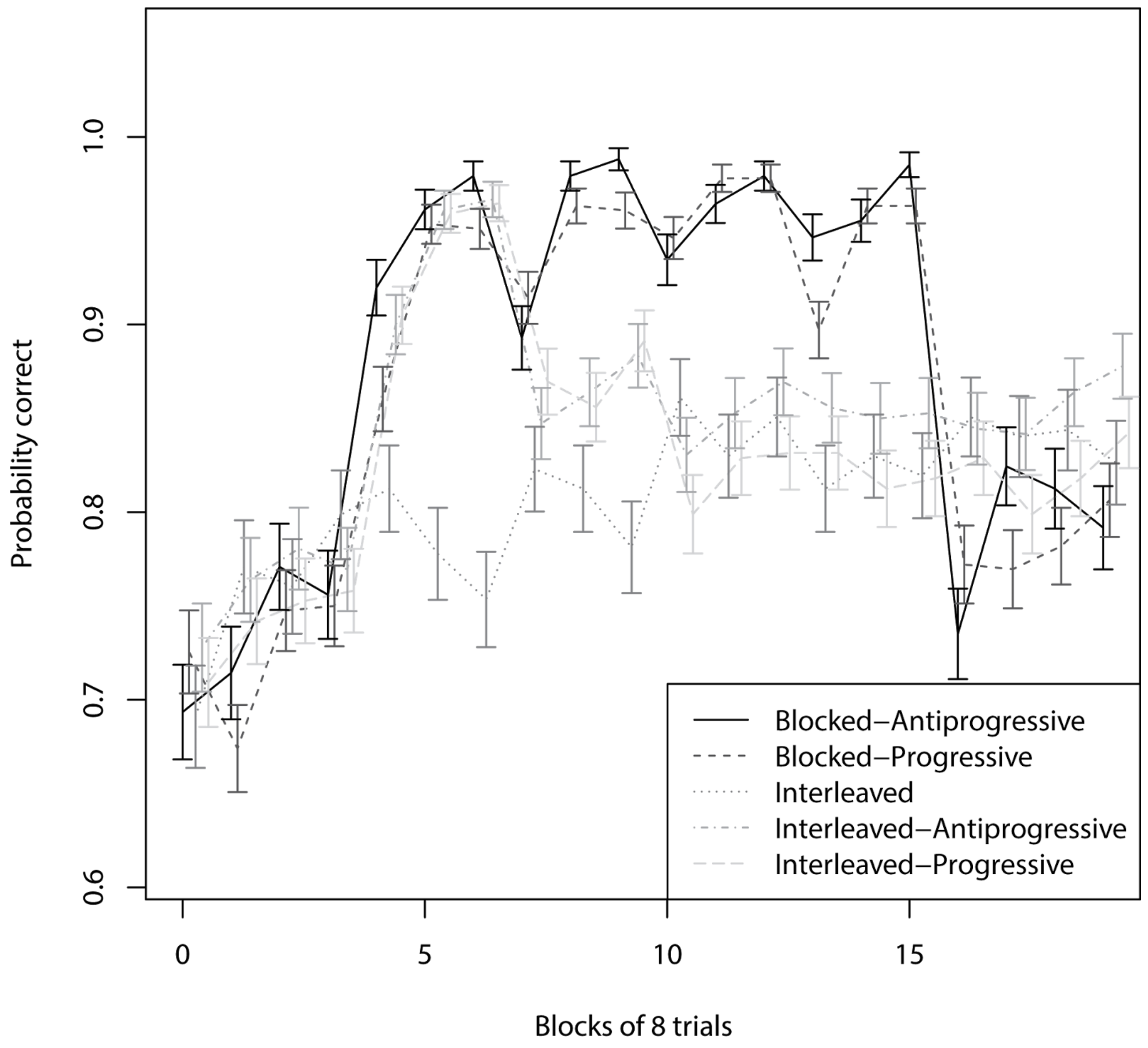


Figure 1. Performance in all conditions plotted in blocks of 8 trials. Error bars are 1 *SE* confidence intervals. First 4 and last 4 blocks represent pre- and post-test trials.

The second effect that AFM was unable to capture was the benefit of spacing/interleaving that we saw in the interleaved conditions with the ANOVA analysis. This was described in Section 3 as a significant benefit for the 2 conditions which employed more interleaving. We can also see this effect in Figure 1 as a visible difference between the interleaving and blocked conditions at posttest. This "spacing" effect, which is also very common in verbal memory experiments [16], was not by itself as strong as the effect of forgetting, but it has important implications for education. If musical educators can make use of this effect, our data suggest they may enhance learning. While AFM does not capture such an effect in the original incarnation, more complex models can capture such data. For example, Pavlik and Anderson [16] describe one such model, which functions by proposing less decay for as a function of the increased difficulty of more widely spaced practice. Unfortunately, such models have several parameters and no analytic method of solution, so, solving these models is extremely difficult due to issues of time and local minima. To resolve some of these issues we wanted to find a way to fit a similar model as Pavlik and Anderson, that relied less on an ad hoc, difficult to solve (albeit accurate) model form and more on an established model formalism (logistic regression).

The problem was that both of the effects we wanted to model, forgetting and spacing, depend on a model where each observation is predicted by a parameterized function of prior knowledge (not just the count, as in AFM), and typical logistic formalisms only allow the independent data for each observation to be used to predict the dependent effect. Of course, AFM solves this by keeping a simple count of prior practices for a skill or item-type and adding it to the data for each row so that these prior events can have an effect. This will not work easily for decay however, unless we want to have learning decrease linearly when other items are practiced. However, linear forgetting has never been considered a viable model [17]. To work around this limitation, we decided to model decay by adding a parameter that captures a percentage loss for each item as each other item is practice (exponential decay).

So, for example, if we look at the vector of prior events for some arbitrary item we might notice it is practiced at opportunity 3 and opportunity 7. Normally, in AFM this would be represented as the vector 0, 0, 0, 1, 1, 1, 1, 2, 2, 2..., thus the student would have 1 prior counted for the prediction of how they will perform on trial

7, and if they performed on trial 10, they would be credited 2 priors when computing their chance. Note, how the credit is lagged so that when we compute performance for trial 3 we do not (quite sensibly) have any priors, but when we compute the prediction for trial 4, we then have 1 prior.

With the new mechanism we introduce a "forgetting rate", *d*, that we estimated and applied to computing the prior credit vector. This decay was applied to each prior practice independently, so that if decay was say .7 (for our example above) we would have the vector 0, 0, 0, .7, .49, .34, .24, 0.17, 0.12, 0.08, and the vector 0, 0, 0, 0, 0, 0, 0, .7, .49, .34 summed equals 0,0, 0, .7, .49, .34, .24, .87, .61, .43. So, given the .7 parameter value, the student would have .24 prior counted when they perform on trial 7, and if they performed on trial 10, they would be credited .43 prior decayed strength when computing their chance.

Next, we wanted to add spacing effects to the model by using the ideas from Pavlik and Anderson [16]. In this work, Pavlik and Anderson proposed that long-term learning benefit (decayed remnant) for spaced practice was an inverse function of the current strength. To adapt this model we used the decaying strength vector as an exponent in a power function model where we estimated the base as another new parameter, *g*. So, for example, given a spacing parameter *g*=.005, we find that .005^0 = 1 long term learning (e.g. for the first trial, which is the 3rd opportunity in our example above) while .005^.24 = .28 long-term learning for the 7th opportunity. Long-term learning is a new vector that works in addition to the decay strength to predict performance. In our example, we would sum 0 , 0, 0, 1, 1, 1, 1, 1, 1, 1 for the first practice and 0, 0, 0, 0, 0, 0, 0 , .28, .28, .28 for the second practice. This long-term learning is permanent.

This model was estimated by nesting a logistic general linear model (GLM) within a general gradient descent optimization function. This wrapper optimization took the decay and spacing parameters, transformed the data vectors based on those parameters, and then computed the optimal logistic model and outputted the fit of that model to the wrapper. The wrapper then used the internal model's fit to adjust spacing and decay by brute force gradient descent steps (the bounded BFGS method from the `optim` function, built into R), to get a global optimization for the wrapped GLM function given the decay and spacing parameters. Figure 2 shows this optimization structure in R code where `temp` is a vector that holds the decay and spacing parameters.

```
model <- function(temp, data) {
   compute data
      as a function of temp
   compute GLM model fit using data
   return log likelihood fit}

optim(temp, model, method
   =c("L-BFGS-B"),lower=0, upper=1))
```

Figure 2. Wrapper optimization loop pseudocode.

The GLM model included fixed effects to capture the 2x4 main effects and interactions caused by the particular tones and intervals, and the prior decayed strength and prior long-term learning for the particular stimuli. Figure 3 shows the GLM structure (i.e. the independent variables that predict the dependent), which shows how we fit a single coefficient for the effect of prior decayed strengths, and a single coefficient for the effect of the long-term benefits (using the `I` function in R allows us to use the vector sum since each vector applies independently of the prediction). This means that the data vectors for these values were linearly scaled inside the GLM, while being created in the wrapper. This allows us to fit a much more complex model than if we just used the wrapper, since brute force gradient descent would have been prohibitively slow with 3 (or more) parameters. Instead, putting the GLM in the wrapper allows us to fit the minimum number of non-linear parameters (2) inside the slow brute force procedure, and then optimize several more parameters in the efficient GLM logistic function. Table 3 shows the more complex AFM-decay-space model compared to two simpler models via cross validation. The R code for the model equation first finds a parameter for the 8 decaying vectors for the 8 components (octave0, etc.). Fitting a single parameter for the effect of each of the 8 vectors simplifies the model under that assumption that forgetting is equivalent for each register by interval combination (using the `I` function in R allows us to sum vectors since each vector applies independently of the prediction). Similarly, we also assume a single parameter for the permanent learning vectors (soctave0, etc.), which account for the long-term learning from spaced practice for each stimulus type. Finally, the interval by tone interaction captures fixed-effect differences that may be due to average effects of poor fidelity of the participants' audio speakers or hearing in some registers and any other specific differences in the baseline performance with each register by interval pair. However, it might be noted that the variety of significant differences for tone and interval were not well controlled for (e.g. order of introduction) in our design, so we choose not to analyze them here.

```
answer ~

I(octave0 + octave10 + octave20 + octave30 +
tritone0 + tritone10 + tritone20 +
tritone30) +

I(soctave0 +
    soctave10 + soctave20 + soctave30 +
stritone0 + stritone10 + stritone20 +
stritone30) +

interval * tone
```

Figure 3. Logistic GLM model structure.

We tested 3 models with 5 runs of 10 fold cross validation to confirm the model generalized to our data as shown in Table 2. The three models were AFM, which simply summed the prior practices in for each of the 8 stimuli, AFM-decay, which included the new decay mechanism, and AFM-decay-space, which further layered the spacing effect mechanism into the model and is shown in Figure 3. While the test analysis reveals strong significant difference for the AFM-decay model for both *r* and *MAD*, the AFM-decay-spacing model was barely (Z=2.16, p<.05) better in terms of *r* and not significant for MAD. We added spacing effects after decay effects, since spacing effects are very small compared to decay effects in an experiment with only one-session, and thus a spacing effect mechanism may inappropriately capture effects due to decay unless decay effects are removed first (the spacing mechanism may actually improve the model in such a case, but parameter values will not meaningfully indicate a benefit to spacing, perhaps even the inverse).

Table 2. Cross validation results.

Model	Train		Test	
	Spearman *r (SE)*	*MAD (SE)*	Spearman *r (SE)*	*MAD (SE)*
AFM	0.10974 (0.00047)	0.25315 (0.0005)	0.1067 (0.00392)	0.25347 (0.00203)
AFM-decay	0.24013 (0.00044)	0.24167 (0.00053)	0.23783 (0.00402)	0.24206 (0.00239)
AFM-decay-space	0.25211 (0.00037)	0.23987 (0.00042)	0.24921 (0.0034)	0.24026 (0.00197)

It was also interesting to check how well the model could be used to simulate the experiment. Figure 4 shows graphically how well the model captures the aggregate effects. Note that even the error bars are of very similar magnitude. This simulation was constructed by generating random number from 0 to 1 that were than compared to the model of each trial to determine whether the trial was responded to correctly in the simulated result.

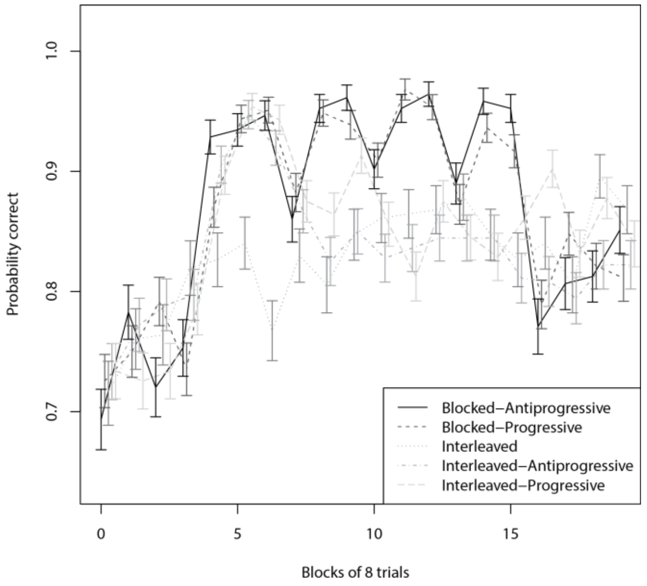


Figure 4. Simulation of experiment

Finally, we wanted to see if there was any general transfer by interval type. While normally we might expect spacing to be an important factor in this effect, the simplicity of experiment (as noted) results in a 50/50 chance of either interval for each trial, so spacing between intervals does not have a great deal of variability. Because of this we used a generalization model that merely tracked the intervals prior practice count, but also used the performance factors analysis (PFA) formalism to track the interval counts depending on success or failure. The PFA method works just like AFM, but counts prior success and prior failure practices instead of simply the count of undifferentiated prior practice [15]. Purely to improve simplicity, we also choose not to account for the fixed interval x tone effects, which did not appear to change the other model coefficients much, despite reducing the fit as expected. Figure 5 shows this model structure. This model adds on 2 PFA components to track learning as a function of prior failures (trif + octf) or success (tris + octs) count for each interval type. Again we fit a single parameter for both intervals under the assumption that are learned at equivalent rates. Again we used the I function to sum the columns since they were mutually exclusive predictors in the equation.

```
answer ~

I(octave0 + octave10 + octave20 + octave30 +
    tritone0 + tritone10 + tritone20 +
tritone30) +

I(soctave0 +
    soctave10 + soctave20 + soctave30 +
stritone0 + stritone10 +
    stritone20 + stritone30) +

I(tris + octs) +

I(trif + octf)
```

Figure 5. GLM model structure with PFA generalization

4.1 Application of Discovered Model to Pedagogical Inference

The model discovered (Figure 5) is useful because it can be used to make pedagogical inference combined with a model of costs for the actions the model allows. The combined model allows us to consider the long-term gains from different conditions of practice relative to the current practice costs. Figure 6 additionally describes a model of practice costs (time spent in practice) as a function of prior practice. This simple model implies a maximal cost for success with unpracticed items, which decreases to a minimum as practice accumulates. This simple model predicts latency cost of success, and we also estimated latency cost of failure and review practice at 7.054 seconds using the overall average from the data. In the model we used the decaying short-term strengths as the predictor.

```
latency ~

I(1/(1 + octave0 + octave10 + octave20 +
    octave30 + tritone0 + tritone10 +
tritone20 + tritone30)
```

Figure 6. LM model structure for costs.

Then we can extract the parameters from both models. See table 3.

Table 3. Parameters used to in pedagogical inference.

Parameter	Value
d	.628
g	.00106
fixed cost failure	7.054
logistic intercept	0.99
spacing coefficient	.131
decay coefficient	2.36
PFA gain failure coeff.	-.106
PFA gain success coeff.	.0144
latency intercept	.154
latency coeff.	1.296

The values from Table 3 then allow us to construct an Excel simulation (available from the first author) of the optimality conditions for our task by examining when spaced gain and PFA parameter gain (the undecaying learning gains) are maximal relative to the time spent on practice. To do this we plot the learning efficiency (gain / time cost, where both gain and time cost are conditional on success or failure in the calculation) at various levels of prior knowledge (probability values as inferred from the effect of the short-term strength) to find an optimum for the efficiency that allows us to see the optimal probability at which to practice each item. Figure 7 shows the long-term gain curve, the current cost curve, and the optimal efficiency curve.

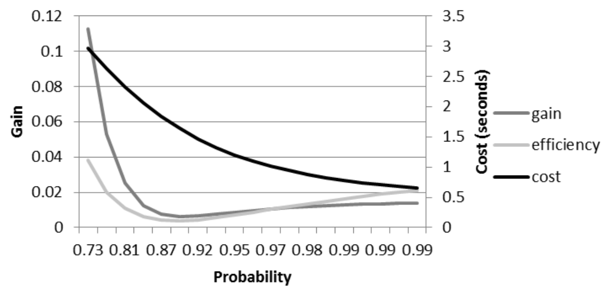


Figure 7. Optimality function.

The results imply it is always best to widely space, since efficiency is maximal with a low probability correct, which would come from very wide spacing between repetitions. However, this may also be because the data was not strong enough to draw conclusions very clearly in this case. Small differences in the fit of the spacing effect result in large changes to the predictions (note the difference in the gain curves for Figures 7 and 8). Indeed, the experiment only had weak power to determine the shape of the spacing effect gain because even in our most intermixed conditions spacing averaged only 8 intervening trials. Because of this the experiment may have poor power when extrapolating to inferences about spacing that imply much wider spacings than were actually used to parameterize the model. As can be seen in Figure 8, when spacing effects are strong the prediction changes.

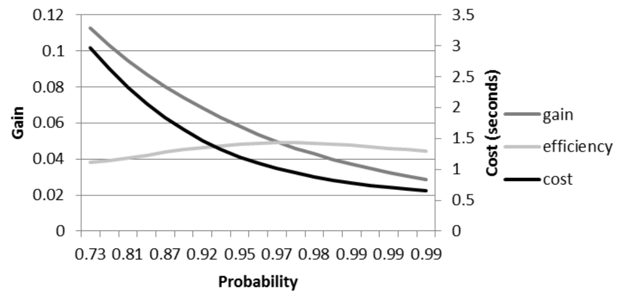


Figure 8. Optimality with spacing parameter (only) changed to .4.

Another problem in trusting the optimality model is due to the noise introduced by only having 2 response options in the experimental design. This makes it hard to identify if success are true success or only guesses, since a guess has a 50% chance of success. This consequently means the PFA parameters are averaging over some of the effects of guessing, thus blunting the quantitative difference between success and failure. This can also have a large effect on the optimization, since the gain curve shape depends on the PFA parameters since they produce the difference in gain for different rates of correctness probability.

So, in addition to what the model reveals about the processes of forgetting and spacing in our data, the model also allows this sort of principled speculation on how the model might be improved if we collect better data to parameterize it. To correct these two limitations of the data we will need to add several more intervals to the practice mix in succeeding experiments. Additional intervals will allow for more spacing (since we have more options for other items to practice when spacing one interval type) and will also improve the effectiveness of the PFA parameters by reducing the noise inherent in modeling success that has a high rate of guessing. In addition to providing more resolution for the spacing effect and PFA parameters, more intervals also will allow much deeper analysis of generalization, since generalization will no longer be a simple binary distinction, but rather a complex categorization.

5. CONCLUSIONS

Although the ability to discern musical intervals is a basic skill vital for almost every musician—beginner or expert—there is a shortage of empirical studies on effective teaching techniques for this skill. In this study, we created a computerized system that tracked participants' identification performance during the process of musical interval learning. The results suggest that our teaching method caused improvement from pretest to posttest, and that an interleaved order was more effective for interval learning. Mathematical models of the data revealed that, while participants improved as a result of our program, there were robust patterns in the practice trials between pretest and posttest. The practice trials showed learning within each block and quick forgetting from one block to the next.

The model we have made is not as detailed as [16], but because of that simplicity, it was possible to more easily fit the model. Importantly, unlike the Pavlik and Anderson model, practices in this new AFM variant model capture spacing effects as permanent learning rather than learning that is merely more durable. Indeed, this new model is in some respects closer to work that has modeled forgetting and spacing using a distribution of units that decay exponentially, some more quickly and some more slowly [14]; however, the model in the current paper is far simpler since it only uses 2 units, one permanent that is a function of current strength and one temporary with relatively quick exponential decay. Others have looked at decay in an educational data mining context, but this work has been at a coarser grain-size looking at forgetting over sessions, and not at the event level [22].

From the perspective of music pedagogy, our training paradigm highlights the importance of the learning sequence in the process of musical interval learning. Ear training and aural skills courses nearly always progress in a rote manner whereby musical relationships are taught serially based on their apparent difficulty (e.g., tones, intervals, chords, harmonies). Our data demonstrate that this typical curriculum is relatively inefficient. Instead, interleaving intervals—here, across multiple pitch registers—seems to promote more efficient learning. Presumably, the higher effectiveness of interleaving in music learning results from having to map sound to meaning across a more diverse acoustic space. Interleaving multiple pitch relationships and registers during the learning processing thus reinforces the learned label for musical sounds across multiple contexts, promoting greater more effective learning. Future work should investigate whether interleaving other musical parameters during an interval learning paradigm (e.g., changes in instrumental timbre) would yield even more robust effects and efficient learning than the changes in register employed presently.

Our paradigm could also be extended to explore novel pitch learning in domains other than music. For example, the effects of spacing and interleaving could be explored in the learning of lexical pitch patterns of tonal languages (e.g., Mandarin Chinese). Unlike English, in these languages, changes in pitch at the syllable level signal word meaning and hence, are entirely novel to nonnative speakers. Future work could thus examine the role of spacing and interleaving in learning the important components of a second language and in maximizing the speed of its acquisition.

6. ACKNOWLEDGMENTS

Our thanks to the University of Memphis for support and funding of this research.

7. REFERENCES

[1] Bidelman, G.M. and Heinz, M.G. 2011. Auditory-nerve responses predict pitch attributes related to musical consonance-dissonance for normal and impaired hearing. *The Journal of the Acoustical Society of America*. 130, 3 (2011), 1488.

[2] Bidelman, G.M. and Krishnan, A. 2009. Neural correlates of consonance, dissonance, and the hierarchy of musical pitch in the human brainstem. *The Journal of Neuroscience*. 29, 42 (2009), 13165–13171.

[3] Draney, K.L. et al. 1995. A measurement model for a complex cognitive skill. *Cognitively diagnostic assessment*. P.D. Nichols et al., eds. 103–125.

[4] Fujioka, T. et al. 2004. Musical training enhances automatic encoding of melodic contour and interval structure. *Journal of cognitive neuroscience*. 16, 6 (2004), 1010–1021.

[5] Hannon, E.E. and Trainor, L.J. 2007. Music acquisition: effects of enculturation and formal training on development. *Trends in Cognitive Sciences*. 11, 11 (2007), 466–472.

[6] Jeffries, T.B. 1967. The Effects of Order of Presentation and Knowledge of Results on the Aural Recognition of Melodic Intervals. *Journal of Research in Music Education*. 15, 3 (1967), 179.

[7] Kameoka, A. and Kuriyagawa, M. 1969. Consonance theory part I: Consonance of dyads. *The Journal of the Acoustical Society of America*. 45, 6 (1969), 1451–1459.

[8] Krumhansl, C.L. 1990. *Cognitive foundations of musical pitch*. Oxford University Press, USA.

[9] Levelt, W.J.M. et al. 2011. Triadic Comparisons of Musical Intervals. *British Journal of Mathematical and Statistical Psychology*. 19, 2 (2011), 163–179.

[10] McDermott, J.H. et al. 2010. Individual differences reveal the basis of consonance. *Current Biology*. 20, 11 (2010), 1035–1041.

[11] McDermott, J.H. et al. 2010. Individual differences reveal the basis of consonance. *Current Biology*. 20, 11 (2010), 1035–1041.

[12] Moreno, S. et al. 2011. Short-term music training enhances verbal intelligence and executive function. *Psychological science*. 22, 11 (2011), 1425–1433.

[13] Moreno, S., and Bidelman, G.M.S. and Research. Understanding neural plasticity Hearing, and cognitive benefit through the unique lens of musical training. *Hearing Research*.

[14] Mozer, M.C. et al. 2009. Predicting the optimal spacing of study: A multiscale context model of memory. *Advances in Neural Information Processing Systems*. Y. Bengio et al., eds. NIPS Foundation. 1321–1329.

[15] Pavlik Jr., P.I. et al. 2009. Performance Factors Analysis -- A New Alternative to Knowledge Tracing. *Proceedings of the 14th International Conference on Artificial Intelligence in Education*. V. Dimitrova and R. Mizoguchi, eds. 531–538.

[16] Pavlik Jr., P.I. and Anderson, J.R. 2005. Practice and forgetting effects on vocabulary memory: An activation-based model of the spacing effect. *Cognitive Science*. 29, 4 (2005), 559–586.

[17] Rubin, D.C. and Wenzel, A.E. 1996. One hundred years of forgetting: A quantitative description of retention. *Psychological Review*. 103, 4 (1996), 734–760.

[18] Samplaski, A. 2005. Interval and interval class similarity: results of a confusion study. *Psychomusicology: Music, Mind & Brain*. 19, 1 (2005), 59–74.

[19] Schellenberg, E.G. 2005. Music and cognitive abilities. *Current Directions in Psychological Science*. 14, 6 (2005), 317–320.

[20] Schellenberg, E.G. and Trainor, L.J. 1996. Sensory consonance and the perceptual similarity of complex-tone harmonic intervals: Tests of adult and infant listeners. *The Journal of the Acoustical Society of America*. 100, (1996), 3321.

[21] Spada, H. and McGraw, B. 1985. The assessment of learning effects with linear logistic test models. *Test design: Developments in psycholgoy and psychometrics*. S. Embretson, ed. Academic Press.

[22] Wang, Y. and Beck, J.E. 2012. Using Student Modeling to Estimate Student Knowledge Retention. *Proceedings of the 5th International Conference on Educational Data Mining*. J. Yacef, K., Zaïane, O., Hershkovitz, H., Yudelson, M., and Stamper, ed. 200–203.

Tuned Models of Peer Assessment in MOOCs

Chris Piech
Stanford University
piech@cs.stanford.edu

Jonathan Huang
Stanford University
jhuang11@stanford.edu

Zhenghao Chen
Coursera
zhenghao@coursera.org

Chuong Do
Coursera
cdo@coursera.org

Andrew Ng
Coursera
ng@coursera.org

Daphne Koller
Coursera
koller@coursera.org

ABSTRACT

In massive open-access online courses (MOOCs), peer grading serves as a critical tool for scaling the grading of complex, open-ended assignments to courses with tens or hundreds of thousands of students. But despite promising initial trials, it does not always deliver accurate results compared to human experts. In this paper, we develop algorithms for estimating and correcting for grader biases and reliabilities, showing significant improvement in peer grading accuracy on real data with 63,199 peer grades from Coursera's HCI course offerings — the largest peer grading networks analysed to date. We relate grader biases and reliabilities to other student factors such as engagement, performance as well as commenting style. We also show that our model can lead to more intelligent assignment of graders to gradees.

1. INTRODUCTION

The recent increase in popularity of massive open-access online courses (MOOCs), distributed on platforms such as Udacity, Coursera and EdX, has made it possible for anyone with an internet connection to enroll in free, university level courses. However while new web technologies allow for scalable ways to deliver video lecture content, implement social forums and track student progress in MOOCs, we remain limited in our ability to evaluate and give feedback for complex and often open-ended student assignments such as mathematical proofs, design problems and essays. Peer assessment — which has been historically used for logistical, pedagogical, metacognitive, and affective benefits [18] — offers a promising solution that can scale the grading of complex assignments in courses with tens or even hundreds of thousands of students.

Initial MOOC-scale peer grading experiments have shown promise. A recent offering of an online Human Computer Interaction (HCI) course demonstrated that on average, student grades in a MOOC exhibit some agreement with staff-given grades [13]. Despite their initial successes, there remains much room for improvement. It was estimated that 43% of student submissions in the HCI course were given a grade that fell over 10 percentage points from a corresponding staff grade, with some submissions up to 70pp from staff given grades. Thus a critical challenge lies in how to reliably obtain accurate grades from peers.

In this paper, we present the largest peer grading networks analysed to date with over $63,000$ peer grades. Our central contribution is to use this unprecedented volume of peer assessment data to extend the discourse on how to create an effective grading system. We formulate and evaluate intuitive probabilistic peer grading models for estimating submission grades as well as grader biases and reliabilities, allowing ourselves to compensate for grader idiosyncrasies. Our methods improve upon the accuracy of baseline peer grading systems that simply use the median of peer grades by over 30% in root mean squared error (RMSE).

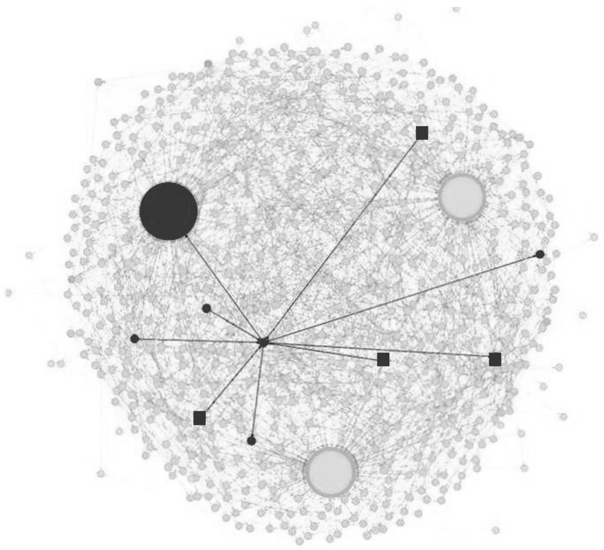


Figure 1: **Peer-grading network: Each node is a learner with edges depicting who graded whom. Node size represents the number of graders for that student. The highlighted learner shown above graded five students (circular nodes) and was in turn graded by four students (square nodes).**

In addition to achieving more accurate scoring for peer grading, we also show how fair scores (where our system arrives at a similar level of confidence about every student's grade) can be achieved by maintaining estimates of uncertainty of a submission's grade.

Finally we demonstrate that grader related quantities in our statistical model such as bias and reliability have much to say about other educationally relevant quantities. Specifically we explore summative influences: what variables correspond with a student being a better grader, and formative results: how peer grading affects future course participation. With the large amount of data available to us, we are

Table 1: Data Sets

	First HCI	Second HCI
Students	3,607	3,633
Assignments	5	5
Submissions	6,702	7,270
Peer Grades	31,067	32,132

able to perform detailed analyses of these relationships that would have been difficult to validate with smaller datasets. These results have notable relevance to the study of education in that the relationships observed depict hypotheses for the dynamics of students grading.

Because peer grading is structurally similar in both MOOCs and traditional brick and mortar classrooms, these results shed light on best practices across both mediums. At the same time, our work helps to describe the unique dynamics of peer assessment in a very new setting — one which may be part of a future with cheaper, more accessible education.

2. DATASETS

In this work, we use datasets collected from two consecutive Coursera offerings of Human Computer Interaction (HCI), taught by Stanford professor Scott Klemmer. The HCI courses used a calibrated peer grading system [17] in order to assess weekly student submissions for assignments which covered a number of different creative design tasks for building a web site. On every assignment, each student evaluated five randomly selected submissions (one of which was a "ground truth" submission, discussed below) based on a rubric[1], and in turn, was evaluated by four classmates. The final score given to a submission was determined as the median of the corresponding peer grades.[2] Peer grading was anonymized so that students could not see who they were evaluating, or who their evaluators were.

After the first offering (HCI1), the peer grading system was refined in several ways. Among other things, HCI2 featured a modified rubric that addressed some of the shortcomings of the original peer grading scheme and peer graders were divided into language groups (English and Spanish). Counting just those who submitted at least one assignment in the English offerings of the class, there were 3,607 students from the first offering (HCI 1) and 3,633 students from the second offering (HCI 2). These students came from diverse backgrounds (with a majority of students from outside of the United States). Collectively, these 7,240 students from around the world created 13,972 submissions, receiving 63,199 peer grades in total. See Table 1 for a summary of the dataset. To prevent overfitting our models to one instance of the class we used the data from HCI2 as a hold out set.

The software for the peer grading framework used by the HCI courses was designed to accommodate experimental validation of peer grading. A small number (3-5) of submissions for each assignment were set aside to serve as "ground truth." For each assignment, every student was given a ground truth submission to grade. Since there were thousands of students in the class, the ground truth submissions were "super-graded" (they had, on average, 160 assessments). Of note, the students were not told that one of the submissions they were assigned to mark belonged to the ground truth set. We hypothesized that by that law of large numbers, the mean of hundreds of student grades should tend towards the correct peer grade. In addition to having hundreds of student grades, the ground truth submissions were also evaluated by volunteer staff graders. However upon deeper analysis it seemed that the mean student grade was more consistently accurate with respect to the rubric than the volunteer staff grade which was not exempt from interrater inconsistency. In order to measure something as complex as a student's grade, it is more useful to take the average of hundreds of evaluations. Figure 1 shows the network of gradee-grader relationships on Assignment 5 of HCI1, where the three super-graded ground truth submissions are clearly visible.

3. PROBABILISTIC MODELS

The ideal peer grading system for a MOOC should satisfy the following desiderata: it should (1) provide highly reliable/accurate assessment, (2) allocate a balanced and limited workload across students and course staff, (3) be scalable to class sizes of tens or hundreds of thousands of students, and (4) apply broadly to a diverse collection of problem settings. In this section we discuss a number of ways to formulate a probabilistic model of peer grading to address these desiderata. The models that we introduce allow for us to algorithmically compensate for factors such as grader biases and reliabilities while maintaining estimates of uncertainty in a principled way. Through our paper, we will use v and u to refer to students in the peer grading network (each student is also a grader).

Our models assume the existence of the following quantities which are either observed or latent (unobserved) variables.

- **True scores:** We assume that every student u is associated with a submission that has a *true underlying score*, denoted s_u, which is unobserved and to be estimated.
- **Grader biases:** Every grader v is associated with a bias, $b_v \in \mathbb{R}$. These bias variables reflect a grader's tendency to either inflate or deflate her assessment by a certain number of percentage points.
- **Grader reliabilities:** We also model grader reliability, $\tau_v \in \mathbb{R}^+$, reflecting how close on average a grader's peer assessments tend to land near the corresponding submission's true score after having corrected for bias. Reliabilities will always correspond to the inverse variance of a normal distribution.
- **Observed grades:** Finally, $z_u^v \in \mathbb{R}$ is the observable score given by grader v to submission u.

Below we present, in order of increasing complexity, three statistical models that we have found to be particularly effective.

[1] See the Appendix: `http://stanford.edu/~cpiech/bio/papers/appendices/edm13_appendix.pdf`

[2] Our description is somewhat of a simplification — students also performed self-assessments and were given the higher of the median and their self grade provided that the two were within five percentage points of each other. We did not consider self assessments in this work.

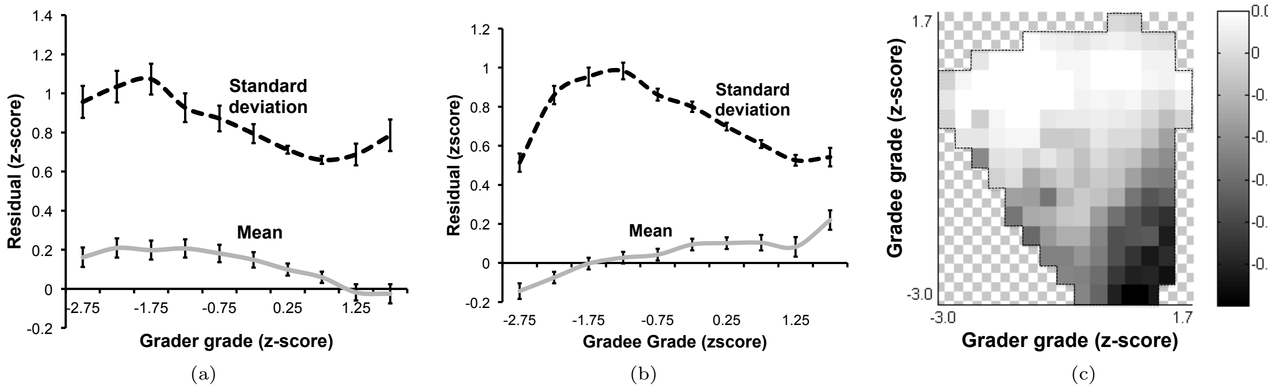


Figure 2: (a) The relationship between a grader's homework performance (her grade) and statistics (mean/standard deviation) of grading performance (residual from true grade). (b) The relationship between a gradee's homework performance against statistics of assessments for her submissions. (c) Visualization of all three variables simultaneously, where intensity reflects the mean residual z-score. Empty boxes mean that there is not enough data available to compute a reliable estimate.

3.1 Model PG$_1$ (Grader Bias and Reliability)

Our first model, **PG**$_1$ puts prior distributions over the latent variables and assumes for example that while an individual grader's bias may be nonzero, the average bias of many graders is zero. Specifically,

$$\begin{aligned}
\text{(Reliability) } & \tau_v \sim \mathcal{G}(\alpha_0,\, \beta_0) \text{ for every grader } v,\\
\text{(Bias) } & b_v \sim \mathcal{N}(0,\, 1/\eta_0) \text{ for every grader } v,\\
\text{(True score) } & s_u \sim \mathcal{N}(\mu_0,\, 1/\gamma_0) \text{ for every user } u, \text{ and}\\
\text{(Observed score) } & z_u^v \sim \mathcal{N}(s_u + b_v,\, 1/\tau_v),\\
& \text{for every observed peer grade.}
\end{aligned}$$

$\mathcal{G}$ refers to a gamma distribution. The variables α_0, β_0, η_0, μ_0 and γ_0 are hyperparameters for prior beliefs of each variable distribution. In our experiments, we also consider a simplified version of Model **PG**$_1$ in which the reliability of every grader is fixed to be the same value. We refer to this simpler model in which only the grader biases are allowed to vary as **PG**$_1$**-bias**.

3.2 Model PG$_2$ (Temporal Coherence)

The priors for the grader variables can play a particularly important role in the above model due to the fact that we typically only have about 4-5 grades to estimate the bias and reliability of each grader. A simple way to obtain more data per grader is to leverage observations made about the grader from previous assignments. To pose a model, we must understand the relationship of a grader's bias and reliability at homework T to that at homework T'. Is it the same or does it change over time?

To answer this question, we examine the correlation between the estimated biases from Model **PG**$_1$ using the HCI1 dataset (see Section 2). Between consecutive assignments, a grader's biases have a Pearson correlation of 0.33 which represents a utilizable consistency. Grader reliability, on the other hand, has a low correlation. We therefore posit Model **PG**$_2$ which allows for grader biases at homework T to depend on those at homework $T-1$ (and implicitly, on all prior homeworks). Specifically, Model **PG**$_2$ assumes:

$$\begin{aligned}
\tau_v^{(T)} &\sim \mathcal{G}(\alpha_0, \beta_0) \text{ for every grader } v,\\
b_v^{(T)} &\sim \mathcal{N}(b_v^{(T-1)}, 1/\omega_0) \text{ for every grader } v,\\
s_u^{(T)} &\sim \mathcal{N}(\mu_0, 1/\gamma_0) \text{ for every user } u, \text{ and}\\
z_u^{v,(T)} &\sim \mathcal{N}(s_u^{(T)} + b_v^{(T)}, 1/\tau_v^{(T)}),\\
&\quad \text{for every observed peer grade.}
\end{aligned}$$

Model **PG**$_2$ requires that we normalize grades across different homework assignments to a consistent scale. In our experiments, for example, we have noticed that the set of grader biases had different variances on different homework assignments. Using a normalized score (z-score), however, allows us to propagate a student's underlying bias while remaining robust to assignment artifacts.

Note that while a model which captures the dynamics of true scores and reliabilities across assignments can be similarly imagined, we have focused only on the dynamics of bias for this work (which contributes the most towards improved accuracy while still being equitable).

3.3 Model PG$_3$ (Grader/Student Interplay)

A unique aspect of peer grading is that graders are themselves students with submissions being graded. Consequently, it is of interest to understand and model the relationship between one's grade and one's grading ability — for example, knowing that a student scored well on his assignment may be cause for placing more trust in that student as a grader.

In Figure 2, we show experiments exploring the relationships between the grader specific latent variables. In particular, we observe that high scoring students tend to be somewhat more reliable as graders (see details of the experiment in Section 4). Model **PG**$_3$ formalizes this intuition by allowing the reliability of a grader to depend on her own grade:

$$\begin{aligned}
b_v &\sim \mathcal{N}(0,\, 1/\eta_0) \text{ for every grader } v,\\
s_u &\sim \mathcal{N}(\mu_0,\, 1/\gamma_0) \text{ for every user } u, \text{ and}\\
z_u^v &\sim \mathcal{N}\left(s_u + b_v,\, \frac{1}{\theta_1 s_v + \theta_0}\right),\\
&\quad \text{for every observed peer grade.}
\end{aligned}$$

Note that Model **PG**$_3$ extends **PG**$_1$ by introducing new dependencies, allowing us to use a student's submission score to estimate her grading ability. At the same time Model

$\mathbf{PG}_3$ is more constrained, forcing grader reliability to depend on a single parameter instead of being allowed to vary arbitrarily, and thus prevents our model from overfitting.

Ethics and Incentives. If we are to use probabilistic inference to score students in a MOOC, the end goal could not simply be to optimize for accuracy. We must also consider fairness when it comes to deciding what variables to include in the model. It might be tempting, for example, to include variables such as race, ethnicity and gender into a model for better accuracy, but almost everyone would agree that these factors could not be fairly used within a scoring mechanism even if they improved prediction accuracy. Another example might be to model the temporal coherence of student grades (we observe a particularly strong temporal correlation between students' grades — with 0.46 Pearson coefficient — of consecutive homework assignments). But incorporating this temporal coherence for students scores into a scoring mechanism would not allow for students to be given a "clean slate" on each homework.

An interesting but subtle facet of $\mathbf{PG}_3$ is that by modelling a correlation between grader reliability and how well the grader did on the assignment, not only does getting a better grade on an assignment influence the model into thinking a particular student is a more reliable grader. It is also the case that if a student is a more reliable grader, the model is influenced into thinking that the student did better on the assignment. This relationship would allow for a student to "game" the mechanism into believing that they did better by grading as accurately as possible. Thus $\mathbf{PG}_3$ may in fact incentivize good grading. Giving students bonus points for better grading is not a new idea. However the nuance of $\mathbf{PG}_3$ is that these bonus points are justified in a statistical sense.

3.4 Inference and evaluation.

Given a probabilistic model of peer grading such as those discussed above, we would like to infer the values of the unobserved variables such as the true score of every submission, or the bias and reliability of each student as a grader. Inference can be framed as the problem of computing the posterior distribution over the latent variables conditioned on all observed peer grades.

Computing this posterior is nontrivial, since all of the variables are correlated with each other. For example, having good estimates of the biases of all of the graders to submission u would allow us to better estimate u's true score, s_u. However to estimate each bias b_v, we would have to have good estimates of the true scores of *all* of the submissions graded by v. We must therefore reason circularly.

To address this apparent chicken and egg problem, we turn to simple approximate inference methods. In the experiments reported in Section 4, we use Gibbs sampling [7], which produces a collection of samples from the (approximate) desired posterior distribution. These samples can then be used to estimate various quantities of interest. For example, given samples $s_u^1, s_u^2, \ldots, s_u^T$ from the posterior distribution over the true score of submission u, we estimate the true score as: $\hat{s}_u \equiv \frac{1}{T}\sum_{t=1}^{T} s_u^t$. We can also use the samples to quantify the uncertainty of our prediction by estimating the variance of the samples from the posterior, which we use in Section 4 when we examine peer grading efficiency. Note that while the ordinary Gibbs sampling algorithm can be performed in "closed form" for Models $\mathbf{PG}_1$ and $\mathbf{PG}_2$, Model $\mathbf{PG}_3$ requires numerical approximation due to the coupling of a submission's true score s_u with that of its grader, s_v. Visually we observe rapid mixing for our Gibbs chains, and in the experiments shown in Section 4, we use 800 iterations of Gibbs sampling, discarding the initial 80 burn-in samples.

Expectation-maximization (EM) is an alternative approximate inference approach, where we treat the true scores and grader biases as parameters and then use an iterative coordinate descent based algorithm to obtain point estimates of parameters. In practice, we find that both the Gibbs and EM approaches behave similarly. In general EM has the advantage of being significantly faster while obtaining posterior credible intervals is more natural using Gibbs. On the peer grading dataset the two methods produce analogous results. For example, $\mathbf{PG}_1$ with Gibbs and EM have RMSE scores of 5.42 and 5.43 on the first dataset respectively and with Gibbs running in roughly 5 minutes and EM running in 7 seconds. We refer the reader to the appendix for the full algorithmic details of Gibbs as well as EM.

Evaluation. To measure peer grading accuracy, we repeatedly simulate what score would have been assigned to each ground truth submission had it been peer graded. Our evaluation of how well we would have graded a single ground truth submission uses a two step methodology (based on the evaluation method of [13]): (1) We run inference using all of our data, except the peer grades of the ground truth submission being evaluated. This gives us an estimate of each grader's biases and reliabilities as well as model priors that were independent of the submission being evaluated. (2) We run simulations where we sampled four student assessments randomly from the pool of peer grades for the ground truth submission, estimate the submission's grade using the sample of assessments and record the residual between our estimated grade and the "true" grade. For each ground truth submission we run 3000 such simulations, from which we report the RMSE, the number of simulations which fell within five, and ten percentage points of the true score, the average standard deviation of the errors over each ground truth and the worst misgrade that the simulations produced.

We compare each of our probabilistic models to the grade estimation algorithm used on Coursera's platform. In the baseline model, the score given to students is the median of the four peer grades they received. Specifically, the baseline estimation does not take into account an individual grader's biases or reliabilities. Nor does it incorporate prior knowledge about the distribution of true grades.

4. EXPERIMENTAL RESULTS

4.1 Accuracy of reweighted peer grading

Using probabilistic models leads to substantially higher grading accuracy. In our experiments we are able to reduce the RMS error on our prediction of the ground truth grade by 33% from 7.95 to 5.30. Similarly, on the second offering of the course we were able to reduce error by 31% from 6.43 to 4.73. For the second offering, this means that the num-

Table 2: Comparison of models on the two HCI courses

	HCI 1					HCI 2				
	Baseline	**PG_1-bias**	**PG_1**	**PG_2**	**PG_3**	Baseline	**PG_1-bias**	**PG_1**	**PG_2**	**PG_3**
RMSE	7.95	5.42	5.40	5.40	**5.30**	6.43	4.84	4.81	4.75	**4.73**
% Within 5pp	51	69	69	**71**	70	59	72	73	73	**74**
% Within 10pp	81	92	94	94	**95**	88	96	96	**97**	**97**
Mean Std	7.23	5.00	4.96	4.92	**4.77**	6.19	4.57	**4.52**	4.53	**4.52**
Worst Grade	-43	-34	**-30**	-32	**-30**	-36	-26	-26	**-25**	-26

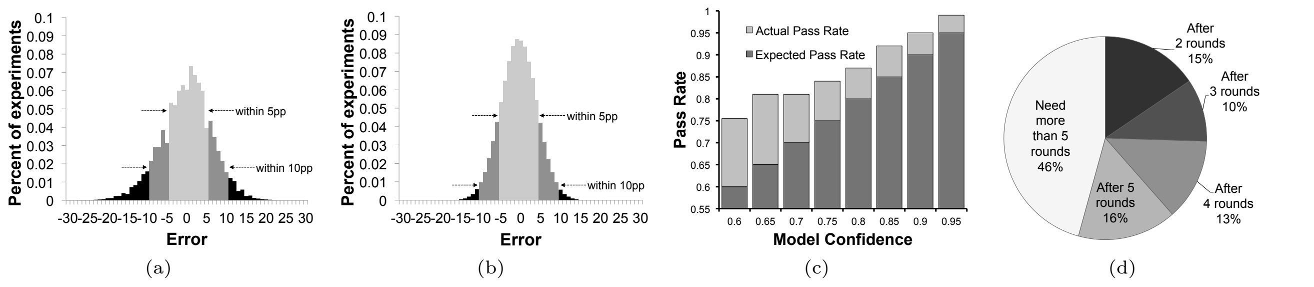


Figure 3: (a) Histogram of errors made using the baseline (median) scoring mechanism. (b) Histogram of errors using PG_3. (c) A comparison of model confidence (x-axis) and actual success rate of predictions (y-axis), where being above the diagonal (dark bars) is better. (d) Number of submissions for which our model can declare "confidence" after K rounds of grading.

ber of students who received grades within 10 percentage points (pp) of their grade increased from 88% to 97%. Figures 3(a), 3(b) show the effect of using Model $\mathbf{PG_3}$ as a scoring mechanism on the histogram of grading errors and Table 2 shows the complete results for each model. Due to course improvements, we observe that students in HCI2 were significantly more consistent as graders compared to students in HCI1. However, we remark that every one of our models run on HCI1 outperforms the baseline grading system run on HCI2 with respect to every metric, indicating that the best gains in peer grading are likely to come from both an improved class design as well as statistical modeling.

Our results show that Models $\mathbf{PG_3}$ (with coupled grader score and reliability) and $\mathbf{PG_2}$ (with temporal coherence) yield the best results, with Model $\mathbf{PG_3}$ outperforming the other models with respect to most metrics. But the single change that provides the most significant gains in accuracy is obtained by estimating each grader's bias (Model **PG_1-bias**). This simple model is responsible for 95% of our reduction in RMSE. The other changes all contribute comparatively smaller improvements to a more accurate model.

Our evaluation setup also allows us to test how accurate we would have been, had we had more than four grades per student. If the class had increased the number of grades that each student received to five (instead of four), our model could reduce RMSE error on the first and second offering of HCI to 4.19 and 4.36 respectively.

Surprisingly while modeling grader bias is particularly effective, modeling grader reliability does little to improve our performance. To dig deeper into this result we test our model on a synthetic dataset — one generated exactly from Model $\mathbf{PG_1}$. When using this synthetic data with only four grades per student it is difficult for the model to correctly estimate grader reliability. Modeling variance for each grader only seems to have a notable impact when students grade many assignments (more than 10). This experiment also suggests why $\mathbf{PG_3}$ is more useful than $\mathbf{PG_1}$. Though $\mathbf{PG_1}$ contains more expressive power than $\mathbf{PG_3}$, estimating only two parameters for grader reliability (θ_0 and θ_1) is more statistically tractable with only four grades per student than estimating a reliability, τ_v, for each grader.

4.2 Fairness and efficiency in peer grading

One of the advantages of using a probabilistic model for peer grading is that we can obtain a belief distribution over grades (as opposed to a single score) for each student. These distributions give us a natural way of calculating how confident the model is when it predicts a grade for a student which opens up the possibility of a more equitable allocation of graders. For example, at a given point midway through the peer grading process, our model may be highly confident in its prediction for a given student's score, but very unsure in its prediction for another student. In this situation, to ensure that each student gets fair access to quality feedback, we could reassign graders to gradees such that submissions which have low-confidence scores are given to more and/or better graders.

The first step towards more fair allocation of grades is to ask ourselves: how accurate are our estimates of confidence? For example, we would like to know how to interpret what it means in practice when our Bayesian model is 90% confident that its prediction of a learner's true score is within 10pp of the actual true score.

To better understand our confidence estimates, we run the following experiment: We first performed a large number of peer grading simulations on ground truth. From each simulation we calculate how confident our model is that the grade it predicts is within 5%, 7%, and 10%, of the true score, respectively. We then bin the estimated confidences into ranges 0-5%, 5-10%, etc. After collecting over 5000 predictions per range, we test the pass rate of each range.

For example, suppose we select four assessments of the same ground truth submission in a simulation. If our model reports a 72% confidence — based on those four assessments — that our predicted grade is within 5pp of the true score, we add that estimate to the set of predictions in the 70% to 75% confidence range. When we test this confidence range the example prediction "passes" if its estimate is in fact within 5pp of the ground truth score.

One worry is that our model might be overconfident about its predictions even when wrong. However the results, shown in Figure 3(c), demonstrate that our confidence estimates are on the conservative side — for example over 95% of the time that our model claims it is between 90 and 95% confident of a prediction, the model's estimate is correct.

Since we have reason to believe that our confidence values are accurate, we can employ our posterior belief distributions to better allocate grades. To understand how much benefit we could get out of improved grade allocation, we estimate at what point in the grading process we were confident about each submission's score. For each homework assignment, we simulate grading taking place in rounds. In the first round, we only include the first grade submitted by each grader (which may have been a ground truth grade). In the second round, we included the first two, etc. For each round we run our model using the corresponding subset of grades and count the number of submissions for which we are over 90% confident that our predicted grades were within 10pp of the student's true grade.

After only two rounds of grading we are highly confident in our estimated grade for 15% of submissions (this generally means that the submission has a grade close to the assignment mean, and has two similar grades from graders). Figure 3(d) shows how the set of confident submissions grows over the grading rounds. Our experiment demonstrates a clear opportunity for grades to be reallocated as well as a pressing need for some submissions to get more grades. For 54% of students, after all rounds, we are still unsure of their submission's true score.

4.3 Graders in the context of the MOOC

Applying probabilistic models to peer grading networks allows us to increase our grade accuracy and better allocate what submissions students should grade. Another product of our work is an assignment — with a belief distribution — for a true score, grader bias and grader reliability for each student. We can use this large dataset to derive new understanding about peer grading as both a formative and summative assessment. We focus our investigation on two questions, (1) what factors influence how well a student grades? and (2) how does grading ability affect future class performance in a MOOC?

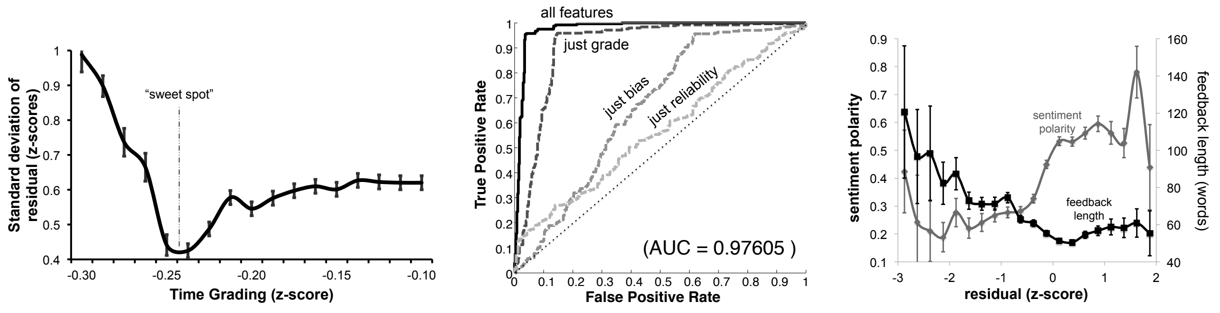
Influential factors for grader ability. To explore what factors influence how well a student grades we compare grading residual (how far off a grader's score is from our model estimated true score) to: time spent grading, grader grade, and gradee grade.

Time spent grading shows a particularly interesting trend (Figure 4(a)). As hypothesized, students that "snap grade" their peers' work (the students whose time spent grading has a z-score of less than -0.30), are both unreliable (the variance of their residuals is over 1 standard deviation away from the gradee's true score) and tend to slightly inflate grades. More surprising is that over the tens of thousands of grades, there is a "sweet spot" of time spent grading. Students who grade assessments with a time that has a z-score of around -0.25 have significantly lower residual standard deviations (with p-value < 0.001, diff $= 0.3$ standard deviations) than students who take a long time to grade (i.e., time spent grading has a z-score $>$ -0.20). This sweet spot is only visible when we look at normalized grading times. For most assignments in the HCI class, the sweet spot corresponds to around 20 minutes grading. This may reflect both that with any less time a grader does not have enough of a chance to fully examine her gradee's work, and that a long grading session may mean that the grader had trouble understanding some facet of the submission.

Examining the relationship between grader grade, gradee grade and how they affect the residual also shows a set of notable trends. Graders that score higher on assignments have close to monotonically decreasing biases (Figure 2(a)). Getting a better grade on the homework in general makes students more reliable graders; with the notable exception that the students that get the best grades (+1.75 z-score) are not as accurate as the students who do very well (+.75 z-score, $p = 0.04$). The superlative submissions — both the best and the worst — are the easiest to grade, and the submissions which are one standard deviation below the mean are the hardest (Figure 2(b)). Finally, our results show that students are least biased when grading peers with similar score (Figure 2(c)). The best students significantly downgrade the worst submissions and the worst students notably inflate the best submissions.

In addition to numerical scores, graders were asked to provide feedback in the form of free form text comments to their gradees. In order to understand the relationship between grading performance and commenting style, we compare grading residual against the comment length as well as sentiment polarity of the comment (Figure 4(c)). To measure the polarity of a comment, we use the sentiment analysis word list from [15] and implement a simple sentiment analyzer that returns a (normalized) polarity score (positive or negative) proportional to the sum of word valences over the comment. For both comment length and polarity, we filter out all non-English words. We observe that comments that correspond to larger negative residuals are typically significantly longer, suggesting perhaps that students write more about the weaknesses of a submission than strong points. That being said, we observe that overall, the comments mostly range in polarity from neutral to quite positive, suggesting that rather than being highly negative to some submissions, many students make an effort to be balanced in their comments to peers.

Grader ability and future performance. We also tested what signal grading ability has with predicting future participation. Based on the theory that the best graders are intrinsically motivated, we hypothesized that being a reliable grader would add a different dimension of information to a student's engagement which we should be able to use to



(a) (b) (c)

Figure 4: (a) Grader consistency (measured using standard deviation of grading residual) as a function of time spent grading. (b) ROC curve comparing performance (with linear SVM) at predicting future class participation given a student's grade, bias, reliability or all three. (c) Commenting style (length of comment and sentiment polarity) as a function of grading residual.

better predict future engagement. We tested this hypothesis by constructing a classification task in which we predict whether a student would participate in the next assignment (or conversely which students would "drop out"). In addition to the student's grade, we experimented with including grader bias and reliability as features in a linear classifier. Our results (Figure 4(b)) show that including grader bias and reliability improved our predictive ability by 5pp from an area under the curve (AUC) score of 0.93 to an AUC of 0.98. Properties about how a student grades, captures a dimension of their engagement which is missed by their assignment grade.

5. RELATED WORK

The statistical models we present in this paper are part of a long tradition of models which have been proposed for the purposes of aggregating information from noisy human labelers or workers. Many of these works adapt classical item-response theory (IRT) models [3] to the problem of "grading without an answer key" and appear in the literature from educational aptitude testing [10, 16, 14], to cultural anthropology [4, 12], and more recently to HCI in the context of human computation and crowdsourcing [19]. In educational testing, for example, Johnson [10] and Rogers et al. [16] propose models for combining human judgements of essays. These papers analyze *dedicated human graders* who each evaluated hundreds of essays, allowing for a rich model to be fitted on a per-grader basis. In contrast, with peer grading in MOOCs, each student only assesses a handful of assignments, necessitating more constrained models.

In a recent paper, and in a setting perhaps most similar to our own, Goldin et al. [9, 1, 8] use Bayesian models for peer grading in a smaller scale classroom setting. As in our own work, [8] posits a grader bias, and in fact incorporates rubric-specific biases, but does not consider many of the issues raised here such as grading task reallocation or the relationship between grader bias and student engagement, for example.

One of the central themes of the crowdsourcing literature, that of balancing label accuracy against labor cost, is one which MOOC peer grading systems must contend with as well. In such problems, one typically receives a number of noisy labels (for example in an image tagging task) and the challenge lies in (1) resolving the "correct" label (often discrete, but sometimes continuous) and (2) deciding whether to hire more labelers for a given task. Explosion of interest in recent years has led to widespread applications of crowdsourcing [2, 11]. For example in image annotation, Whitehill et al. [19] present a method similar to our own in which they model discrete "true image labels" as well as labeler accuracy. While our work draws from the crowdsourcing literature, the problem of peer grading is unique in several ways. For example, the fact that the graders are also gradees in peer grading is quite different from typical crowdsourcing settings in which there is a dichotomy between the labelers and the items being labeled, and motivates different models (such as Model $\mathbf{PG}_3$). In crowdsourcing applications, the end goal often lies in determining the true labels rather than to understand anything about the labelers themselves, whereas in peer grading, as we have shown, the insights that we can glean about the graders have educational value.

A similar problem to peer-grading is the paper assignment problem for the peer review process in academic conferences. While related in that the central challenge of both problems involves fusing disparate human opinions about open-ended creative work, many of the specific challenges are distinct. For one, side information plays a much larger role in peer review, where conference chairs typically rely heavily on personal or elicited knowledge of reviewer expertise or citation link structure to assign reviewer roles [5]. Peer grading on the other hand seems less sensitive to personal preferences, where a single submission should be equally well graded by a large fraction of students in the course.

6. DISCUSSION AND FUTURE WORK

Our paper presents methods for making large scale peer grading systems more dependable, accurate, and efficient. We show that there is much to be gained by maintaining estimates of grader specific quantities such as bias and reliability. In addition to improving peer grading accuracy by up to 30%, these quantities give unique insights into peer grading as a formative and summative assessment. Due to these promising results, our implementation is being used and evaluated in the third offering of HCI.

Moreover, trends among the latent variables that are coloured in by our model suggest results that could benefit the understanding of peer grading as a pedagogical tool. For example, understanding the factors that cause the "sweet spot" in grading time that we observed would be helpful in teaching graders. The trend between time spent grading and reliability also raises the question of how to incentivize students to spend enough time grading to provide careful and high quality feedback to their peers, particularly in an open-acess course. Using model $\mathbf{PG}_3$ for scoring, as we discussed, makes a student's score dependent on grading performance, and may be one way to build a justified incentive directly into the scoring mechanism. There are also a number of challenging theoretical open questions on the mechanism design issues behind peer grading which [6] has taken steps to address. Similarly, the subtle patterns between grader score and reliability that are visible given our volume of data add an interesting piece of evidence to explore from an educational perspective. While, it is unsurprising that good students grade better, we also observe that the very best students in the class, were worse graders than students in the 70th percentile. What is the force behind this trend?

There remain a number of issues to be addressed in future work. We have considered the problem of determining which submissions need to be allocated additional graders. However, deciding which grader is best for evaluating a particular submission is an open problem whose solution could depend on a number of variables, from the writing styles of the grader and gradee to their respective cultural or linguistic backgrounds, a particularly important issue for the global scale course rosters that arise in MOOCS. Moreover, in our study we ovserved that the mean of hundreds of students who graded the same assignment was more reliable than volunteer staff grades. This points to a more in-depth investigation into how accurate "expert" grades really are. Finally, it is not clear how to present scores which are calculated by a complicated peer grading model to students. While this communication might be easy when a student's final grade is simply set to be the mean or median of peer grades, does each student need to know the inner workings of a more sophisticated statistical backend? Students may be unhappy with the lack of transparency in grading mechanisms, or on the other hand might feel more satisfied with their overall grade.

As MOOCs become more widespread, the need for reliable grading and feedback for open ended assignments becomes ever more critical. By addressing the shortcomings of current peer grading systems, we hope that students everywhere can get more from peer grading and consequently, more from their free online, open access educational experience.

Acknowledgments

We thank Chinmay Kulkarni and Scott Klemmer for providing assistance with the HCI datasets and Leonidas Guibas and John Mitchell for discussions and support. Jonathan Huang is supported by an NSF Computing Innovations Fellowship.

7. REFERENCES

[1] K. Ashley and I. Goldin. Toward ai-enhanced computer-supported peer review in legal education. In *24th International Conference on Legal Knowledge and Information Systems (JURIX)*, volume 235, 2011.

[2] Y. Bachrach, T. Minka, J. Guiver, and T. Graepel. How To Grade a Test Without Knowing the Answers - A Bayesian Graphical Model for Adaptive Crowdsourcing and Aptitude Testing. In *The 29th Annual International Conference on Machine Learning*, ICML '12, 2012.

[3] F. Baker. *The basics of item response theory.* ERIC Clearinghouse on Assessment and Evaluation, University of Maryland, College Park, 2001.

[4] W. H. Batchelder and A. K. Romney. Test theory without an answer key. *Psychometrika*, 53:71–92, 1988.

[5] L. Charlin, R. Zemel, and C. Boutilier. A framework for optimizing paper matching. In *Proceedings of Uncertainty in Artificial Intelligence (UAI'11)*, 2011.

[6] A. Dasgupta and A. Ghosh. Crowdsourced judgement elicitation with endogenous proficiency. In *Proceedings of the 22th international conference on World wide web. ACM, 2013.*, 2013.

[7] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE PAMI*, (6):721–741, 1984.

[8] I. Goldin. Accounting for peer reviewer bias with bayesian models. In *Proceedings of the Workshop on Intelligent Support for Learning Groups at the 11th International Conference on Intelligent Tutoring Systems*, 2012.

[9] I. M. Goldin and K. D. Ashley. Peering inside peer review with bayesian models. In *Proceedings of the 15th international conference on Artificial intelligence in education*, AIED'11, pages 90–97, Berlin, Heidelberg, 2011. Springer-Verlag.

[10] V. E. Johnson. On bayesian analysis of multi-rater ordinal data: An application to automated essay grading. *Journal of the American Statistical Association*, 91:42–51, 1996.

[11] E. Kamar, S. Hacker, and E. Horvitz. Combining human and machine intelligence in large-scale crowdsourcing. In *In AAMAS*, 2012.

[12] G. Karabatsos and W. Batchelder. Markov chain estimation for test theory without an answer key. *Psychometrika*, 68(3):373–389, 2003.

[13] C. Kulkarni, K. Pang-Wei, H. Le, D. Chia, K. Papadopoulos, D. Koller, and S. R. Klemmer. Scaling self and peer assessment to the global design classroom. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2013.

[14] R. J. Mislevy, R. G. Almond, D. Yan, and L. S. Steinberg. Bayes nets in educational assessment: Where the numbers come from. In *Proceedings of the fifteenth conference on uncertainty in artificial intelligence*, pages 437–446. Morgan Kaufmann Publishers Inc., 1999.

[15] F. Å. Nielsen. A new anew: Evaluation of a word list for sentiment analysis in microblogs. *arXiv preprint arXiv:1103.2903*, 2011.

[16] S. Rogers, M. Girolami, and T. Polajnar. Semi-parametric analysis of multi-rater data. *Statistics and Computing*, 20(3):317–334, July 2010.

[17] A. A. Russell. Calibrated peer review - a writing and critical-thinking instructional tool. *Teaching Tips: Innovations in Undergraduate Science Instruction*, page 54, 2004.

[18] P. M. Sadler and E. Good. The impact of self-and peer-grading on student learning. *Educational assessment*, 11(1):1–31, 2006.

[19] J. Whitehill, P. Ruvolo, T. fan Wu, J. Bergsma, and J. Movellan. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in Neural Information Processing Systems 22*, pages 2035–2043. MIT Press, 2009.

Does Representational Understanding Enhance Fluency – Or Vice Versa? Searching for Mediation Models

Martina A. Rau
Human-Computer Interaction Institute
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213, USA
marau@cs.cmu.edu

Richard Scheines
Department of Philosophy
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213, USA
scheines@cmu.edu

Vincent Aleven
Human-Computer Interaction Institute
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213, USA
aleven@cs.cmu.edu

Nikol Rummel
Institute of Educational Research
Ruhr-Universität Bochum
Universitätsstraße 150
44801 Bochum, Germany
nikol.rummel@rub.de

ABSTRACT

Conceptual understanding of representations and fluency in using representations are important aspects of expertise. However, little is known about how these competencies interact: does representational understanding facilitate learning of fluency (understanding-first hypothesis), or does fluency enhance learning of representational understanding (fluency-first hypothesis)? We analyze log data obtained from an experiment that investigates the effects of intelligent tutoring systems (ITS) support for understanding and fluency in connection-making between fractions representations. The experiment shows that instructional support for both representational understanding and fluency are needed for students to benefit from the ITS. In analyzing the ITS log data, we contrast the understanding-first hypothesis and the fluency-first hypothesis, testing whether errors made during the learning phase mediate the effect of experimental condition. Finding that a simple statistical model does not the fit data, we searched over all plausible causal path analysis models. Our results support the understanding-first hypothesis but not the fluency-first hypothesis.

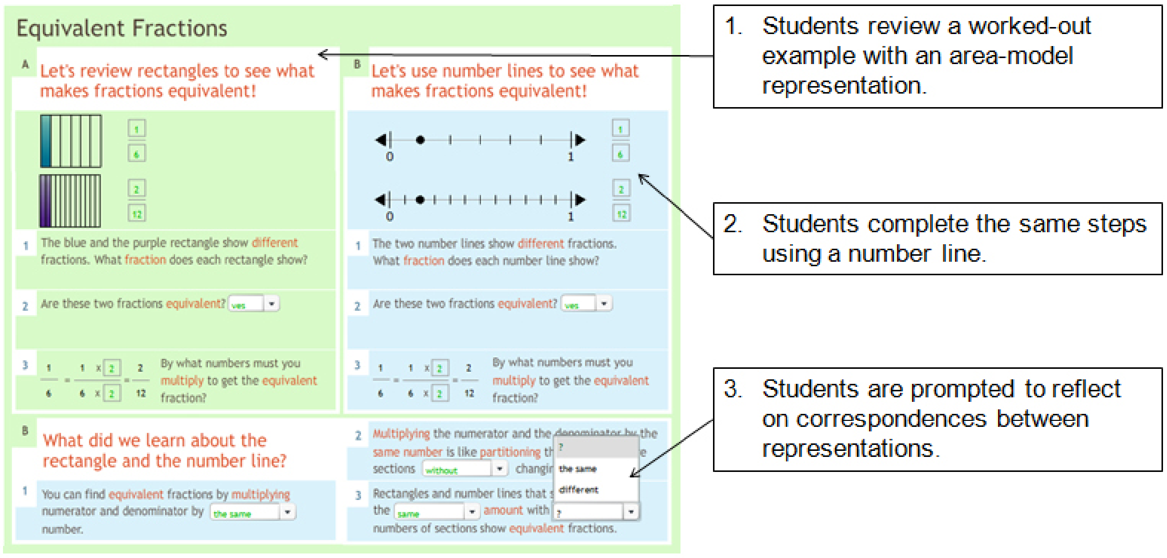
Keywords

Causal path analysis modeling, multiple representations, intelligent tutoring systems.

1. INTRODUCTION

Representational understanding and *representational fluency* are important aspects of learning in any domain [1]. When working with representations (e.g., formulae, line graphs, path diagrams), students need conceptual understanding of these representations (representational understanding). Students also need to use the representations to solve problems fast and effortlessly (representational fluency). Science and mathematics instruction typically employs multiple graphical representations to help students learn about complex domains [2]. For instance, instructional materials for fractions use circle and rectangle diagrams to illustrate fractions as parts of a whole, and number lines to depict fractions in the context of measurement [3-5]. Multiple representations have been shown to lead to better learning than a single representation, provided that students make connections between them [6-7]: to benefit from the multiplicity of representations, students need to conceptually understand how different representations relate to one another, and they need to translate between them [8-11]. Yet, students find it difficult to make these connections [8], and tend not to make them spontaneously [12]. Therefore, they need to be supported in doing so [7]. Based on [1], we distinguish between *representational understanding* as conceptual understanding of connections between different graphical representations, and *representational fluency* as the ability to fast and effortlessly make these connections. To benefit from multiple graphical representations, students need to acquire both representational understanding [8], and they need to develop representational fluency [13].

In the present paper, we use log data obtained from a classroom experiment that uses a successful type of intelligent tutoring system (ITS) to help students learn about fractions while comparing different ways to support *representational understanding* and *representational fluency*. The experiment demonstrates that both instructional support for representational understanding and representational fluency are necessary for students to benefit from multiple graphical representations of fractions [14]. The goal of the present paper is to augment the findings from the traditional analysis of pretest and posttest data by using causal path analysis modeling to analyze mediation effects that can explain the nature of *how* representational understanding and representational fluency interact. Does representational understanding facilitate students' acquisition of representational fluency? Or does representational fluency enhance students' ability to acquire representational understanding? We contrast two competing hypotheses. According to the *understanding-first hypothesis*, representational understanding equips students with the necessary knowledge about structural correspondences between graphical representations and about what differences between the representations are incidental, allowing students to attend to relevant aspects of the graphical representations while developing representational fluency. Therefore, students who receive support for representational understanding should make fewer errors on fluency-building problems compared to students who do not receive support for representational understanding. By contrast, the *fluency-first hypothesis* predicts that representational fluency frees up the cognitive resources that students need to acquire understanding of these connections. Therefore, students who receive fluency-building support should make fewer errors on problems supporting representational understanding compared to students who do not receive fluency-building support. The answer to the question of how acquisition of representational understanding and representational fluency interact has important implications for the instructional design of ITSs and other educational technologies. If representational understanding enhances the acquisition of representational fluency (understanding-first hypothesis), instructional materials should support representational understanding before representational fluency. If, on the other hand, representational fluency facilitates students' acquisition of representational understanding(fluency-first hypothesis), instructional materials should support representational fluency before supporting representational understanding.


Equivalent Fractions

A Let's review rectangles to see what makes fractions equivalent!

1 The blue and the purple rectangle show different fractions. What fraction does each rectangle show?

2 Are these two fractions equivalent? yes

3 By what numbers must you multiply to get the equivalent fraction?

B Let's use number lines to see what makes fractions equivalent!

1 The two number lines show different fractions. What fraction does each number line show?

2 Are these two fractions equivalent? yes

3 By what numbers must you multiply to get the equivalent fraction?

B What did we learn about the rectangle and the number line?

1 You can find equivalent fractions by multiplying numerator and denominator by the same number.

2 Multiplying the numerator and the denominator by the same number is like partitioning th… sections without changi…

3 Rectangles and number lines that s… the same amount with ? numbers of sections show equivalent fractions.

1. Students review a worked-out example with an area-model representation.

2. Students complete the same steps using a number line.

3. Students are prompted to reflect on correspondences between representations.

$\frac{1}{6}$
$\frac{2}{12}$
0
1
$\frac{1}{6} = \frac{1 \times 2}{6 \times 2} = \frac{2}{12}$
?
the same
different

Figure 1. Worked example support for representational understanding: students use a worked example with a rectangle (part A, upper left) to guide their work on a fractions problem with a number line (part B, upper right). At the end (part C, bottom), students are prompted to integrate both representations by responding to drop-down menu questions.

To gain further insights into *how* support for representational understanding and representational fluency affect students' interactions with an ITS for fractions, we employ causal path analysis. In doing so, we contrast mediation models that correspond to the understanding-first hypothesis, and to the fluency-first hypothesis. Specifically, we investigate whether errors that students make during the learning phase mediate the interaction effect between support for representational understanding and representational fluency on students' learning. Our results are in line with the understanding-first hypothesis, but not with the fluency-first hypothesis.

The remainder of this paper is structured as follows. We first describe the ITS that we used to carry out the experimental study. We then provide a brief overview of the experimental design and the results obtained from the analysis of pretests and posttests. The main focus of this paper is on describing the causal path analysis we conducted to investigate the interaction of instructional support for representational understanding and representational fluency on students' learning behaviors as identified by the tutor log data. We end by discussing the implications of our analysis for the instructional design of learning materials, and by outlining open questions that future research should be address.

2. THE FRACTIONS TUTORING SYSTEM

The Fractions Tutor used in the experiment is a type of Cognitive Tutor. Cognitive Tutors are grounded in cognitive theory and artificial intelligence. Cognitive Tutors have been shown to lead to substantial learning gains in a number of studies [15]. We created the Fractions Tutor with Cognitive Tutor Authoring Tools [16]. The design of the tutor interfaces and of the interactions students engage in during problem solving are based on a number of small-scale user studies, a knowledge component model developed based on Cognitive Task Analysis of the learning domain [17], and a series of in vivo experiments [6, 12, 18].

The Fractions Tutor uses multiple interactive graphical representations (circles, rectangles, and number lines) that are typically used in instructional materials for fractions learning [2-3, 5]. The Fractions Tutor covers a comprehensive set of topics ranging from identifying fractions from graphical representations, to equivalent fractions and fraction addition. Taken together, the Fractions Tutor comprises about ten hours of supplemental instructional material. Students solve tutor problems by interacting both with fractions symbols and with the graphical representations. As is common with Cognitive Tutors, students receive error feedback and hints on all steps. In addition, each tutor problem includes conceptually oriented prompts to help students relate the graphical representations to the symbolic notation of fractions.

3. EXPERIMENT

The goal of the experimental study (cf. [14] for a detailed description) was to investigate the hypothesis that students learn more robustly when receiving instructional support for both representational understanding and support for representational fluency. We conducted a classroom experiment with 599 4th- and 5th-grade students from five elementary schools in the United States. Students worked with the Fractions Tutor for about ten hours during their regular mathematics class.

We contrasted two experimental factors. One factor, *support for representational understanding* in making connections had three levels: no support, auto-linked support in which the Fractions Tutor automatically made changes in one representation as students manipulated another, and worked examples. Figure 1 provides an example of the Fractions Tutor problem that uses worked examples (WEs) to support representational understanding. Students used a worked example with a familiar representation as a guide to make sense of an isomorphic problem with a less familiar representation. This factor was crossed with a second experimental factor, namely, whether or not students received *support for representational fluency* in making connections: students had to visually estimate whether different types of graphical representations showed the same fraction. Figure 2 shows an example of a fluency-building problem (FL). Students in all conditions worked on 80 tutor problems: eight problems per topic (e.g., equivalent fractions, addition, subtraction, etc.). In each topic, the first four tutor problems were single-representation problems (i.e., they included only a circle, only a rectangle, or only a number line, and no connection-making support). The last four tutor problems were multiple-representation problems and differed between the experi-

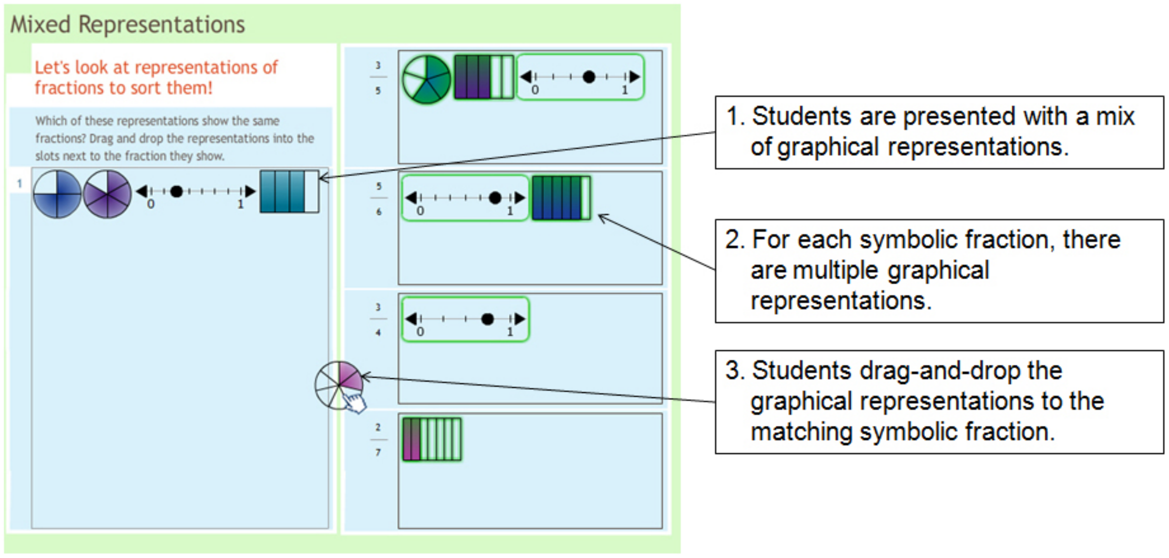

Mixed Representations
Let's look at representations of fractions to sort them!
Which of these representations show the same fractions? Drag and drop the representations into the slots next to the fraction they show.
1. Students are presented with a mix of graphical representations.
2. For each symbolic fraction, there are multiple graphical representations.
3. Students drag-and-drop the graphical representations to the matching symbolic fraction.

Figure 2. Fluency-building support: students sort graphical representations by dragging-and-dropping them into slots that show equivalent fractions.

mental conditions. For instance, students in the worked examples only condition (WE) received four worked examples problems. Students in the fluency-only condition (FL) received four fluency-building problems. Students in the worked examples plus fluency condition (WE-FL) received two worked examples problems, followed by two fluency-building problems. Table 1 illustrates this procedure for two consecutive topics for each of these three conditions. The same sequence of eight problems was repeated for each of the ten topics the Fractions Tutor covered.

Table 1. Problem sequence per condition: for each topic, problems 1-4 (P1-P4) are single-representation problems (S); problems 5-8 are multiple-representation problems: worked examples (WE, blue-underlined) or fluency-building problems (FL, green-italicized).

Cond.	Topic	P1	P2	P3	P4	P5	P6	P7	P8
WE	1	S	S	S	S	WE	WE	WE	WE
	2	S	S	S	S	WE	WE	WE	WE
	…	…							
FL	1	S	S	S	S	FL	FL	*FL*	*FL*
	2	S	S	S	S	FL	FL	*FL*	*FL*
	…	…							
WE-FL	1	S	S	S	S	WE	WE	*FL*	*FL*
	2	S	S	S	S	WE	WE	*FL*	*FL*
	…	…							

Results based on the analysis of pretest, immediate posttests, and delayed posttest (administered one week after the immediate posttest) from 428 students confirmed the hypothesis that a combination of instructional support for representational understanding and representational fluency is most effective: the interaction between support for understanding and fluency was significant, $F(2, 351) = 3.97$, $p < .05$, $\eta_p^2 = .03$, such that students who received both types of support performed best. Worked examples are the more effective type of support for representational understanding, when paired support for representational fluency: within the conditions with support for representational fluency, there was a significant effect of support for representational understanding, $F(2, 343) = 4.34$, $p < .05$, $\eta_p^2 = .07$. However, within the conditions without support for representational fluency, there was no significant effect of support for representational understanding ($F < 1$). Finally, our results show an advantage of the WE-FL condition over the number-line control, $t(115) = 2.41$, $p < .05$, $d = .27$.

The results from the experimental study raise interesting new questions about the relation between representational understanding and representational fluency. It is surprising that there were no significant main effects for support for representational understanding or representational fluency alone; only the combination of both enhanced students' learning from multiple graphical representations. Did support for understanding enable students to benefit from fluency-building support, or vice versa? We address this question in the remainder of this paper.

4. DATA SET

The analyses in this paper are based on the data obtained from the experimental study just described. Students in the experiment received a pretest on the day before they started to work with the Fractions Tutor. The day after students finished working with the Fractions Tutor, they received an immediate posttest. One week after the immediate posttest, students were given a delayed posttest. All three tests were equivalent (i.e., they contained the same items with different numbers). Students worked with the Fractions Tutor for about ten hours and had to complete each tutor problem. All interactions with the Fractions Tutor were logged.

4.1 Selecting Conditions to Include into Causal Path Analysis Modeling

In the light of the interaction effect between support for representational fluency and support for representational understanding through worked examples, the experimental conditions of interest for further analyses are worked example (WE), fluency (FL), and worked examples paired with support fluency (WE-FL). We thus selected these three conditions to include into the causal path analysis model. A total of 190 students were included in the analysis ($n = 59$ in the WE condition, $n = 73$ in the FL condition, and $n = 58$ in the WE-FL condition). Table 2 shows the means and standard deviation of students' performance on pretest, immediate and delayed posttest by condition.

4.2 Defining Mediation Variables

As the goal was to investigate whether support for representational understanding helps students benefit from support for representational fluency or vice versa, we compared students' *performance on worked-example problems* (i.e., support for representational understanding) between the WE and the WE-FL condition, and students' *performance on fluency-building problems* between the FL and the WE-FL condition. Specifically, we compared performance on those tutor problems that were *the same* across these pairs of conditions. To compare the WE and WE-FL conditions, we used errors students made on problems P5 and P6 (see the blue-underlined problems in Table1). To compare the FL and WE-FL conditions, we used errors students made on problems P7 and P8 (see the green-italic problems in Table1). We expect that, if representational understanding facilitates the acquisition of representational fluency, students in the WE-FL condition will make fewer errors on fluency-building problems than students in the FL condition. If representational fluency facilitates the acquisition of representational understanding, we expect the WE-FL condition to make fewer errors on worked-examples problems than students in the WE condition.

Table 2. Means and standard deviation (in parentheses) on pretests and posttests per condition.

Condition	Pretest	Immediate posttest	Delayed posttest
WE	.36 (.22)	.43 (.20)	.49 (.26)
FL	.31 (.21)	.37 (.22)	.44 (.24)
WE-FL	.39 (.21)	.52 (.24)	.58 (.26)

A first step in this analysis was to use the tutor log data to identify measures of errors that students made on these problems. Rather than using the overall error rate, we applied the knowledge component model [17] that underlies the problem structure of the Fractions Tutor to categorize the errors students made while working on the tutor problems. Doing so allows for a much more fine-grained analysis of students' errors than the overall error rate does. The knowledge component model describes a meaningful set of steps within a tutor problem which provide practice opportunities for practicing a "unit" of knowledge. For example, every time a student is asked to enter the numerator of a fraction, he/she has the opportunity to practice knowledge about what the numerator of a fraction is. Worked-example problems and fluency-building problems cover a different set of knowledge components, but the same knowledge components occur repeatedly across different worked example problems and fluency-building problems, respectively. Altogether, the knowledge component model led to 12 types of errors that students could make on worked-example problems, and 11 types of errors that students could make on fluency-building problems.

Next, we had to narrow the number of error categories to include in the causal path analysis model. We included only those error types which (1) were significant predictors of students' posttest performance, while controlling for pretest performance, and (2) significantly differed between conditions. To determine whether an error type was a significant predictor of students' immediate posttest performance, we conducted linear regression analyses with posttest performance as the dependent variable, and pretest performance and number of error type as predictors.

To determine whether error types differed significantly between conditions, we conducted Chi-square tests with number of error type as dependent variable and condition as independent variable (i.e., WE vs. WE-FL for error types that students could make on worked-example problems, and FL vs. WE-FL for error types that students could make on fluency-building problems). For both analyses, we adjusted for multiple comparisons using the Bonferroni correction. On worked-example problems, six error types differed significantly between conditions, but only two error types were significant predictors of posttest performance (both of them passed both the Chi-square test and the regression test). On fluency-building problems, eight error types differed significantly between conditions, and four were significant predictors of posttest performance (three of them passed both the Chi-square test and the regression test). Table 3 provides an overview of the error types we selected for further analyses.

Table 3. Selected error types and number of error-types per condition.

Error type	Description	# in WE	# in FL	# in WE-FL
place1Error	Locating 1 on the number line given a dot on the number line and the fraction it shows	150	n/a	222
SE-Error	Self-explanation error, response to reflection questions in drop-down menu format	1320	n/a	1629
equivalenceError	Finding equivalent fraction representations	n/a	2899	2157
improperMixedError	Finding representations of improper fractions	n/a	1380	1608
NameCircleMixedError	Finding circle representations that show the same fraction as a number line or a rectangle	n/a	355	126

5. PATH ANALYSIS MODELING

In order to investigate whether and how error types mediate the effect of condition, we first specified, estimated, and tested two path analytical structural equation models [19-20] – one which compared the WE and WE-FL conditions using error types made on the worked-example problems as mediators, and one which compared the FL and WE-FL conditions using error types made on the fluency-building problems as mediators. Structural equation models provide a unified framework within which to test mediation hypotheses, to estimate total effects, and also to separate direct from indirect effects. The models that represented our hypotheses in both experiments were decisively rejected by the data, and in such a case it is not appropriate to use the model to test mediation hypotheses or estimate effects. Our strategy was to use the Tetrad IV program[1] to search for alternative models that

[1] Tetrad, freely available at www.phil.cmu.edu/projects/tetrad, contains a causal model simulator, estimator, and over 20 model search algorithms, many of which are described and proved asymptotically reliable in [23] Spirtes, P., Glymour, C. and Scheines, R. *Causation, Prediction, and Search.* MIT Press, 2000.

are both theoretically plausible and consistent with the data. In this section, we describe the path analytic models that represent our hypotheses, describe the search algorithms we use to find for alternative models, and briefly summarize the results of our search.

5.1 Modeling our Hypotheses

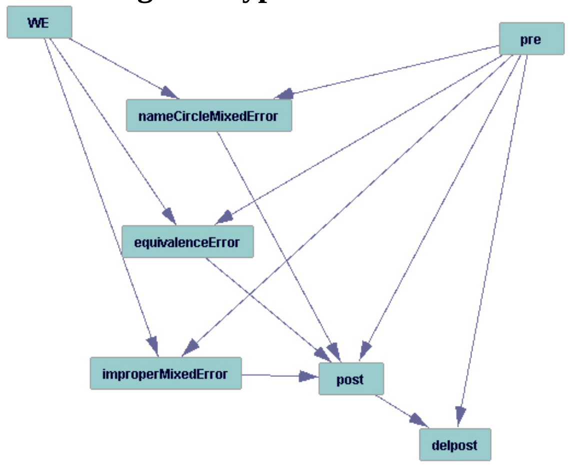


Figure 2. Path model for understanding-first hypothesis.

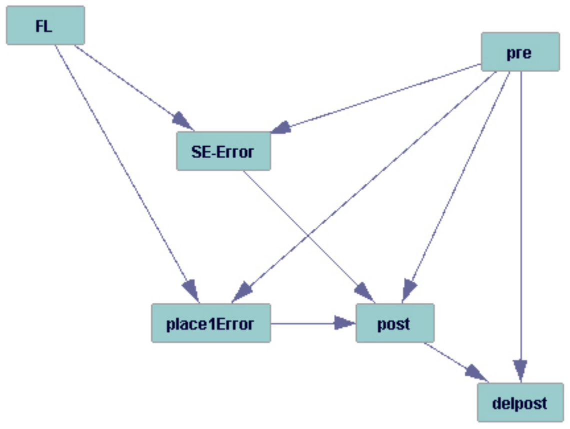


Figure 3. Path model for fluency-first hypothesis.

Our model hypotheses correspond to the understanding-first hypothesis and the fluency-first hypothesis described above. The understanding-first hypothesis predicts that support for representational understanding enhances students' ability to benefit from fluency-building problems by equipping students with the knowledge they need to attend to relevant features of the graphical representations while developing representational fluency. Therefore, students who receive support for representational understanding should make fewer errors on fluency-building problems compared to students who do not receive support for representational understanding. Therefore, the understanding-first hypothesis predicts that support for representational understanding increases learning by reducing the number of errors made on fluency-building problems. Figure 3[2] depicts the model we specified to test the understanding-first hypothesis. Each node in the path model refers to a variable in the data set: *WE* = whether or not students receive worked-example support for representational understanding (i.e., whether they are in the FL vs. in the FL-WE condition), *nameCircleMixedError*, *equivalenceError*, and *improperMixedError* being the errors students could make on fluency-building problems (see Table 3), *pre* = performance on the pretest, *post* = performance on the immediate posttest, *delpost* = performance on the delayed posttest.

The fluency-first hypothesis predicts that support for representational fluency enhances students' ability to benefit from support for representational understanding because representational fluency frees up the cognitive resources that students can invest in sense-making processes that lead to representational understanding. Therefore, students who receive support for representational fluency should make fewer errors on worked-example problems, compared to students who do not receive support for representational fluency. Therefore, the fluency-first hypothesis predicts that support for representational fluency increases learning by reducing the number or errors made on worked-example problems. Figure 4 depicts the model that we specified to test the fluency-first hypothesis. Each node in the path model refers to a variable in the data set: *FL* = whether or not students receive support for representational fluency (i.e., whether they are in the WE vs. in the FL-WE condition), *SE-Error* and *place1Error* being the errors students could make on worked-example problems (see Table 3), *pre* = performance on the pretest, *post* = performance on the immediate posttest, *delpost* = performance on the delayed posttest.

Using normal theory maximum likelihood to estimate the parameters of these models, we find that in each case the deviation between the estimated and the observed covariance matrix is too large to be explained by chance (for the model for the understanding-first hypothesis in Figure 3: $\chi^2 = 30.88$, df = 9, $p < .0001$[3], and for the model for the fluency-first hypothesis in Figure 4: $\chi^2 = 49.14$, df = 6, p < .0001), thus the models do not fit the data and the parameter estimates cannot be trusted[4].

5.2 Model Search

To search for alternatives, we used the GES algorithm in Tetrad IV along with background knowledge constraining the space of models searched [19] to those that are theoretically tenable and compatible with our experimental design. In particular, we assume that our intervention variables are exogenous, that our intervention variables are causally independent, that the pretest is exogen-

[2] In path models of this type, also called "causal graphs" [22] Ibid., each arrow, or directed edge, represents a direct causal relationship relative to the other variables in the model. For example, in Figure 3 the condition is a direct cause of the mediator variables, but only affect the posttest indirectly through these mediators.

[3] The usual logic of hypothesis testing is inverted in path analysis. The p-value reflects the probability of seeing as much or more deviation between the covariance matrix implied by the estimated model and the observed covariance matrix, conditional on the null hypothesis that the model that we estimated was the true model. Thus, a low p-value means the *model* can be rejected, and a high p-value means it cannot. The conventional threshold is .05, but like other alpha values, this is somewhat arbitrary. The p-value should be higher at low sample sizes and lowered as the sample size increases, but the rate is a function of several factors, and generally unknown.

[4] We also tested variations of these models in which we added direct paths from the condition variables to the post-test and delayed post-test. These variants are also clearly rejected by our data.

ous and causally independent of intervention, that the mediators are prior to the immediate posttest and to the delayed posttest, and that the immediate posttest is prior to the delayed posttest. Even under these constraints, there are at least 2^{25} (over 33 million) distinct path models for the understanding-first hypothesis, and 2^{25} (over 33 million) for the fluency-first hypothesis.

The qualitative causal structure of each linear structural equation model can be represented by a Directed Acyclic Graph (DAG). If two DAGs entail the same set of constraints on the observed covariance matrix,[5] then we say that they are empirically indistinguishable. If the constraints considered are independence and conditional independence, which exhaust the constraints entailed by DAGs among multivariate normal varieties, then the equivalence class is called a *pattern* [20-21]. Instead of searching in DAG space, the GES algorithm achieves efficiency by searching in pattern space. The algorithm is asymptotically reliable,[6] and outputs the *pattern* with the best Bayesian Information Criterion (BIC) score.[7] The pattern identifies features of the causal structure that are distinguishable from the data and background knowledge, as well as those that are not. The algorithm's limits are primarily in its background assumptions involving the non-existence of unmeasured common causes and the parametric assumption that causal dependencies can be modeled with linear functions.

5.3 Results

Figure 5 shows a model found by GES for the understanding-first hypothesis, with coefficient estimates included. The model fits the data reasonably well[8] ($\chi^2 = 16.10$, df = 6, $p = .013$). Students with higher pretest scores make fewer nameCircleMixedErrors, and they perform better on the immediate and the delayed posttest. Receiving worked-example support for representational understanding (i.e., being in the WE-FL condition and not in the FL condition) increases nameCircleMixedErrors, which in turn decreases performance on the immediate posttest. In other words, nameCircleMixedErrors mediate a negative effect of worked examples on students' learning. Receiving worked-example support for representational understanding also reduces equivalenceErrors and improperMixedErrors. Since making more improperMixedErrors leads to worse performance on the immediate and the delayed posttests, equivalenceErrors and improperMixedErrors mediate the positive effect of the worked-example support on students' learning. Support for representational understanding through worked examples does not have a direct impact on students' posttest performance. The overall positive effect of worked examples on students' learning through equivalenceErrors and improperMixedErrors is larger than the negative effect through nameCircleMixedErrors. (See Table 3 for a description of the errors.)

[5] An example of a testable constraint is a vanishing partial correlation, e.g., $\rho_{XY.Z} = 0$.

[6] Provided the generating model satisfies the parametric assumptions of the algorithm, the probability that the output equivalence class contains the generating model converges to 1 in the limit as the data grows without bound. In simulation studies, the algorithm is quite accurate on small to moderate samples.

[7] All the DAGs represented by a pattern will have the same BIC score, so a pattern's BIC score is computed by taking an arbitrary DAG in its class and computing its BIC score.

[8] The usual logic of hypothesis testing is inverted in path analysis: a *low* p-value means the model can be rejected.

Figure 6 shows a model found by GES for the fluency-first hypothesis. The model fits the data well ($\chi^2 = 8.32$, df = 5, $p = .14$). Students with higher pretest scores make fewer SE-Errors and perform better on both posttests. Having fluency-building support (i.e., being in the WE-FL condition as opposed to being in the WE condition) increases SE-Errors, which reduces performance on the immediate and the delayed posttest. In other words, SE-Errors mediate a negative effect of fluency-building support. There are no further mediations of having fluency-building support, but there is a direct positive effect of fluency-building support on students' performance on the immediate posttest. (See Table 3 for a description of the errors.)

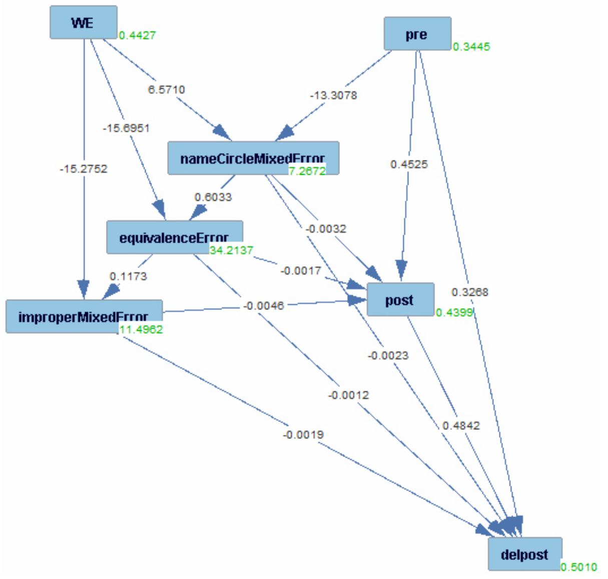


Figure 4. The model found by GES for the understanding-first hypothesis, with parameter estimates included.

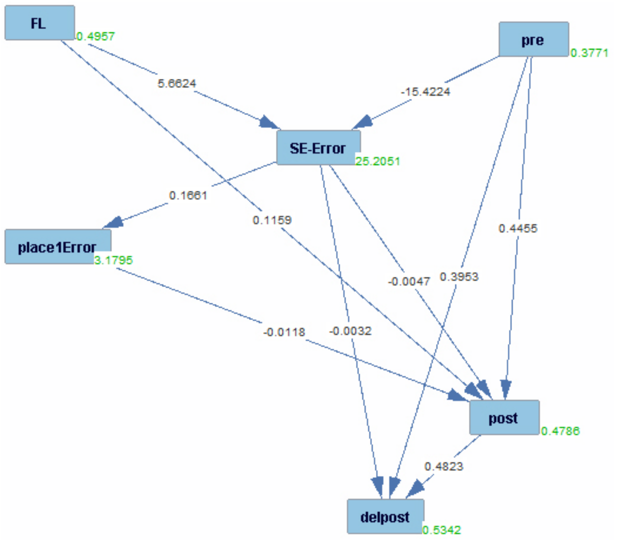


Figure 5. The model found by GES for the fluency-first hypothesis, with parameter estimates included.

6. DISCUSSION

Taken together, results from the causal path analysis models support the understanding-first hypothesis but not the fluency-first hypothesis: receiving worked-example support for representational understanding helps students learn from fluency-building problems. The model in Figure 5 demonstrates that, although students who receive worked-example support make more nameCircleMixedErrors, they make fewer equivalenceErrors and improperMixedErrors. NameCircleMixedErrors are possible early in the Fractions Tutor curriculum, whereas equivalenceErrors and improperMixedErrors occur later in the Fractions Tutor curriculum. The analysis therefore suggests that support for representational understanding reduces errors later during the learning phase, which leads to better overall learning. This finding is particularly interesting when we recall that we only compare the errors students make on fluency-building problems P7 and P8 (see Table 1). For the FL condition, problems P5 and P6 are also fluency-building problems, whereas for the WE-FL condition, problems P5 and P6 are worked-examples problems. That is, students in the FL condition receive more practice on fluency-building problems, which should *increase* their performance on fluency-building problems. Based on practice effects, we would thus expect that students in the FL condition would *outperform* students in the WE-FL condition on problems P7 and P8 (e.g., P7 is the first time the WE-FL condition encounters a fluency-building problem, but the third time the FL condition encounters a fluency-building problem, for the given topic). However, we find the opposite for errors that occur later in the curriculum: worked-example support for representational understanding leads to better performance on fluency-building problems, even compared to students who received more practice on the same types of fluency-building problems. Since higher performance on these problems (i.e., fewer equivalenceErrors and fewer improperMixedErrors) leads to better performance on the immediate posttest, while controlling for pretest, it seems that support for representational understanding prepares students to learn better from subsequent fluency-building problems.

The model in Figure 6 does not provide support for the fluency-first hypothesis. We do not find evidence that fluency-building support helps students benefit from support for representational understanding. Although we find a direct positive effect of fluency-building support on students' learning, the mediation effect shown in Figure 6 is evidence that receiving fluency-building support comes at the cost of lower performance on worked-examples problems: students tend to make more SE-Errors and more place1Errors. This finding is somewhat expected. Students in the WE condition work on twice as many worked-examples problems than the WE-FL condition, so they receive more practice on the worked-examples problems compared to the WE-FL condition (see Table 1). As students in the WE-FL condition have less practice on worked-examples problems, they are expected to perform somewhat worse on those problems – and that is what the model in Figure 6 confirms. Yet, since we do not find evidence that receiving fluency-building support also benefits students' learning from worked-example support for representational understanding, our results do not support the fluency-first hypothesis.

Our findings from path analysis modeling demonstrate the importance of model search. None of our initial hypothesis models fit the data, but there are thousands of plausible alternatives. Further, estimating path parameters with a model that does not fit the data is scientifically unreliable. Parameter estimates, and the statistical inferences we make about them with standard errors etc., are all conditional on the model specified being true everywhere except the particular parameter under test.

Even if our initial hypotheses had fit the data well, it would have been important to know whether there were alternatives that explained the same data. The GES algorithm implemented in Tetrad IV enabled us to find plausible models that fit the data well. The models we found in Figures 5 and 6 allow us to estimate and test path parameters free from the worry that the model within which the parameters are estimated is almost surely mis-specified, as is the case for the models in Figures 3 and 4.

Several caveats need to be emphasized, lest we give the false impression that we think we have "proved" the causal relationships that appear in the path diagrams shown in Figures 5 and 6. First, the GES algorithm assumes that there are no unmeasured confounders (hidden common causes), an assumption that is almost certainly false in this and in almost any social scientific case, but one that is routinely employed in most observational studies.[9] In future work, we will apply algorithms (e.g., FCI) that do not make this assumption, and see whether our conclusions are robust against this assumption. Second, although we did include intervention interaction in our model search and did test for interactions between pretest and mediators, by no means were our tests exhaustive, and by no means can we rely on the assumption that the true relations between the variables we modeled are linear, as the search algorithms assume. The assumption of linear relationships is reasonable but not infallible. Third, we have a sample of 190 students, and although that is sizable compared to many Cognitive Tutor studies, model search reliability goes up with sample size, but down with model complexity and number of variables, and is impossible to put confidence bounds on finite samples [23].

7. CONCLUSIONS AND FUTURE WORK

Our findings provide important insights into the nature of the interaction between students' acquisition representational understanding and representational fluency. Our analysis supports the notion that the acquisition of representational understanding enhances students' ability to benefit from instructional support for representational fluency, more so than the other way around. Therefore, our findings suggest that instruction should provide support for representational understanding *before* providing support for representational fluency.

Although our analyses provide support for the understanding-first hypothesis, but not for the fluency-first hypothesis, both remain valid hypotheses. One important caveat of the analyses presented here is that, within each curricular topic of the Fractions Tutor, all students in the WE-FL condition received support for representational understanding before support for representational fluency (i.e., there was no FL-WE condition). We therefore cannot draw definite conclusions about the relative effectiveness of providing support for representational understanding before support for representational fluency (WE-FL) and providing support for representational fluency before support for representational understanding (FL-WE). Our findings based on causal path analysis modeling merely suggest that the WE-FL condition would lead to better learning than a FL-WE condition. This notion remains to be tested empirically.

Future research should also investigate whether our findings are specific to the domain of fractions learning, and to the acquisition of representational understanding and representational fluency in making connections between multiple graphical representations. Graphical representations are universally used as instructional tools to emphasize and illustrate conceptually relevant aspects of the domain content. Furthermore, in any given domain, students

[9] Although our data are from a study in which we intervened on intervention, we did not directly intervene on our mediator or outcome variables. Thus these parts of our model are subject to the same assumptions as a non-experimental study.

need to develop representational fluency in using graphical representations to solve problems, and they need to effortlessly translate between different kinds of representations. But representational understanding and representational fluency are not limited to learning with graphical representations: representational understanding and representational fluency also play a role in using symbolic and textual representations. For example, should students acquire representational fluency in applying a formula to solve physics problems before understanding the conceptual aspects the formula describes, or should they first conceptually understand the phenomenon of interest and then learn to apply a formula to solve problems related to that phenomenon? This is a crucial question for instructional design and one that remains open. While the analysis presented in this paper takes an important step towards answering this question by providing novel insights into how representational understanding and representational fluency interact, more research is needed to investigate implications and applications related to the question of how best to support students to develop expertise with representational understanding and representational fluency.

The use of search algorithms over plausible causal path analysis models is a promising method to analyze the effects of instructional interventions on, because we can get insights into *how* an intervention affects problem-solving behaviors, and *how* these effects account for the advantage of one intervention over the other. Basing our analysis on cognitive task analysis and knowledge component modeling, we make use of common techniques in the analysis of tutor log data [23]. The results from our causal path analysis not only provide insights into the nature of the interaction of the experimental study, but also raise new hypotheses that can be empirically tested in future research. Thereby, our findings illustrate that causal path analysis modeling is a useful technique to augment regular tutor log data analysis.

8. ACKNOWLEDGEMENTS

We thank the NSF REESE-21851-1-1121307, the PSLC, funded by NSF award number SBE-0354420, Ken Koedinger, the Datashop team, and the students and teachers.

9. REFERENCES

[1] Koedinger, K.R., Corbett, A.T. Perfetti, C.: Knowledge-Learning-Instruction Framework: Bridging the Science-Practice Chasm to Enhance Robust Student Learning. Cognitive Science, 36, 757–798 (2012).

[2] NMAP: Foundations for Success: Report of the National Mathematics Advisory Board Panel. U.S. Government Printing Office (2008).

[3] Cramer, K. Using models to build an understanding of functions. *Mathematics teaching in the middle school*, 6, 310-318 (2001).

[4] Charalambous, C.Y., Pitta-Pantazi, D.: Drawing on a Theoretical Model to Study Students' Understandings of Fractions. Educational Studies in Mathematics, 64, 293-316 (2007).

[5] Siegler, R. S., Thompson, C. A. and Schneider, M. An integrated theory of whole number and fractions development. *Cognitive Psychology*, 62, 273-296 (2011).

[6] Rau, M.A., Aleven, V. Rummel, N.: Intelligent tutoring systems with multiple representations and self-explanation prompts support learning of fractions. In: Dimitrova, V. et al. (eds) *Proceedings of the 2009 conference on Artificial Intelligence in Education*. IOS Press, Amsterdam, The Netherlands. pp. 441-448 (2009).

[7] Bodemer, D., Ploetzner, R., Feuerlein, I., Spada, H.: The Active Integration of Information during Learning with Dynamic and Interactive Visualisations. Learning and Instruction, 14, 325-341 (2004).

[8] Ainsworth, S.: DeFT: A conceptual framework for considering learning with multiple representations. Learning and Instruction, 16, 183-198 (2006).

[9] de Jong, T., Ainsworth, S. E., Dobson, M., Van der Meij, J., Levonen, J., Reimann, P., Simerly, C. R., Van Someren, M. W., Spada, H. and Swaak, J. *Acquiring knowledge in science and mathematics: The use of multiple representations in technology-based learning environments*. Oxford (1998).

[10] Bodemer, D. and Faust, U. External and mental referencing of multiple representations. *Computers in Human Behavior*, 22, 27-42 (2006).

[11] Schwonke, R., Berthold, K. and Renkl, A. How multiple external representations are used and how they can be made more useful. *Applied Cognitive Psychology*, 23, 1227-1243 (2009).

[12] Rau, M.A., Rummel, N., Aleven, V., Pacilio, L., Tunc-Pekkan, Z.: How to schedule multiple graphical representations? A classroom experiment with an intelligent tutoring system for fractions. In: Aalst, J.v., et al. (eds) The future of learning: *Proceedings of the 10th ICLS*, ISLS, Sydney, Australia. 64-71 (2012).

[13] Kellman, P. J., Massey, C. M. and Son, J. Y. Perceptual Learning Modules in Mathematics: Enhancing Students' Pattern Recognition, Structure Extraction, and Fluency. *Topics in Cognitive Science*, 22009), 285-305.

[14] Rau, M. A., Aleven, V., Rummel, N. and Rohrbach, S. *Sense Making Alone Doesn't Do It: Fluency Matters Too! ITS Support for Robust Learning with Multiple Representations*. Springer Berlin / Heidelberg (2012).

[15] VanLehn, K. The relative effectiveness of human tutoring, intelligent tutoring systems and other tutoring systems. *Educational Psychologist*, 46, 197-221 (2011).

[16] Aleven, V., McLaren, B. M., Sewall, J. and Koedinger, K. R. Example-tracing tutors: A new paradigm for intelligent tutoring systems. *International Journal of Artificial Intelligence in Education*, 19, 105-154 (2008).

[17] Baker, R. S. J. d., Corbett, A. T. and Koedinger, K. R. The difficulty factors approach to the design of lessons in intelligent tutor curricula. *International Journal of Artificial Intelligence in Education*, 17, 341-369 (2007).

[18] Rau, M. A., Aleven, V. and Rummel, N. Blocked versus Interleaved Practice with Multiple Representations in an Intelligent Tutoring System for Fractions, In: Yacef, K. et al. (eds), *Proceedings of the 5th Intl Conference of ITS*, Springer Berlin / Heidelberg (2010).

[19] Chickering, D. M. Optimal Structure Identification with Greedy Search. *Journal of Machine Learning Research*, 507-554 (2002).

[20] Pearl, J. *Causality: Models, Reasoning, and Interference*. Cambridge University Press (2000).

[21] Spirtes, P., Glymour, C. and Scheines, R. *Causation, Prediction, and Search*. MIT Press (2000).

[22] Robins, J., Scheines, R., Spirtes, P. and Wasserman, L.: Uniform Consistency in Causal Inference. *Biometrika*, 90, 491 – 515 (2003).

[23] Koedinger, K. R. et al.: A data repository for the EDM community: The PSLC Data-Shop, In: C. Romero (ed.), *Handbook of educational data mining* (pp. 10-12). Boca Raton, FL: CRC Press (2010).

Predicting Standardized Test Scores from Cognitive Tutor Interactions

Steve Ritter, Ambarish Joshi, Stephen E. Fancsali, Tristan Nixon
Carnegie Learning, Inc.
437 Grant Street, Suite 918
Pittsburgh, PA 15219
1-412-690-2442
{sritter, ajoshi, sfancsali, tnixon}@carnegielearning.com

ABSTRACT
Cognitive Tutors are primarily developed as instructional systems, with the goal of helping students learn. However, the systems are inherently also data collection and assessment systems. In this paper, we analyze data from over 3,000 students in a school district using Carnegie Learning's Middle School Mathematics tutors and model performance on standardized tests. Combining a standardized pretest score with interaction data from Cognitive Tutor predicts outcomes of standardized tests better than the pretest alone. In addition, a model built using only 7th grade data and a single standardized test outcome (Virginia's SOL) generalizes to additional grade levels (6 and 8) and standardized test outcomes (NWEA's MAP).

Keywords
Cognitive Tutors, Assessment, Mathematics.

1. INTRODUCTION
Cognitive Tutors are primarily developed as instructional systems, with a focus on improving student learning. While the systems continually assess student knowledge with respect to a set of underlying knowledge components [6], the standard for effectiveness of an educational system is usually taken to be the ability of that system to produce improved performance on an external measure, typically a standardized test.

Carnegie Learning's Cognitive Tutors for mathematics have done well on such measures [13, 15, 18] but, in these studies, the tutoring system has been, essentially, treated as a black box. We know that, as a whole, students using a curriculum involving the Cognitive Tutor outperformed students using a different form of instruction on standardized tests, but we don't know what specific aspects of tutor use were associated with improved performance. An understanding of the process variables (time, errors, hint usage and other factors) that are correlated with learning can provide us with insight into the specific student activities that seem to lead to learning. Another perspective on the Tutor is that, if we are able to strongly correlate Cognitive Tutor data with standardized test data, then the Cognitive Tutor itself may be considered an assessment, which is validated with respect to the external standardized test. In addition, to the extent that we can identify process variables that predict external test scores, we can provide guidance to teachers as to expectations for their students on the state examinations.

In most cases, Carnegie Learning does not have access to student-level outcome data on standardized tests. For the study reported here, we partnered with a school district in Eastern Virginia. The district provided Carnegie Learning with student data for all 3224 middle school students who used Cognitive Tutor in the district during the 2011/12 school year. The data included student demographics and outcomes on the Virginia Standards of Learning (SOL) assessment and NWEA's Measures of Academic Progress (MAP) assessment. The district also provided MAP scores from the Fall of 2011, which provides information about student abilities prior to their encounter with the Cognitive Tutor. This dataset allows us to explore which particular behaviors within Cognitive Tutor are associated with improved outcomes and to test whether we are better able to predict outcomes knowing student behaviors within the tutor than we could be able to predict from demographic and prior knowledge variables.

While other analyses [5, 14] have modeled outcomes based on tutor process data, the current analysis goes beyond previous efforts by building a model based on a single grade level and outcome and then applying the model to two outcome measures across three grade levels.

1.1 Virginia's Standards of Learning (SOL) Assessment
The Virginia Department of Education's Standards of Learning (SOL) provide minimum expectations for student knowledge for several subjects at the end of each grade level or after specific courses [21]. Students take standardized tests based on the mathematics SOL annually for grades 3 through 8 as well as after taking particular courses after grade 8 (e.g., after taking Algebra I). The SOL exam includes multiple choice items as well as "technology enhanced" items that may include drag-and-drop, fill-in-the-blank, graphing and "hot spot" identification in a picture.

With the advent of the No Child Left Behind Act, there is great interest in developing predictive models of student performance on high-stakes tests like the SOL mathematics assessment to identify students that may need remediation. Cox [8], for example, develops regression models to predict grade 5 SOL mathematics assessment scores using student scores on Mathematics Curriculum Based Measurement (M-CBM) benchmark assessments, which are shorter, formative assessments that track student progress over time. This study reports that M-CBM alone can account for roughly 40% to 60% of the variance in SOL assessment scores across three Virginia school districts.

1.2 NWEA's Measures of Academic Progress® (MAP) Assessment and RIT score
NWEA's MAP is a computer-based adaptive assessment. MAP assessments deliver student scores on the Rasch Unit (RIT) scale, an equal interval measurement, intended to provide scores comparable across grade levels and to help track student progress from year to year [20]. For the district in our study, the MAP assessment was administered in both the fall and spring semesters. NWEA recently published data correlating MAP performance

with Virginia SOL assessment performance for over 5,000 students. For 2012 data they report correlations (as Pearson's r) of 0.759, 0.797, and 0.75 between MAP scores and SOL mathematics assessment scores for grades 6, 7, and 8, respectively [19]. These figures are comparable to the correlations we report here between RIT and SOL in grades 6 and 7, but the correlation for grade 8 was lower (r of 0.755, 0.704 and 0.551 for grades 6, 7, 8, respectively).

1.3 Cognitive Tutor

The Cognitive Tutor presents a software curriculum as a sequence of "units," which are major topics of instruction. Units are divided into one or more sections, which represent subtopics. Each section is associated with one or more skills (or knowledge components), which are the target of the mastery learning. Each section has a large pool of available problems (from 10s to 1000s, depending on the section), which are chosen to remediate students on each of the target skills. The Tutor considers a student to have mastered a section when Bayesian Knowledge Tracing [7] judges that there is at least a 95% probability that the student knows the skill. When a student masters all of the skills in a section, the system allows the student to proceed to the next section (or unit, if this is the final section in a unit). When the student completes a problem without mastering all of the skills for that section, the Tutor picks a new problem for the student, focusing on the skills that still need to be mastered. Students require different numbers of problems to master all of their skills.

Although the intent of the system is for students to progress through topics as they master them, we recognize that there will always be some students who are not learning from the tutor's instruction (or who are learning too slowly). For this reason, each section specifies a maximum number of problems. If the student reaches the maximum without mastering all skills in the section, the student is advanced to the next section without mastery. Teachers are notified of this advancement in reports. Thus, in our models, we make a distinction between sections (or skills) encountered and sections (or skills) mastered. Teachers may also manually move a student to another point in the curriculum, which would also result in an encounter with a section (or skill) without mastery.

Problems within the Cognitive Tutor are completed in steps, each of which represents a discrete entry into the system (such as filling in a text field). The tutor evaluates each step and provides immediate feedback. Students can also ask for help on each step. The number of steps required to complete a problem depends on the complexity of the problem and the particular strategy that the student uses to complete the problem. Some problems may be completed in 5-10 steps but others may require 30 or more. Within a section, problems typically require the same (or very similar) number of steps. Because the variability of problems across sections affects the potential for making errors and asking for hints (as well as the expected time to complete the problem), we normalize hints, errors and time within each section and then calculate an average across sections, for each student. This normalization also helps account for the fact that different students encountered different sections (depending on grade level and custom sequences of units), so we should have different expectations about time, hints and errors per problem within the sections they did encounter.

The software includes various instructional activities and supports in addition to the Cognitive Tutor. These include basic lesson text, multiple-choice tests, step-by-step examples and other components. None of these other activities directly affect our assessment of mastery. For this reason, we distinguish between "problem time" (the time that students spend within Cognitive Tutor problems) and the total time that students spend logged in to the tutor.

2. DISTRICT IMPLEMENTATION

The students in this study used Cognitive Tutor software developed for middle school mathematics (grades 6, 7 and 8) in 12 schools in the district. The software was used by 3224 students: 1060 in sixth grade; 1354 in seventh grade and 810 in eighth grade.

Carnegie Learning delivers software sequences aligned to educational standards for these grades. The sixth grade sequence contains 45 units and 131 sections; the seventh grade sequence contains 34 units and 92 sections and the eighth grade sequence contains 37 units and 88 sections. The school district also created software sequences targeted towards students who were performing below grade level, as part of a Response to Intervention (RTI) implementation [10].

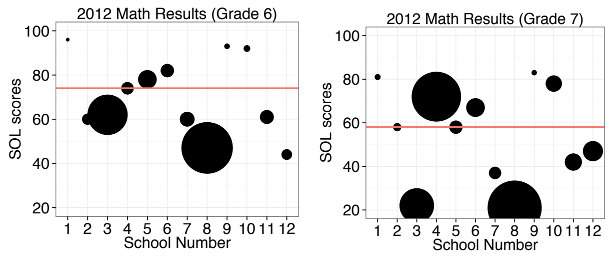
Carnegie Learning recommends that, when used as a basal school curriculum, students use the Cognitive Tutor as part of their math class two days/week. Assuming 45-minute classes and a 180-day school year, this would result in approximately 54 hours of use. Due to scheduling issues, absenteeism and other factors, it is common for districts to average only 25 hours, however. RTI implementations may involve more intensive (5 days/week) practice on prerequisite skills, typically completed in an RTI class which takes place in parallel with work in the basal class. Thus, students in an RTI sequence may be asked to work on the Tutor twice as much (or more) than those who are not in an RTI sequence. On the other hand, if students in the RTI class are able to demonstrate mastery of the target material, they are removed from that class, so the RTI class does not necessarily last all year.

The Tutor is available to students through a browser, so they can go home (or elsewhere) and continue work that they began at school. However, many students do not use the Tutor outside of class and so, for most students, the amount of time that they spend with the tutor is dictated by the frequency with which their teacher tells them to use the software.

Our analysis does not distinguish between students who used the Tutor in an RTI capacity, as a basal curriculum or in some other capacity.

Across the schools and grade levels in our data set, usage varied widely, from median of 1.05 hours in grade 8 in one school to a median of 29.86 hours in grade 7 in a different school.

Figure 1 provides a schematic for understanding overall performance of students at the schools involved in the study. There are three graphs, representing the three grade levels. Each school is represented by a dot, with the size of the dot proportional to the number of students in that school and grade level. The vertical position of the dot represents the school's overall 2012 SOL score. The horizontal line represents the state average for the grade level. The figure shows that students in the district reflect a range of abilities relative to the state, with students somewhat underperforming the state mean in all grades.



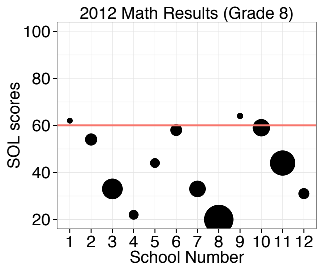


Figure 1: School-level SOL math results for the schools in the study. The size of the dot represents the number of students in the study. The horizontal line represents the state average.

3. ANALYSIS APPROACH

Our primary purpose in this work is to build a general model that can be used to predict outcomes from standardized tests based on usage and behaviors within Cognitive Tutor. To this end, we build a model based on a subset of the students and a single outcome variable and then test the model on the rest of the students and an additional outcome variable. Since the seventh grade cohort was the largest in our data, we chose to build the model on the seventh graders. We chose SOL as the outcome measure in building our model, because it is the most important outcome to the schools using Cognitive Tutor.

Since we expected use of Cognitive Tutor to influence outcomes only if there was some substantial use of Cognitive Tutor over the school year, we excluded students who, in total, spent less than 5 hours using Cognitive Tutor. This reduced the number of students considered by the model from 1354 7th graders (and 3224 students in all grades) to 940 7th graders (and 2018 students overall).

In order to explore the influence of different kinds of information, we constructed 5 models:

- M1 – includes only RIT pretest score as a predictor
- M2 – includes only Cognitive Tutor process variables
- M3 – includes the Cognitive Tutor process variables used in M2, plus student demographics
- M4 – includes RIT pretest score plus student demographics
- M5 – includes M3 variables, plus RIT pretest score

To build M2, we used stepwise regression to identify the Cognitive Tutor process variables, of those considered (see Variable Selection, below), that minimized the Bayesian Information Criterion (BIC). This model considered only 7th grade data and only SOL as an outcome. To build M3, we included those variables selected for M2 and student demographic variables and used stepwise regression (using BIC) to choose the most predictive demographic variables, keeping the process variables fixed. We used 10-fold cross-validation of these models to ensure that we were not overfitting, but the cross-validated models were very similar to the model found by simple stepwise regression, so we only consider the stepwise regression model here.

3.1 Cognitive Tutor Process Variables

Since overall usage of the Tutor varied substantially between students, we reasoned that a good model would take into account variables that represent this overall usage (either as represented by time or by various completion measures). Since most usage of the tutor is controlled by the teacher (or the class that the student is in), variance within a class might be better captured by metrics representing activities taken within a problem.

Many of the variables we considered were highly skewed in the data, and so, following common practice, we applied log transforms to them. For example, Figure 2 (left) shows the distribution of time spent on the tutor (excluding students who spent fewer than 5 hours). Although the median of the distribution is 19.8 hours, there is a long tail of students who spend a much longer time using the tutor. A log transform produces the more normal distribution shown in Figure 2b.

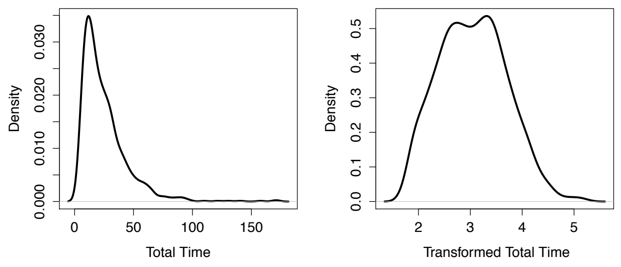


Figure 2: Distribution of total time on the tutor (left graph) and the log-transformed distribution (right graph).

Since we are modeling students that completed different mathematics topics (different units and sections within Cognitive Tutor), we normalized many of these process variables within section. This normalization results in variables that represent the difference between a student's behavior and the average student's behavior, in standard deviation units, so that values are comparable across sections of the curriculum. In cases where we log transformed the data, normalization followed the log transformation. To compute a value for each student, we averaged the normalized (z) scores across all of the sections that the student encountered. For "aggregate" variables, which sum across sections of the curriculum, we normalized with respect to the distribution of data across the full school year. Since we normalize all variables in the model, the magnitude of the coefficients gives a sense of the relative impact of each variable.

Based on some preliminary correlations with outcomes in these data and other datasets, we considered 13 Cognitive Tutor process variables for our model:

Aggregate variables

- **Total_time**: the total amount of time that the student was logged in to the tutor. This variable was log transformed.
- **Total_problem_time**: the amount of time that students spent on the problem-solving activities within the tutor. This differs from total_time by excluding time spent reading lesson content, viewing worked examples and

several other activities. This variable was log transformed.

- **Percent_non_problem_time**: the percentage of time that the student was logged in to the software but did things other than solving problems in Cognitive Tutor. This variable was log transformed.
- **Sections_encountered**: the total number of sections attempted by the student. This variable was log transformed.
- **Skills_encountered**: the total number of skills within the sections that the student encountered. This variable was log transformed.
- **Percent_skills_mastered**: the percentage of skills encountered that reached mastery. This variable was subtracted from 1 and log transformed.
- **Percent_sections_mastered**: the percentage of sections encountered that were mastered. This variable was subtracted from 1 and log transformed.
- **Sections_mastered_per_hour**: This is the average number of sections mastered by the student for each hour of time spent on problems. We consider this variable to be an indicator of the efficiency with which the student uses the system. Efficiency may be affected by the student's prior knowledge and conscientiousness but also by the extent to which teachers are effective in assisting students when they get stuck. This variable was log transformed.

Section-normalized variables

- **Time_per_problem**: the average amount of time spent on each problem. This variable was log transformed and normalized
- **Hints_per_problem**: the average of the number of hints requested in each problem. This variable was log transformed and normalized.
- **Errors_per_problem**: the average of the number of errors committed in each problem. This variable was log transformed and normalized.
- **Assistance_per_problem**: the average of the sum of the number of hints and the number of errors in each problem. This variable is an indicator of the extent to which students struggle to complete problems and was log transformed and normalized.
- **Problems_per_section**: the number of problems required to master the skills in the section (or, for non-mastered sections, the maximum number of problems in the section). This variable was log transformed and normalized.

3.2 Demographic Variables

In addition to process variables, we considered the following student demographic variables (note that statistics are calculated based on the 2018 students with more than 5 hours usage):

- **Sex**: male or female. In our sample, there were 1027 boys and 991 girls.
- **Age:** in days (as of 06/01/12). In 6th grade,, the mean was 4493 with standard deviation 163. In 7th grade, mean was 4868 with standard deviation of 172. In 8th grade, the mean was 5269, with standard deviation of 184.
- **Lunch_status**: this is a common proxy for socio-economic status. We coded this as a binary variable indicating whether students were eligible for free or reduced lunch prices. 72.5% of students were in this category.
- **Limited_English_Proficiency**: A binary variable coding whether the student was identified as having a poor mastery of English. 6.9% of students in our sample were identified as being in this category.
- **Race**: The school provided coding in six categories, shown here:

Description	Students
American Indian	20
Asian	72
Black/African American	1064
White	785
Hawaiian /Pac. Islander	5
Multi-racial	72

- **Hispanic_origin**: A binary variable representing whether the student is of Hispanic origin. 17.1% of students were identified as Hispanic.
- **Special_education_status**: We coded this status as representing four categories: no special status, learning disability, physical disability or other (which included students with multiple disabilities and those that were listed as special ed but not classified). 9.6% of students were identified with a learning disability, 8.4% with physical disability and 1% with other.

4. RESULTS

4.1 Fitted Models

The process variables found in M2 and included in M3 and M4 were *total_problem_time*, *skills_encountered*, *sections_encountered*, *assistance_per_problem* and *sections_mastered_per_hour*. A summary of the standardized model (M2) coefficients is shown in Table 1.

Table 1: Cognitive Tutor process variables and standardized coefficients included in the models predicting SOL.

Variable	Coefficient	p value
assistance_per_problem	-0.351	<2e-16
sections_encountered	0.422	0.004028
sections_mastered_per_hour	0.390	6.71E-10
skills_encountered	-0.456	0.000141
total_problem_time	0.258	0.000502

Three variables (*total_problem_time*, *skills_encountered* and *sections_encountered*) represent aggregate usage of the tutor. *Skills_encountered* is entered with a negative coefficient, perhaps trading off with the (positive) *sections_encountered* and indicating that students benefitted from completing a larger number of sections, particularly if the sections involved a relatively small number of skills. Although it is tempting to interpret sections with small numbers of skills as simpler (or shorter) sections, the number of skills tracked in a section is not always a measure of complexity, particularly in sections that include a number of skills that many students have previously mastered [17].

The model also includes *assistance_per_problem*, and *sections_mastered_per_hour*. Together, these variables reflect students' ability to efficiently work through problems, staying on task, making few errors and not overly relying on hints.

Demographic variables found in M3 and used in M4 and M5 included lunch_status and Age. Student of low socio-economic status (i.e., those that qualify for free and reduced-price lunches indicated by *lunch_status*) perform significantly worse on the SOL, even after accounting for Cognitive Tutor process variables. *Age* is also a factor in the model indicating that, after accounting for other model variables, students who are older tend to underperform their younger classmates. Although significant, the age effect is small. One year increase in age, within grade, results in a score reduction of 8 points. Consider that 8-point difference relative to the larger differences in school-level SOL scores shown in Figure 1.

Table 2: Variables and standardized coefficients for M3 applied to SOL

Variable	Coefficients	p value
assistance_per_problem	-0.340	<2e-16
sections_encountered	0.369	0.011183
sections_mastered_per_hour	0.368	4.04E-09
skills_encountered	-0.403	0.000663
total_problem_time	0.240	0.001043
lunch_status	-0.106	3.30E-05
age	-0.071	0.003737

We were surprised to find that special education status was not a significant predictor. Figure 3 shows why: in these data, student SOL scores do not vary much with respect to special education status.

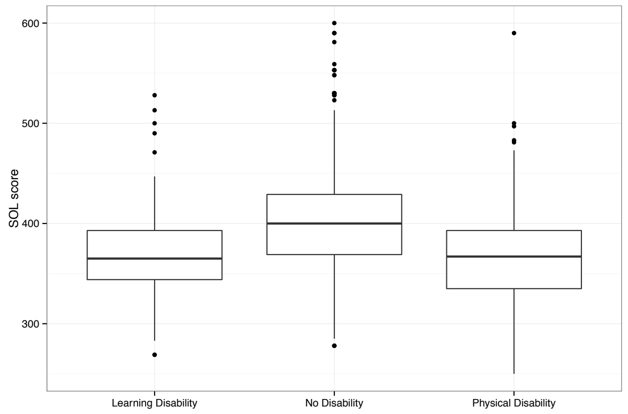


Figure 3: SOL score by special education status.

Table 3 shows a summary of the complete M5. Once RIT is included, *age* and *sections_encountered* are no longer significant predictors, although all other predictors are still significant.

A summary of the model fits is shown in Table 4. It is notable that *RIT_pretest* predicts SOL scores somewhat better than the Cognitive Tutor process model (R^2 of 0.50 for M1 vs. 0.43 for M2) and that adding demographics to either Cognitive Tutor process variables (M3) or RIT alone (M4) increases the predictive validity of the model only slightly. The combination of RIT and Cognitive Tutor process variables (M5) increases the fit of the model substantially, compared to either M3 or M4. This may indicate that the RIT pretest and the Cognitive Tutor process variables are capturing different and complementary aspects of student knowledge, motivation and preparedness.

Despite containing the most variables, M5 is the only model that shows a substantially lower BIC score than M1 (RIT alone), indicating that these variables, in combination, provide substantial explanatory power.

Table 3: Variables and standardized coefficients for M5, as applied to SOL

Variable	Coefficients	p value
assistance_per_problem	-0.134	1.02E-05
sections_encountered	0.188	0.141868
sections_mastered_per_hour	0.272	7.80E-07
skills_encountered	-0.262	0.011882
total_problem_time	0.219	0.000669
lunch_status	-0.070	0.00166
age	-0.028	0.197271
RIT pretest	0.476	<2e-16

Table 4: Summary of fits for models of SOL

Model	Number of variables	BIC	R^2
M1 (RIT)	1	2041.451	0.50
M2 (CT)	5	2181.015	0.43
M3 (CT+Demog)	7	2167.764	0.45
M4 (RIT+Demog)	3	2030.582	0.51
M5 (Full)	8	1928.369	0.57

4.2 Generalizing to other grades

Table 5 shows fit for the models as applied to sixth and eighth grade students' SOL scores (and as applied to the full population of students), using the variables and coefficients found by fitting the seventh grade student data. The model fits the sixth grade data remarkably well, with an R^2 of 0.62, higher even than the fit to the seventh grade data. The fit to eighth grade data is not as strong, with an R^2 of 0.32. This may be due to both the smaller original population in eighth grade and the relatively low usage. Median usage for eighth graders was only 16.3 hours (as opposed to 20.3 hours in sixth grade), and only 438 students (54%) of eighth graders used the tutor for more than five hours.

Table 5: Summary of fits of the model as applied to SOL for held-out students (grades 6 and 8) and to the whole population (grades 6,7,8)

R^2	Grade 6	Grade 8	All grades
M1	0.57	0.30	0.40
M2	0.46	0.18	0.38
M3	0.46	0.18	0.42
M4	0.57	0.30	0.43
M5	0.62	0.32	0.51

Figure 4 demonstrates the fit of M5 to the SOL data for all grade levels.

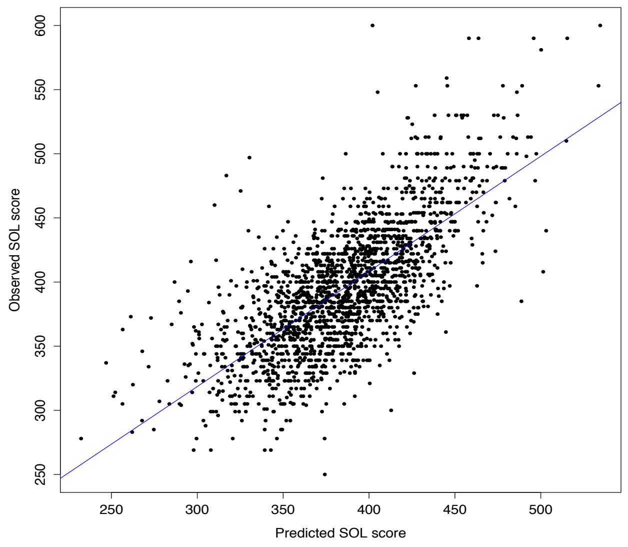


Figure 4: Relationship of predicted SOL using M5 to actual SOL scores for students at all grade levels.

Although the fit is very good (R^2=0.51), the model appears to be slightly underpredicting SOL scores for the best students and slightly overpredicting SOL scores for the worst students.

4.3 Generalizing to the RIT posttest

Our next question was whether the variables we found for modeling SOL would also produce a good model of RIT score. In order to do this, we used the variables from M5 and regressed the model against the RIT posttest score, again fitting the seventh grade students. Table 6 shows the resulting standaradized coefficients.

Note that, with RIT as the outcome variable, *sections_mastered_per_hour*, *total_problem_time* and *age* are no longer significant predictors for the model. It may be that RIT, as an adaptive test, imposes less time pressure on students, since there is no apparent set of questions to be completed in a fixed amount of time.

Table 6: Standardized coefficients and p values for M5, predicting RIT posttest scores

Variable	Coefficients	p value
assistance_per_problem	-0.186	1.68E-15
sections_encountered	0.267	0.006
sections_mastered_per_hour	0.031	0.45747
skills_encountered	-0.206	0.00927
total_problem_time	0.003	0.95771
lunch_status	-0.044	0.0092
age	0.008	0.64494
RIT pretest	0.677	<2e-16

Table 7 shows fits for the five models, as applied to the RIT posttest. As expected, RIT pretest predicts RIT posttest better than the RIT pretest predicts the SOL posttest (R^2 for M1/SOL for 7th grade is 0.50 vs. 0.72 for M1/RIT). Even given this good fit for M1, process variables and demographics significantly improve the model, reducing BIC from 1502 to 1409.

Table 7: R^2 and BIC for models as applied to RIT posttest.

	R^2				BIC
	Grade 6	Grade 7	Grade 8	All	Grade 7
M1	0.67	0.72	0.59	0.68	1502.128
M2	0.46	0.49	0.26	0.41	2085.297
M3	0.48	0.50	0.27	0.40	2077.530
M4	0.68	0.72	0.59	0.68	1501.578
M5	0.71	0.75	0.60	0.71	1408.899

The fit to the full population is illustrated in Figure 5.

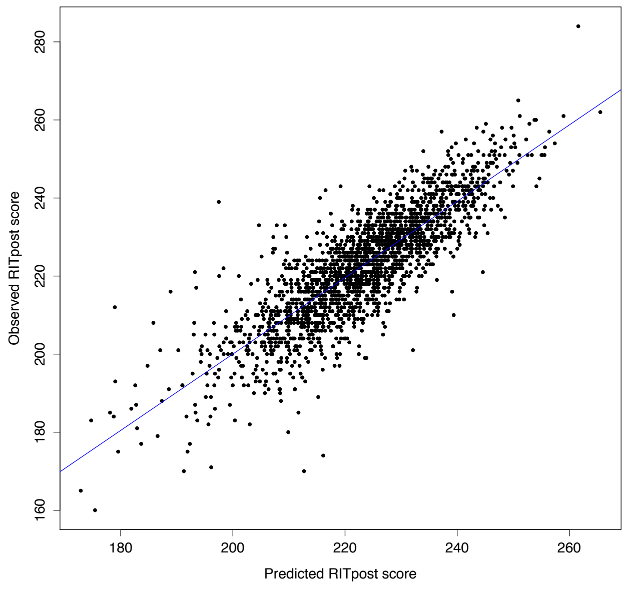


Figure 5: Relationship of predicted RIT posttest scores using M5 to actual RIT posttest scores for students at all grade levels.

To this point, we have been considering the full population to be the population of students who used Cognitive Tutor for at least 5 hours. Figure 6 shows how the model applies to all students who used the Cognitive Tutor for any period of time.

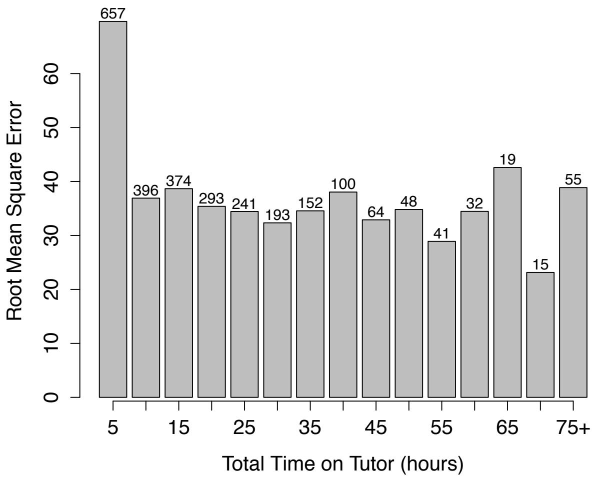


Figure 6: M5 Root Mean Square Error (from predicted SOL outcome), as a function of total time on the Tutor. Numbers above the bars represent the number of students represented by the bar.

Figure 6 demonstrates that the model's ability to predict SOL scores decreases dramatically for students with low Cognitive Tutor usage. This is to be expected: outcomes for students who used the software very infrequently are only lightly influenced by anything they may have learned from the Tutor, and the relatively small amount of data that the Tutor was able to collect from such students represents a noisy estimate of the student's abilities. Our choice of 5 hours as a cutoff point was based on experience with other data sets. Figure 6 validates this cutoff for our data.

5. DISCUSSION

The work presented here provides a good model of how we might use Cognitive Tutor, either with or without additional data, to predict student test outcomes on standardized tests. The model was able to generalize to different student populations, and the variables found for a model to predict SOL provided strong predictions of RIT as well.

Surprisingly to us, demographic factors proved to be relatively unimportant to our models.

Since we were able to improve on the RIT pretest model by adding Cognitive Tutor process variables, our efforts show that such variables provide predictive power beyond that provided by a standardize pretest, even when the pre- and post-test are identical (as in the case with the RIT outcome). A consideration of the types of information that may be contained in Cognitive Tutor data but not in pretest data provide us with guidance on how we might extend this work and improve our model. We will consider 5 broad categories of factor: learning, content, question format, process and motivation.

Learning: The most obvious difference between the RIT pretest and the Cognitive Tutor process variables is that the RIT provide information about students at the beginning of the school year, while Cognitive Tutor data is collected throughout the year. One extension of this work in exploring the role of learning would be to look at how well the model presented here would predict outcomes if we only considered data from the first six months (or three months – or less) of the school year. If such an early prediction model were to work, it could act as an early warning system for teachers and administrators [1].

Content: Although both RIT and Cognitive Tutor are designed to align to the SOL, it is possible that differences in that alignment are responsible for some of the improvement that Cognitive Tutor provides over RIT alone in predicting SOL scores. Feng et al.[9], using the ASSISTment system, built a one-parameter IRT model to predict test outcomes, an approach which allows them to weight different ASSISTment items better with respect to the outcome variable. Our models considered all skills and sections to have equivalent predictive power, but a more sophisticated model could take content alignment into account.

Question format and problem solving: Cognitive Tutor problems involve multiple steps and the kind of strategic decision making that is characteristics of problem solving. Traditional standardized tests are multiple choice or single-step fill-in-the-blank and tend to assess more procedural and less conceptual knowledge. In past research [13], Cognitive Tutor students have shown stronger performance (relative to control) on open-ended responses than on traditional standardized tests. Part of the prediction within SOL may be due to the closer alignment between Cognitive Tutor and the technology-enhanced SOL question types. Most states in the United States (but not Virginia) have adopted the Common Core State Standards. Two consortia, Partnership for Assessment of Readiness for College and Careers (PARCC) and Smarter Balanced Assessment Consortium (SBAC), are developing new assessments to align with these standards. Both consortia are including substantial non-traditional items, including some multi-step problem solving. In order to align with the Common Core assessments, the 2012-13 version of MAP includes technology-enhanced items, similar to those in SOL. It remains to be seen whether such a change would account for some of the variance now explained by Cognitive Tutor variables in our models.

Process: By process, we mean the way that students go about working in Cognitive Tutor. In the SOL models, the strongest Cognitive Tutor predictor of outcomes was the number of sections completed per hour. We characterize this variable as coding some kind of efficiency; it captures students who are able to get to work and stay on task (and also who are able to generally succeed at the task). Unlike most standardized tests, online systems like Cognitive Tutor have the ability to take both time and correct performance into account.

In models of both SOL and RIT, a strong predictor was the amount of assistance (hints and errors) required by the student. Although assistance can be an indicator of lack of knowledge (leading to large numbers of hints and errors), it may also be an indicator of lack of confidence (in students who ask for hints rather than risk making an error) or gaming the system.

In this paper, we have only considered variables aggregated at the level of problem. For example, our data considers the number of hints and errors per problem but not the pattern or timing of those hints and errors. This simplification was required because more detailed data were not available for all students in this data set. However, other work [e.g. 2, 3] has shown that more detailed "detectors" of gaming and off-task behavior, which rely on patterns and timing of actions within Cognitive Tutor, can be strong predictors of outcomes. We would expect such detectors to be more sensitive to student behavior than the relatively coarse-grained measures used here.

Motivation and non-cognitive factors: Much recent work has pointed to the powerful effect that attitudes towards learning can have on standardized test outcomes [11, 12]. Pardos et al. [16] were able to use detectors of student affect (including boredom, concentration, confusion, frustration) to predict standardized test outcomes. Such affect detectors have already been developed for Cognitive Tutor [4] and, in principle, could be added to our models.

While we are very encouraged with the results that we have seen in this paper, we recognize that more detailed data may provide us better ability to predict student test outcomes from Cognitive Tutor data.

6. ACKNOWLEDGMENTS

This work was supported, in part, by LearnLab (National Science Foundation), award number SBE-0836012. Michael Yudelson provided helpful suggestions on our approach to analysis.

7. REFERENCES

[1] Arnold, K.E. 2010. Signals: Applying Academic Analytics. *Educause Quarterly*, 33, 1.

[2] Baker, R.S.J.d. 2007. Modeling and Understanding Students' Off-Task Behavior in Intelligent Tutoring Systems. In *Proceedings of ACM CHI 2007: Computer-Human Interaction*, 1059-1068.

[3] Baker, R.S., Corbett, A.T., Koedinger, K.R. 2004. Detecting Student Misuse of Intelligent Tutoring Systems. *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*, 531-540.

[4] Baker, R.S.J.d., Gowda, S.M., Wixon, M., Kalka, J., Wagner, A.Z., Salvi, A., Aleven, V., Kusbit, G., Ocumpaugh, J., and Rossi, L. 2012. Towards Sensor-Free Affect Detection in Cognitive Tutor Algebra. In *Proceedings of the Fifth International Conference on Educational Data Mining*, 126-133.

[5] Beck, J. E., Lia, P. and Mostow, J. 2004. Automatically assessing oral reading fluency in a tutor that listens. *Technology, Instruction, Cognition and Learning*, 1, pp.61-81.

[6] Corbett, A.T. and Anderson, J. R. 1992. Student modeling and mastery learning in a computer-based programming tutor. In C. Frasson, G. Gauthier and G. McCalla (Eds.), *Intelligent Tutoring Systems: Second international conference proceedings* (pp. 413-420). New York: Springer-Verlag.

[7] Corbett, A.T., Anderson, J.R. 1995. Knowledge tracing: modeling the acquisition of procedural knowledge. *User Modeling & User-Adapted Interaction* 4, (1995), 253-278.

[8] Cox, P.A. 2011. *Comparisons of Selected Benchmark Testing Methodologies as Predictors of Virginia Standards of Learning Test Scores*. Doctoral Thesis. Virginia Polytechnic Institute and State University.

[9] Feng, M., Heffernan, N.T., & Koedinger, K.R. 2009. Addressing the assessment challenge in an Online System that tutors as it assesses. *User Modeling and User-Adapted Interaction: The Journal of Personalization Research (UMUAI journal)*. 19(3), 243-266, August, 2009.

[10] Gersten, R., Beckmann, S., Clarke, B., Foegen, A., Marsh, L., Star, J. R., & Witzel, B. 2009. *Assisting students struggling with mathematics: Response to Intervention (RtI) for elementary and middle schools* (NCEE 2009-4060). Washington, DC: National Center for Education Evaluation and Regional Assistance, Institute of Education Sciences, U.S. Department of Education. Retrieved from http://ies. ed.gov/ncee/wwc/publications/practiceguides/

[11] Good, C., Aronson, J., & Harder, J. A. 2008. Problems in the pipeline: Stereotype threat and women's achievement in high-level math courses. Journal of Applied Developmental Psychology, 29, 17-28.

[12] Good, C., Aronson, J., & Inzlicht, M. 2003. Improving Adolescents' Standardized Test Performance: An Intervention to Reduce the Effects of Stereotype Threat. Journal of Applied Developmental Psychology, 24, 645-662.

[13] Koedinger, K. R., Anderson, J. R.., Hadley, W. H.., & Mark, M. A. 1997. Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education, 8*, 30-43.

[14] McCuaig, J. and Baldwin, J. 2012. Identifying successful learners from interaction behaviour. EDM 2012

[15] Pane, J., Griffin, B. A., McCaffrey, D. F. and Karam, R. 2013. Effectiveness of Cognitive Tutor Algebra I at scale. RAND Working Paper WR-984-DEIES.

[16] Pardos, Z., Baker, R.S.J.d., San Pedro, M., Gowda, S.M. & Gowda, S.M. 2013. Affective states and state tests: Investigating how affect throughout the school year predicts end of year learning outcomes.

[17] Ritter, S., Harris, T. H., Nixon, T., Dickison, D., Murray, R. C. and Towle, B. 2009. Reducing the knowledge tracing space. In Barnes, T., Desmarais, M., Romero, C., & Ventura, S. (Eds.) *Educational Data Mining 2009: 2nd International Conference on Educational Data Mining, Proceedings*. Cordoba, Spain.

[18] Ritter, S., Kulikowich, J., Lei, P., McGuire, C.L. & Morgan, P. 2007. What evidence matters? A randomized field trial of Cognitive Tutor Algebra I. In T. Hirashima, U. Hoppe & S. S. Young (Eds.), *Supporting Learning Flow through Integrative Technologies* (Vol. 162, pp. 13-20). Amsterdam: IOS Press.

[19] Northwest Evaluation Association 2012. Virginia linking study: a study of the alignment of the NWEA RIT scale with the Virginia Standards of Learning (SOL). http://www.nwea.org/sites/www.nwea.org/files/resources/VA 2012 Linking Study.pdf

[20] Northwest Evaluation Association 2013. Computer-Based Adaptive Assessments. Retrieved February 23, 2013. http://www.nwea.org/products-services/assessments/

[21] Virginia Department of Education 2013. Standards of Learning (SOL) & Testing. Retrieved February 23, 2013. http://www.doe.virginia.gov/testing/

Predicting College Enrollment from Student Interaction with an Intelligent Tutoring System in Middle School

Maria Ofelia Z. San Pedro[1], Ryan S.J.d. Baker[1], Alex J. Bowers[1], Neil T. Heffernan[2]
[1]Teachers College Columbia University, 525 W 120th St. New York, NY 10027
[2]Worcester Polytechnic Institute, 100 Institute Rd. Worcester, MA 01609

mzs2106@tc.columbia.edu, baker2@exchange.tc.columbia.edu, bowers@tc.edu, nth@wpi.edu

ABSTRACT

Research shows that middle school is an important juncture for a student where he or she starts to be conscious about academic achievement and thinks about college attendance. It is already known that access to financial resources, family background, career aspirations and academic ability are indicative of a student's choice to attend college; though these variables are interesting, they do not necessarily give sufficient actionable information to instructors or guidance counselors to intervene for individual students. However, increasing numbers of students are using educational software at this phase of their education, and detectors of specific aspects of student learning and engagement have been developed for these types of learning environments. If these types of models can be used to predict college attendance, it may provide more actionable information than the previous generation of predictive models. In this paper, we predict college attendance from these types of detectors, in the context of 3,747 students using the ASSISTment system in New England, producing detection that is both successful and potentially more actionable than previous approaches; we can distinguish between a student who will attend college and a student who will not attend college 68.6% of the time.

Keywords

College Enrollment, Affect Detection, Knowledge Modeling, Educational Data Mining

1. INTRODUCTION

The processes leading a student to choose to attend college starts early, and decisions can begin to solidify as early as middle school (ages 12-14). Especially in the United States, successful learning experiences which develop key skills build positive self-beliefs, interests, goals and actions, making students likely to actively seek and plan higher educational goals and career aspirations [31] As students go through middle school, they increasingly find themselves engaged or disengaged from school and learning. This process is driven in part by changes in students' self-perceptions, whether they see themselves as smart and capable of taking the courses in high school. This leads to students making decisions about how academic achievement, certain careers, and college majors fit into their self-perception [14].

It is during middle school that students either start to value academic achievement or begin to get off track and start to become frustrated and disengaged in school [8]. Research findings suggest that middle school students often think about going to college but fail to get support in planning how to achieve this [14].The transition to middle school in the United States has been associated with a decline in academic achievement, performance motivation, and self-perception [33]. For students who fail to develop a plan (or obtain support) for a college future, this has a dramatic effect on what eventually happens to the students.

Disengagement not only leads to negative attitudes about higher education, but also to poorer learning [17, 27, 29]. This leads to an unsuccessful learning experience when they reach high school, ultimately leading to dropping out, or disinterest in pursuing post-secondary education [7]. Multiple studies have shown that it is possible to predict which students will eventually drop out of high school, as early as late elementary or middle school [7, 9, 10, 35], with evidence that some particularly predictive factors include problem behaviors [7, 35] and shifts in academic achievement over time [9, 10]. Indicators of fairly extreme forms of disengaged behavior (low attendance, misconduct) and academic failure in sixth grade have been shown to be strong predictors of students falling off the path towards graduation and therefore eventual college attendance, within longitudinal analysis [7].

By contrast, students who have already made college plans when they are in middle school tend to be more likely to attend college in spite of challenges [13]. They tend to plan appropriate courses to take when they enter high school, and get involved in relevant extracurricular activities that contribute to college admission [14]. In effect, they become interested in achieving a good academic record. Thus, examining the factors that influence students' engagement and disengagement during middle school is crucial so as to understand better the factors that lead students to fail to attend college, and the possible paths to re-engaging students.

However, one of the major limitations to this past research is that it identifies changes in student engagement only through fairly strong indicators of disengagement, such as failing grades [9, 10, 11], problem behaviors such as violence in school [47] and non-attendance [35]. By the time these indicators are commonplace, it may be quite late to make an intervention. If it were possible to identify useful and actionable antecedents to these changes, it might be possible to intervene more effectively.

One potential source of data on early change in engagement is the log files from educational software. In recent years, the use of educational software at the middle school level has expanded considerably, with systems such as ASSISTments [40] being used by rapidly increasing numbers of students. At the same time, educational data mining (EDM) techniques have been applied to logs from educational software to model a range of affect and engagement constructs, including gaming the system [4], off-task behavior [1], carelessness [45], boredom [22, 37, 44], frustration [22, 37, 44], and engaged concentration [22, 37, 44],. Automated detectors of disengaged behaviors can predict differences in learning, both in the relatively short term [cf. 17] and over the course of a year [26, 37]. Within this paper, we extend this work to study how learning and engagement in middle school – as assessed by this type of automated detector – can be used to predict college enrollment. To our knowledge, this is the first study that aims at predicting college enrollment from affect and engagement inferred from the logs of educational software used years earlier. We conduct this research in a data set of 3,747 students who used the ASSISTment system [40], between 2004 and 2007. We discuss which aspects of learning and engagement predict college enrollment, and conclude with a discussion of potential implications for the design and interventions of interactive educational systems for sustained attendance and engagement in school.

2. METHODOLOGY

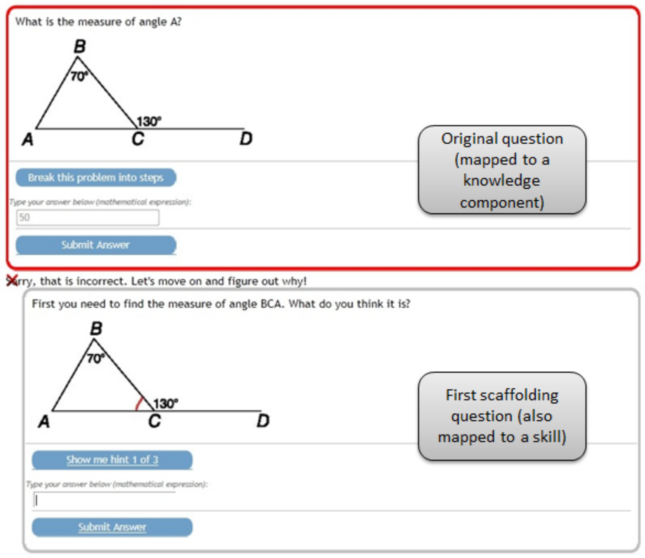
2.1 The ASSISTment System

Within this paper, we investigate this issue within the context of ASSISTments. The ASSISTment system, shown in Figure 1, [40] is a free web-based tutoring system for middle school mathematics that *assesses* a student's knowledge while *assisting* them in learning, providing teachers with detailed reports on the skills each student knows. The ASSISTment system, shown in Figure 1, provides feedback on incorrect answers. Within the system, each mathematics problem maps to one or more cognitive skills. When students working on an ASSISTment answer correctly, they proceed to the next problem. If they answer incorrectly, they are provided with scaffolding questions which break the problem down into its component steps. The last step of scaffolding returns the student to the original question (as in Figure 2). Once the correct answer to the original question is provided, the student is prompted to go to the next question.

2.2 Data

2.2.1 ASSISTments Data

Action log files from the ASSISTment system were obtained for a population of 3,747 students that came from middle schools in New England, who used the system at various times starting from school years 2004-2005 to 2006-2007 (with a few students continuing tutor usage until 2007-2008 and 2008-2009). These students were drawn from three districts who used the ASSISTment system systematically during the year. One district was urban with large proportions of students requiring free or reduced-price lunches due to poverty, relatively low scores on state standardized examinations, and large proportions of students learning English as a second language. The other two districts were suburban, serving generally middle-class populations. Overall, the students made 2,107,108 actions within the software (where an action consisted of making an answer or requesting help), within 494,150 problems, with an average of 132 problems per student. Knowledge, affect, and behavior models were applied to this dataset, creating features that could be used for our final prediction model of college enrollment.

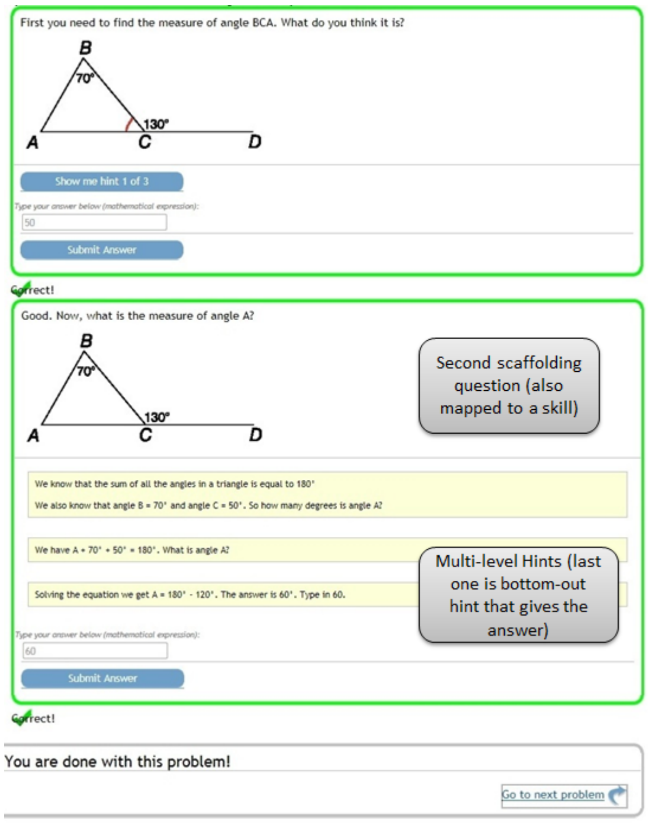

What is the measure of angle A?

rry, that is incorrect. Let's move on and figure out why!

First you need to find the measure of angle BCA. What do you think it is?

B
70°
130°
A
C
D
Break this problem into steps
Type your answer below (mathematical expression):
50
Submit Answer
Original question (mapped to a knowledge component)
Show me hint 1 of 3
Submit Answer
First scaffolding question (also mapped to a skill)

Figure 1. Example of an ASSISTments Problem. If a student gets it incorrect, scaffolding problems are there to aid the student in eventually getting the correct answer.


First you need to find the measure of angle BCA. What do you think it is?

Correct!

Good. Now, what is the measure of angle A?

We know that the sum of all the angles in a triangle is equal to 180°

We also know that angle B = 70° and angle C = 50°. So how many degrees is angle A?

We have A + 70° + 50° = 180°. What is angle A?

Solving the equation we get A = 180° - 120°. The answer is 60°. Type in 60.

Correct!

You are done with this problem!

B
70°
130°
A
C
D
Show me hint 1 of 3
Type your answer below (mathematical expression):
50
Submit Answer
Second scaffolding question (also mapped to a skill)
Multi-level Hints (last one is bottom-out hint that gives the answer)
Type your answer below (mathematical expression):
60
Submit Answer
Go to next problem

Figure 2. Example of Scaffolding in an ASSISTments Problem.

2.2.2 College Enrollment Data

For college enrollment information, enrollment records of these 3,747 students were requested from the National Student Clearinghouse (http://www.studentclearinghouse.org). For the purposes of the analyses in this paper, we identified solely whether each student was enrolled in a college or not, and used this as our labels in training our model. Additional information (such as whether the student graduated from college) is generally available from the Clearinghouse, but will not be available for these students for a few more years.

2.3 Creation of Model Features

In order to predict and analyze college enrollment, we distilled a range of features from the log files of ASSISTments, including student knowledge estimates, student affect (boredom, engaged concentration, confusion), student disengaged behaviors (off-task, gaming the system, carelessness), and other information of student usage (the proportion of correct actions and the number of first attempts on problems made by the student, a proxy for overall usage). These features were either directly distilled from the logs or obtained from automated detectors applied to the data set.

2.3.1 Student Knowledge Features

Estimates of student knowledge were computed using Bayesian Knowledge Tracing (BKT) [20], a model used in several ITSs to estimate a student's latent knowledge based on his/her observable performance. This model can predict how difficult the current problem will be for the current student, based on the skills involved in that problem. As such, this model can implicitly capture the tradeoff between difficulty and skill for the current context. This model can inform us whether student skill is higher than current difficulty (resulting in a high probability of correctness), when current difficulty is higher than student skill (resulting in a low probability of correctness), and when difficulty and skill are in balance (medium probabilities of correctness). To assess student skill, BKT infers student knowledge by continually updating the estimated probability a student knows a skill every time the student gives a first response to a new problem. It uses four parameters, each estimated separately per skill – L_O, the initial probability the student knows the skill; T, the probability of learning the skill at each opportunity to use that a skill; G, the probability that the student will give a correct answer despite not knowing the skill; and S, the probability that the student will give an incorrect answer despite knowing the skill. In this model, the four parameters for each skill are held constant across contexts and students (variants of BKT relax these assumptions). BKT uses Bayesian algorithms after each student's first response to a problem in order to re-calculate the probability that the student knew the skill before the response. Then the algorithm accounts for the possibility that the student learned the skill during the problem in order to compute the probability the student will know the skill after the problem [20]. With the data from the 2004-2005 to 2006-2007 logs, BKT model parameters were fit by employing brute-force grid search [cf. 3].

2.3.2 Affect and Behavior Features

To obtain affect and behavior assessments, we leverage existing detectors we developed of student affect and engaged/disengaged behavior within the ASSISTment system [37], to help us understand student affect and behavior across contexts. Detectors of three affective states are utilized: engaged concentration, boredom, and confusion. Detectors of three disengaged behaviors are utilized: off-task, gaming, and slip or carelessness. These detectors of affect and behavior are identical to the detectors used in [37]. They were developed in a two-stage process: first, student affect labels were acquired from the field observations (reported in [37]), and then those labels were synchronized with the log files generated by ASSISTments at the same time (forming our first dataset). This process resulted in automated detectors that can be applied to log files at scale, specifically the data set used in this project (the 2004-2005 to 2006-2007 data set). To enhance scalability, only log data was used as the basis of the detectors, instead of using physical sensors. The research presented in this paper could not have been conducted if physical sensors were used. The detectors were constructed using only log data from student actions within the software occurring at the same time as or before the observations, making our detectors usable for automated interventions, as well as the discovery with models analyses presented in this paper.

The affect detectors' predictive performance were evaluated using A' [28] and Cohen's Kappa [18]. An A' value (which is approximately the same as the area under the ROC curve [28]) of 0.5 for a model indicates chance-level performance for correctly determining the presence or absence of an affective state in a clip, and 1.0 performing perfectly. Cohen's Kappa assesses the degree to which the model is better than chance at identifying the affective state in a clip. A Kappa of 0 indicates chance-level performance, while a Kappa of 1 indicates perfect performance. A Kappa of 0.45 is equivalent to a detector that is 45% better than chance at identifying affect.

As discussed in [37], all of the affect and behavior detectors performed better than chance. Detector goodness was somewhat lower than had been previously seen for Cognitive Tutor Algebra [cf. 6], but better than had been seen in other published models inferring student affect in an intelligent tutoring system solely from log files (where average Kappa ranged from below zero to 0.19 when fully stringent validation was used) [19, 22, 44]. The best detector of engaged concentration involved the K* algorithm, achieving an A' of 0.678 and a Kappa of 0.358. The best boredom detector was found using the JRip algorithm, achieving an A' of 0.632 and a Kappa of 0.229. The best confusion detector used the J48 algorithm, having an A' of 0.736, a Kappa of 0.274. The best detector of off-task behavior was found using the REP-Tree algorithm, with an A' value of 0.819, a Kappa of 0.506. The best gaming detector involved the K* algorithm, having an A' value of 0.802, a Kappa of 0.370. These levels of detector goodness indicate models that are clearly informative, though there is still considerable room for improvement. The detectors emerging from the data mining process had some systematic error in prediction due to the use of re-sampling in the training sets (models were validated on the original, non-resampled data), where the average confidence of the resultant models was systematically higher or lower than the proportion of the affective states in the original data set. This type of bias does not affect correlation to other variables since relative order of predictions is unaffected, but it can reduce model interpretability. To increase model interpretability, model confidences were rescaled to have the same mean as the original distribution, using linear interpolation. Rescaling the confidences this way does not impact model goodness, as it does not change the relative ordering of model assessments.

2.3.2.1 Application of Affect and Behavior Models to Broader Data Set

Once the detectors of student affect and behavior were developed, they were applied to the data set used in this paper. As mentioned, this data set was comprised of 2,107,108 actions in 494,150 problems completed by 3,747 students in three school districts. The result was a sequence of predictions of student affect and behavior across the history of each student's use of the ASSISTment system.

2.3.2.2 Carelessness Detection using Logs

Different from the process above, the incidence of carelessness within the Cognitive Tutor was traced with a model designed to assess "slips" [2]. Slips in that paper are operationalized in a fashion essentially identical to prior theory of how to identify careless errors [16]. The model used in [2], termed the Contextual Slip model, contextually estimates the probability that a specific student action indicates a slip/carelessness, whenever the student reaches a problem step requiring a specific skill, but answers incorrectly. The probability of carelessness/slip is assessed contextually, and is different depending on the context of the student error. The probability estimate varies based on several features of the student action and the situation in which it occurs, including the speed of the action, and the student's history of help-seeking from the tutor. As such, the estimate of probability of carelessness/slip is different for each student action.

The Contextual Slip model is created using the BKT approach previously discussed. Note that in the BKT model – used in creating the Contextual Slip model – the four parameters for each skill are invariant across the entire context of using the tutor, and invariant across students. We use BKT as a baseline model to create first-step estimations of the probability that each action is a contextual slip. These estimations are not the final Contextual Slip model, but are used to produce it. Specifically, we use BKT to estimate whether the student knew the skill at each step. In turn, we use these estimates, in combination with Bayesian equations, to label incorrect actions with the probability that the actions were slips, based on the student performance on successive opportunities to apply the rule. More specifically, given the probability that the student knows the skill at a specific time, Bayesian equations and the static BKT parameters are utilized to compute labels for the Slip probabilities for each student action (A) at time N, using future information (two actions afterwards – N+1, N+2). In this approach, we infer the probability that a student's incorrectness at time N was due to not knowing the skill, or whether it is due to a slip. The probability that the student knew the skill at time N can be calculated, given information about the actions at time N+1 and N+2 ($A_{N+1,N+2}$), and the other parameters of the Bayesian Knowledge Tracing model:

$$P(A_N \text{ is a Slip} \mid A_N \text{ is incorrect}) = P(L_n \mid A_{N+1,N+2}). \quad (1)$$

This gives us a first estimate that a specific incorrect answer is a slip. However, this estimate uses data on the future, making it impossible to use to assess slip in real-time. In addition, there is considerable noise in these estimates, with estimates trending to extreme values that over-estimate slip in key situations due to limitations in the original BKT model [2]. But these estimated probabilities of slip can be used to produce a less noisy model that can be used in real time, by using them as training labels (e.g. inputs) to machine-learning. Specifically, a linear regression model is created that predicts slip/carelessness contextually. The result is a model that can now predict at each practice opportunity whether an action is a slip, using only data about the action itself, without any future information. This model has been shown to predict post-test scores, even after student knowledge is controlled for [3].

Once these labels are obtained from BKT, the labels are smoothed by training models with each student action originally labeled with the probability estimate of slip occurrence, using information on that student action generated on our tutor logs. For each action, a set of numeric or binary features from the ASSISTments logs were distilled, based on earlier work by [5].

As in previous work to model slipping, the features extracted from each student action within the tutor were used to predict the probability that the action represents a slip/carelessness. The prediction took the form of a linear regression model, fit using M5-prime feature selection in the RapidMiner data mining package [32]. This resulted in numerical predictions of the probability that a student action was a careless error, each time a student made a first attempt on a new problem step. Linear regression was chosen as an appropriate modeling framework when both predictor variables and the predicted variable are numeric. In addition, linear regression functions well with noisy educational data, creating relatively low risk of finding an "over-fit" model that does not function well on new data.

Six-fold student-level cross-validation [24] was conducted to evaluate the carelessness detector's goodness. Cross-validating at this level allows us to assess whether the model will remain effective for new students drawn from the same overall population of students studied. Carelessness models were trained separately per school year of the ASSISTments data set. They were assessed in terms of cross-validated correlation. The carelessness models trained within the ASSISTments data achieved a cross-validation correlation of r = 0.458 on the average.

2.4 Logistic Regression

Models were built to predict whether a student attended college. Aggregate predictor variables were created by taking the average of the predictor feature values for each student, resulting in one record per student (in other words, taking the average boredom per student, average confusion per student, etc.).

A multiple-predictor logistic regression model was fitted to predict whether a student will enroll in college from a combination of features of his or her student affect, engagement, knowledge and other information on student usage (the proportion of correct actions, and the number of first attempts on problems made by the student, a proxy for overall usage) of a tutoring system during middle school. We used logistic regression analysis since we have a dichotomous outcome – whether or not the student would be enrolled in college – resulting in a non-linear relationship between our predictors and outcome variable. Choosing logistic regression allows for relatively good interpretability of the resultant models, while matching the statistical approach used in much of the other work on predicting college attendance [12, 23, 36, 46]. In essence, the logistic model predicts the logit (natural logarithm of an odds ratio [cf. 39]) of an outcome variable from a predictor or set of predictors. The odds ratio in logistic regression is the odds of an event occurring given a particular predictor, divided by the odds of an event occurring given the absence of that particular predictor. An odds ratio over 1.0 signifies that the independent variable increases the odds of the dependent variable occurring; correspondingly, an odds ratio under 1.0 signifies that the independent variable decreases the odds of the dependent variable occurring.

Features were selected using a simple backwards elimination feature selection, based on each parameter's statistical significance. All predictor variables were standardized (using z-scores), in order to increase interpretability of the resulting odds ratios (note that this does not impact model goodness or predictive power in any fashion). The odds ratio indicates the odds that a class variable increases per one unit change of a predictor (per one SD change for standardized predictors). Standardizing the predictors enables us to show a clear indication of each predictor's contribution to the class variable (college enrollment).

3. RESULTS

Before developing the model, we looked at our original, non-standardized features and how their values compare between those who were labeled to have attended college and those who have not (Table 1).

Table 1. Features for Students who Attended College (1, n = 2166) and did not Attend college (0, n = 1581)

	College	Mean	Std. Dev.	Std. Error Mean	t-value
Slip/ Carelessness	0	0.132	0.066	0.002	-13.361
	1	0.165	0.077	0.002	(p<0.01)
Student Knowledge	0	0.292	0.151	0.004	-15.481
	1	0.378	0.180	0.004	(p<0.01)
Correctness	0	0.382	0.161	0.004	-17.793
	1	0.483	0.182	0.004	(p<0.01)
Boredom	0	0.287	0.045	0.001	5.974
	1	0.278	0.047	0.001	(p<0.01)
Engaged Concentration	0	0.483	0.041	0.001	-11.979
	1	0.500	0.044	0.001	(p<0.01)
Confusion	0	0.130	0.054	0.001	5.686
	1	0.120	0.052	0.001	(p<0.01)
Off-Task	0	0.304	0.119	0.003	1.184
	1	0.300	0.116	0.002	p=0.237
Gaming	0	0.041	0.062	0.002	8.862
	1	0.026	0.044	0.001	(p<0.01)
Number of First Actions	0	114.500	91.771	2.308	-8.673
	1	144.560	113.357	2.436	(p<0.01)

From Table 1, initial observations show that average knowledge estimate, average correct, number of first actions, average slips/carelessness, and average engaged concentration had higher mean values for students who attended college. Average boredom, average confusion, average off-task and average gaming had higher mean values for those who did not attend college. Conducting an independent samples t-test (equal variances assumed) indicates that, with the exception of off-task, the difference of means of each feature between the two groups are statistically significant.

These observations align with the individual effects of each feature on the prediction of college enrollment. For example, there is a strong positive relationship between college enrollment and average correct answers (CollegeEnrollment = 0.612 Correctness + 0.346, χ^2(df = 1, N = 3747) = 304.141, p < 0.001, Odds Ratio (Correctness) = 1.844), indicating that success in ASSISTments lead to higher probability of attending college. The same strong positive relationship is seen between college enrollment and student knowledge estimate as the student learns with ASSISTments (CollegeEnrollment = 0.543 Student Knowledge + 0.345, χ^2(df = 1, N = 3747) = 236.683, p < 0.001, Odds Ratio (StudentKnowledge) = 1.722). Engaged Concentration is also shown to positively predict college attendance (CollegeEnrollment = 0.403 Engaged Concentration + 0.325, χ^2(df = 1, N = 3747) = 140.557, p < 0.001, Odds Ratio (Engaged Concentration) = 1.497), a finding that supports studies relating this affective state to effective learning [21, 42]. And the more a student uses ASSISTments, the more likely that student will attend college (CollegeEnrollment = 0.321 Number of First Actions + 0.327, χ^2(df = 1, N = 3747) = 79.159, p < 0.001, Odds Ratio(Number of First Actions) = 1.378). One non-intuitive relationship is between carelessness and college enrollment. Taken by itself, the more a student becomes careless or commits more slips, the more likely the student is to attend college (CollegeEnrollment = 0.477 Slip/Carelessness + 0.338, χ^2(df = 1, N = 3747) = 185.208, p < 0.001, Odds Ratio(Slip/Carelessness) = 1.612), evidence in keeping with past results that careless errors are seen in more successful students [16].

Conversely, the more a student is bored, the less likely that student is to attend college (CollegeEnrollment = -0.197 Boredom + 0.318, χ^2(df = 1, N = 3747) = 35.387, p < 0.001, Odds Ratio(Boredom) = 0.821) a result in keeping with past evidence that boredom is associated with poorer learning [38], as well as high school dropout [25, 34, 43]. Confusion also is shown to be negatively associated with eventual college enrollment (CollegeEnrollment = -0.188 Confusion + 0.317, χ^2(df = 1, N = 3747) = 32.051, p < 0.001, Odds Ratio(Confusion) = 0.829). Gaming the system is also negatively correlated with eventual college enrollment (CollegeEnrollment = -0.313 Gaming + 0.314, χ^2(df = 1, N = 3747) = 78.821, p < 0.001, Odds Ratio (Gaming) = 0.731), perhaps unsurprising given its relationship with poorer learning [17].

A model for college enrollment including all data features was developed using Logistic Regression, and cross-validated at the student level (5-fold). The full data set model (Table 2) which included all features achieved a cross-validated A' of 0.686 and cross-validated Kappa value of 0.239. This model was statistically significantly better than a null (intercept-only) model, χ^2(df = 9, N = 3747) = 390.146, p < 0.001. Statistical significance was computed for a non-cross-validated model, as is standard practice.

Table 2. Full Data Set Model of College Enrollment

Features	Coefficient	Chi-Square	p-value	Odds Ratio
Student Knowledge	1.078	16.193	<0.001	2.937
Slip/ Carelessness	-1.100	25.873	<0.001	0.333
Correctness	0.758	33.943	<0.001	2.133
Boredom	0.069	0.308	0.579	1.071
Engaged Concentration	-0.175	2.207	0.137	0.839
Confusion	0.201	20.261	<0.001	1.223
Off-Task	-0.036	0.188	0.665	0.965
Gaming	-0.047	0.720	0.396	0.954
Number of First Actions	0.269	27.094	<0.001	1.308
Constant	0.354	99.735	<0.001	1.421

This model can be refined by removing all features that are not statistically significant, using a backwards elimination procedure. Our final model (Table 3) achieves a cross-validated A' of 0.686 and a cross-validated Kappa value of 0.247, almost identical to the initial model with a full data set. The reduced model is both more parsimonious and more interpretable, so it is preferred. (It is not more generalizable within the initial data set, but its parsimony increases the probability that it will be generalizable to entirely new data sets). This model is also statistically significantly better than the null model, χ^2(df = 6, N = 3747) = 386.502, p < 0.001. Our final model also achieved a fit of R^2 (Cox & Snell) = 0.098, R^2 (Nagelkerke) = 0.132, indicating that our predictors explaining 9.8% to 13.2% of the variance of those who attended college. Note that for our models, our R^2 values serve as measures of effect sizes; when converted to correlations, they represent moderate effect sizes in the 0.31-0.36 range.

Table 3. Final Model of College Enrollment

Features	Coefficient	Chi-Square	p-value	Odds Ratio
Student Knowledge	1.119	17.696	<0.001	3.062
Correctness	0.698	47.352	<0.001	2.010
Number of First Actions	0.261	28.740	<0.001	1.298
Slip/ Carelessness	-1.145	28.712	<0.001	0.318
Confusion	0.217	24.803	<0.001	1.242
Boredom	0.169	12.249	<0.001	1.184
Constant	0.351	100.011	<0.001	1.420

As can be seen in Table 3, the first three predictors (student knowledge, correctness and number of first actions) maintained the same directionality as in Table 1, but slip/carelessness, confusion and boredom flipped direction when incorporated into the final multiple logistic regression model, though each remained significant. For example, in this model, the likelihood of college enrollment increases with boredom, once the other variables are taken into account (e.g. once we control for student knowledge, software use, and so on, and other forms of disengagement). This may be because once we remove unsuccessful bored students, all that may remain are students who become bored because the material is too easy [cf. 37]. Similarly, once we control for other variables in the model, confusion is positively associated with college attendance. Again, once we remove students who are both confused and unsuccessful, all that is likely to remain are students who addressed their confusion productively [cf. 30]. For carelessness, once we control for other variables in the model, it is negatively associated with college attendance. Once we remove careless but successful students, all that is likely to remain are students who haven't overcome their carelessness [cf. 16].

4. DISCUSSION AND CONCLUSION

Many factors influence a student's decision to enroll in college. A lot of them external or social factors: financial reasons, parental support and school support. Another major factor, however, is one's ability and engagement, which develop over early years, and begin to manifest strongly during the middle school years. In this paper, we apply fine-grained models of student knowledge, student affect (boredom, engaged concentration, confusion) and behavior (off-task, gaming, slip/carelessness) to data from 3,747 students using educational software over the course of a year (or more) of middle school to understand how the development of student learning and engagement during this phase of learning, can predict college enrollment. A logistic regression model is developed, and we find that a combination of features of student engagement and student success in ASSISTments can distinguish a student who will enroll in college 68.6% of the time. In particular, boredom, confusion, and slip/carelessness are significant predictors of college enrollment both by themselves and contribute to the overall model of college enrollment.

The relationships seen between boredom and college enrollment, and gaming the system and college enrollment indicate that relatively weak indicators of disengagement are associated with lower probability of college enrollment. Success within middle school mathematics (indicated by correct answers and high probability of knowledge in ASSISTments) is positively associated with college enrollment , a finding that aligns with studies that conceptualize high performance during schooling as a sign of college readiness [41] and models that suggest that developing aptitude predicts college attendance [15, 23].

Findings in our data and final model support existing theories about indicators of college enrollment (academic achievement, grades). More importantly, it further sheds light on behavioral factors the student experiences in classrooms (which are more frequently and in many ways more actionable than the behaviors which result in disciplinary referrals). As the results here suggest, affect and disengagement are associated with college enrollment, suggesting that in-the-moment interventions provided by software (or suggested by software to teachers) may have unexpectedly large effects, if they address negative affect and disengagement. Confused students can be properly guided and encouraged to resolve their confusion. Bored students can be provided with greater novelty to reduce boredom or support in emotional self-regulation. Students who game the system can be given alternate opportunities to learn material bypassed through gaming, as in past successful interventions. Further work can be explored in the interactions of these various factors which influence our predictions.

Future endeavors in evaluating college attendance through data mining of interaction logs (pre-college) can further benefit from including additional possible interaction features in our model. Other machine learning algorithms or modeling can also be employed in our data in further understanding our research problem. It is possible that other classifiers, such as decision trees or support vector machines, may have performed better in predicting college enrollment. However, interpretability of the models may be reduced for these algorithms. Together with findings in this paper, further design considerations for educational software can be investigated that can influence not only effective learning during secondary education, but contribute as well to college interest and readiness.

5. ACKNOWLEDGMENTS

This research was supported by grants NSF #DRL-1031398, NSF #SBE-0836012, and grant #OPP1048577 from the Bill & Melinda Gates Foundation. We also thank Zak Rogoff, Adam Nakama, Aatish Salvi, and Sue Donas for their assistance in conducting the field observations, and Adam Goldstein for help in data processing.

6. REFERENCES

[1] Baker, R.S.J.d. 2007. Modeling and Understanding Students' Off-Task Behavior in Intelligent Tutoring Systems. In *Proceedings of ACM CHI 2007: Computer-Human Interaction*, 1059-1068.

[2] Baker, R.S.J.d., Corbett, A.T., and Aleven, V. 2008. More Accurate Student Modeling through Contextual Estimation of Slip and Guess Probabilities in Bayesian Knowledge Tracing. In *Proceedings of the 9th International Conference on Intelligent Tutoring Systems* (eds Aimeur E. & Woolf B.), Springer Verlag, Berlin, 406-415.

[3] Baker R.S.J.d., Corbett A.T., Gowda S.M., Wagner A.Z., MacLaren B.M., Kauffman L.R., Mitchell A.P., and Giguere S. 2010. Contextual Slip and Prediction of Student Performance after Use of an Intelligent Tutor. In *Proc. UMAP 2010*, 52-63.

[4] Baker, R.S., Corbett, A.T., and Koedinger, K.R. 2004. Detecting Student Misuse of Intelligent Tutoring Systems. In *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*, 531-540.

[5] Baker, R.S.J.d., Goldstein, A.B., and Heffernan, N.T. 2011. Detecting Learning Moment-by-Moment. *International Journal of Artificial Intelligence in Education*, 21, 5-25.

[6] Baker, R.S.J.d., Gowda, S.M., Wixon, M., Kalka, J., Wagner, A.Z., Salvi, A., Aleven, V., Kusbit, G., Ocumpaugh, J., and Rossi, L. 2012. Towards Sensor-Free Affect Detection in Cognitive Tutor Algebra. In *Proc. EDM 2012*, 126-133.

[7] Balfanz, R., Herzog, L., and Mac Iver, D. 2007. Preventing Student Disengagement and Keeping Students on the Graduation Path in the Urban Middle Grade Schools: Early Identification and Effective Interventions. *Educational Psychologist*, 42(4), 223–235.

[8] Balfanz, R. 2009. *Putting middle grades students on the graduation path: A policy and practice brief*. Baltimore, MD: Everyone Graduates Center & Talent Development Middle Grades Program, Johns Hopkins University.

[9] Bowers, A.J. 2010. Analyzing the Longitudinal K-12 Grading Histories of Entire Cohorts of Students: Grades, Data Driven Decision Making, Dropping Out and Hierarchical Cluster Analysis. *Practical Assessment, Research & Evaluation (PARE)*, 15(7), 1-18.

[10] Bowers, A.J. 2010. Grades and Graduation: A Longitudinal Risk Perspective to Identify Student Dropouts. *The Journal of Educational Research*, 103(3), 191-207.

[11] Bowers, A.J., Sprott, R., and Taff, S.A. 2013. Do we Know Who Will Drop Out? A Review of the Predictors of Dropping out of High School: Precision, Sensitivity and Specificity. *The High School Journal*. 96(2), 77-100.

[12] Cabrera, A. F. 1994. Logistic regression analysis in higher education: An applied perspective. *Higher Education: Handbook of Theory and Research*, Vol. 10, 225-256.

[13] Cabrera, A. F., La Nasa, S. M. and Burkum, K, R. 2001. *Pathways to a Four-Year Degree: The Higher Education Story of One Generation*. Center for the Study of Higher Education. Penn State University.

[14] Camblin, S. 2003. The middle grades: Putting all students on track for college. Honolulu, HI: *Pacific Resources for Education and Learning*.

[15] Christensen, J. Melder, and B. Weisbrod. 1975. Factors affecting college attendance. *Journal of Human Resources*, 10 (1975), pp. 174–188

[16] Clements, K. 1982. Careless errors made by sixth-grade children on written mathematical tasks. *Journal for Research in Mathematics Education*, 13, 136–144.

[17] Cocea, M., Hershkovitz, A., and Baker, R.S.J.d. 2009. The Impact of Off-task and Gaming Behaviors on Learning: Immediate or Aggregate? In *Proceedings of the 14th International Conference on Artificial Intelligence in Education*, 507-514

[18] Cohen, J. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement* 20, 1, 37-46.

[19] Conati, C., and Maclaren, H. 2009. Empirically building and evaluating a probabilistic model of user affect. *User Modeling and User-Adapted Interaction* 19, 3, 267-303.

[20] Corbett, A.T., and Anderson, J.R. 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction* 4, 4, 253-278.

[21] Craig, S.D., Graesser, A.C., 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, 241-250.

[22] D'Mello, S.K., Craig, S.D., Witherspoon, A. W., McDaniel, B. T., and Graesser, A. C. 2008. Automatic Detection of Learner's Affect from Conversational Cues. *User Modeling and User- Adapted Interaction*, 18 (1-2), pp. 45-80.

[23] Eccles, J. S., Vida, M. N., and Barber, B. 2004. The relation of early adolescents' college plans and both academic ability and task-value beliefs to subsequent college enrollment. *Journal of Early Adolescence*, 24, 63-77.

[24] Efron, B. and Gong, G. 1983. A leisurely look at the bootstrap, the jackknife, and cross-validation. *American Statistician*, 37, 36–48.

[25] Farrell, Edwin. 1988. Giving Voice to High School Students: Pressure and Boredom, Ya Know What I'm Sayin'? *American Educational Research Journal* 4, 489-502.

[26] Feng, M., Heffernan, N.T., and Koedinger, K.R. 2009. Addressing the assessment challenge in an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction: The Journal of Personalization Research (UMUAI). 19*(3), pp. 243-266.

[27] Fredricks, J. A., Blumenfeld, P. C., and Paris, A. H. 2004. School engagement: Potential of the concept, state of the evidence. *Review of Educational Research*, 74, 59–109.

[28] Hanley, J., and McNeil, B. 1982. The Meaning and Use of the Area under a Receiver Operating Characteristic (ROC) Curve. *Radiology* 143, 29-36.

[29] Lau, S. and Liem, A.D. 2007. The role of self-efficacy, task value, and achievement goals in predicting learning strategies, task disengagement, peer relationship, and achievement outcome. *Contemporary Educational Psychology*, 3, 1-26.

[30] Lee, D.M., Rodrigo, M.M., Baker, R.S.J.d., Sugay, J., and Coronel, A. 2011. Exploring the Relationship Between Novice Programmer Confusion and Achievement. In *Proceedings of the 4th bi-annual International Conference on Affective Computing and Intelligent Interaction, 2011*.

[31] 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, 79-122.

[32] Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M. and Euler, T. 2006. YALE: rapid prototyping for complex data mining tasks. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (eds Ungar L., Craven M., Gunopulos D. & Eliassi-Rad T.), ACM, New York, 935-940.

[33] National Middle School Association research summary #12: Academic achievement. 2002. Retrieved January 14, 2003, from www.nmsa.org/research/ressum12.htm

[34] National Research Council & Institute of Medicine. 2004. *Engaging schools: Fostering high school students' motivation to learn*. Washington, DC: National Academy Press.

[35] Neild, R. C.2009. Falling Off Track during the Transition to High School: What We Know and What Can Be Done. *The Future of Children* 19(1), 53-76. Princeton University.

[36] Nunez, A.M. and Bowers, A.J. 2011. Exploring What Leads High School Students to Enroll in Hispanic-Serving Institutions: A multilevel analysis. *American Educational Research Journal*, 48(6), 1286-1313.

[37] Pardos, Z., Baker, R.S.J.d., San Pedro, M.O.Z., Gowda, S.M., and Gowda, S. 2013. Affective states and state tests: Investigating how affect throughout the school year predicts end of year learning outcomes. In *Proceedings of the 3rd International Conference on Learning Analytics and Knowledge*, 117-124.

[38] Pekrun, R., Goetz, T., Daniels, L.M., Stupnisky, R.H. and Perry, R.P. 2010. Boredom in achievement settings: Exploring control–value antecedents and performance outcomes of a neglected emotion. *Journal of Educational Psychology* 102, 531-549.

[39] Peng, C.-Y. J., Lee, K. L., and Ingersoll, G. M. 2002. An introduction to logistic regression analysis and reporting. *The Journal of Educational Research*, 96(1), 3-14.

[40] Razzaq, L., Feng, M., Nuzzo-Jones, G., Heffernan, N.T., Koedinger, K. R., Junker, B., Ritter, S., Knight, A., Aniszczyk, C., Choksey, S., Livak, T., Mercado, E., Turner, T.E., Upalekar. R, Walonoski, J.A., Macasek. M.A. and Rasmussen, K.P. 2005. The Assistment project: Blending assessment and assisting. In *Proc. AIED 2005*, 555-562.

[41] Roderick, M., Nagaoka, J., and Coca, V. 2009. College readiness for all: The challenge for urban high schools. *The Future of Children* 19(1): 185-210.

[42] Rodrigo, M.M.T., Baker, R.S., Jadud, M.C., Amarra, A.C.M., Dy, T., Espejo-Lahoz, M.B.V., Lim, S.A.L., Pascua, S.A.M.S., Sugay, J.O., and Tabanao, E.S. 2009. Affective and Behavioral Predictors of Novice Programmer Achievement. In *Proc. ACM-SIGCSE 2009*, 156-160.

[43] Rumberger, R. W. 1987. High school dropouts: A review of issues and evidence. *Review of Educational Research*, 57, 101-121

[44] Sabourin, J., Mott, B., and Lester, J. 2011. Modeling Learner Affect with Theoretically Grounded Dynamic Bayesian Networks. In *Proc. ACII 2011*, 286-295.

[45] San Pedro, M.O.C., Baker, R., and Rodrigo, M.M. 2011. Detecting Carelessness through Contextual Estimation of Slip Probabilities among Students Using an Intelligent Tutor for Mathematics. In *Proceedings of 15th International Conference on Artificial Intelligence in Education*, 304-311

[46] Stephan, J. and Rosenbaum, J. 2012. Can High Schools Reduce College Enrollment Gaps with a New Counseling Model? *Educational Evaluation and Policy Analysis*, first published on October 16, 2012. DOI: 10.3102/0162373712462624.

[47] Tobin, T. J. and Sugai, G. M. 1999. Using sixth-grade school records to predict school violence, chronic discipline problems, and high school outcomes. *Journal of Emotional and Behavioral Disorders*, 7, 40-53.

Incorporating Scaffolding and Tutor Context into Bayesian Knowledge Tracing to Predict Inquiry Skill Acquisition

Michael A. Sao Pedro
Worcester Polytechnic Institute
100 Institute Rd.
Worcester, MA 01609 USA
(508) 831-5000
mikesp@wpi.edu

Ryan S.J.d. Baker
Teacher's College, Columbia
525 W. 120th St., Box 118
New York, NY 10027 USA
(212) 678-8329
ryan@educationaldatamining.org

Janice D. Gobert
Worcester Polytechnic Institute
100 Institute Rd.
Worcester, MA 01609 USA
(508) 831-5000
jgobert@wpi.edu

ABSTRACT
In this paper, we incorporate scaffolding and change of tutor context within the Bayesian Knowledge Tracing (BKT) framework to track students' developing inquiry skills. These skills are demonstrated as students experiment within interactive simulations for two science topics. Our aim is twofold. First, we desire to improve the models' predictive performance by adding these factors. Second, we aim to interpret these extended models to reveal if our scaffolding approach is effective, and if inquiry skills transfer across the topics. We found that incorporating scaffolding yielded better predictions of individual students' performance over the classic BKT model. By interpreting our models, we found that scaffolding appears to be effective at helping students acquire these skills, and that the skills transfer across topics.

Keywords
Science Microworlds, Science Simulations, Science Inquiry, Automated Inquiry Assessment, Educational Data Mining, Validation, Bayesian Knowledge Tracing, User Modeling

1. INTRODUCTION
Many extensions to the classic Bayesian Knowledge Tracing (BKT) model [1] have been developed to improve performance at predicting skill within intelligent tutoring systems, and to increase the interpretability of the model. For example, extensions have been made to account for individual student differences [2, 3], to incorporate item difficulty [4], to address learning activities requiring multiple skills [5], and even to incorporate the effects of automated support given by the system [6-8]. Extensions have also been added to increase model interpretability and to provide insight about tutor effectiveness. For example, [6] incorporated scaffolding into BKT to determine if automated support improved students' learning and performance. However, taking into account the differences in tutor contexts, the different facets of an activity or problem in which the same skills are applied, has only been studied in a limited fashion ([8] is one of the few examples). Context is important to consider because skills learned or practiced in one context may not transfer to new contexts [9], [10]. This, in turn, could reduce a model's predictive performance if it is to be used across contexts. Explicitly considering the context in which skills are applied within knowledge modeling may also increase model interpretability and potentially reveal whether some skills are more generalizable, and thus transferrable.

In this paper, we explore the impacts of incorporating two new elements to the BKT framework to track data collection inquiry skills [cf. 11] within the Inq-ITS inquiry learning environment [12]. These elements are scaffolding and change of tutor context. Like [6-8], we incorporate scaffolding by adding an observable and model parameters to account for its potential impacts on learning. We also add parameters and observables to account for change in the tutor context. In this work, we focus on one kind of tutor context, the specific science topic in which students practice and demonstrate inquiry skills. Predictive performance and interpretability of these extensions is addressed using data gathered from students who engaged in inquiry learning within scaffolded Inq-ITS activities on two Physical Science topics [12].

These proposed extensions are motivated by our prior work [13], [14] in constructing BKT models to track skills within an *unscaffolded* activities on a single science topic. Though these models could predict students' performance, we noticed they had very low learning rate parameters. Since then, we added scaffolding to these activities that automatically provides feedback to students when they engage in unproductive data collection. By incorporating scaffolding into our BKT models, we aim to improve prediction and to determine the degree to which scaffolding impacts skill acquisition. In other words, to paraphrase Beck et al. [6], we want to know "Does our help help?" Explicitly modeling this improvement may enhance the learning environment's ability to predict performance. In particular, if the scaffolding we provided is effective, we expect that learning rate should increase when students receive help by the system [cf. 6].

Similarly, the science topic (the context) in which skills are enacted may also play a role in models' predictive capabilities. Specifically, it is possible that inquiry skills may be tied to the science topic in which they are learned [15]. In other words, students who practice and learn inquiry skills in one science topic may not be successful at transferring skill to other science topics [cf. 9, 10]. Thus, from the viewpoint of predicting student performance, changing topics may reduce the success of a standard BKT model at predicting future performance. By explicitly modeling this, we may be able to improve models' predictive capabilities. In addition, by explicitly incorporating the science topic into models, it may become possible to discern from the model parameters the degree to which inquiry skills transfer.

2. INQ-ITS LEARNING ENVIRONMENT
We developed our models to track students' scientific inquiry skills within the Inq-ITS learning environment (www.inq-its.org), formerly known as Science Assistments [12]. This environment aims to automatically assess and scaffold students as they experiment with interactive simulations across several science topic areas such as Physical, Life, and Earth Science. Each Inq-ITS activity is a performance assessment of a range of inquiry skills; the actions students take within the simulation and work products they create are the bases for assessment.

Inq-ITS inquiry activities all have a similar look-and-feel. Each activity provides students with a driving question, and requires them to conduct an investigation with a simulation and inquiry support tools to address that question. These inquiry support tools include a hypothesizing widget, a data analysis widget, and graphs and tables for automatically displaying and summarizing data. The tools not only help students explore and keep track of their progress, but also enable assessment because they make students' thinking explicit [12].

The system also delivers scaffolds to students in a text-based format via a pedagogical agent named Rex, a cartoon dinosaur, shown in Figure 1. Primarily, Rex provides real-time feedback to students as they engage in inquiry. In other words, the system can "jump in" and support students as they work. Determination of who receives scaffolding is performed using both EDM-based detectors and knowledge engineered rules [12]. We will elaborate on these approaches to evaluate the data collection skills relevant to this paper in Section 4.

In this paper, we focus on tracking skills across two Physical Science Topics, Phase Change and Free Fall. We now present an overview of the inquiry activities pertaining to these topics.

2.1 Phase Change and Free Fall Activities

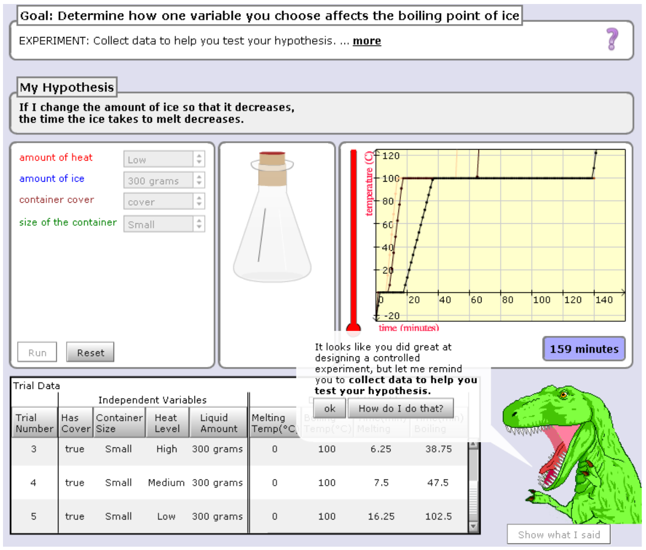
The Phase Change activities [12] (Figure 1) foster understanding about the melting and boiling properties of ice. In these activities, students are given an explicit goal to determine if one of three factors (size of a container, amount of ice to melt, and amount of heat applied to the ice) affects various measurable outcomes (e.g. melting or boiling point). Students then formulate hypotheses, collect and interpret data, and warrant their claims to address the goal. The inquiry process begins by having students articulate a hypothesis to test using a hypothesis widget [12]. The widget is set of pulldown menus that provide a template of a hypothesis. For example, a student may state: "If I change the container size so that it decreases, the time to melt increases."

After stating a hypothesis, students then "experiment" by collecting data to test their hypotheses (see Figure 1). Here, students are shown the Phase Change simulation and graphs that track changes of the ice's temperature over time. Students change the simulation's variables, and then run, pause and reset it to collect their data (trials). A data table tool is also present that shows all the data collected thus far.

Once students decide they collected enough data, they move to the final task, "analyze data". Similar to hypothesizing, students use pull-down menus to construct an argument whether their hypotheses were supported based on the data they collected [12].

The second set of activities we developed, the Free Fall activities [11], are similar to the Phase Change activities. These activities aim to foster understanding about factors that influence the kinetic, potential and mechanical energy of a ball when it is dropped. In these activities, students again try to address a driving question related to Free Fall by conducting an investigation.

As students collect data, they can receive real-time feedback from Rex (if feedback is turned on), as soon as the system detects they are not engaging in productive data collection (Figure 1). For example, if the system detects that a student is designing controlled experiments but is not collecting data to test their hypothesis (two skills associated with good data collection [12]), Rex will tell them "It looks like you did great at designing a controlled experiment, but let me remind you to collect data to help your test your hypotheses." If the student continues struggling, "bottom-out" feedback is given [cf. 1]: "Let me help some more. Just change the [IV] and run another trial. Don't change the other variables. Doing this lets you tell for sure if changing the [IV] causes changes to the [DV]" ([IV] and [DV] are replaced with the student's exact hypothesis) Thus, Rex's scaffolds provide multi-level help, with each level providing more specific, targeted feedback when the same error is made repeatedly, similar to Cognitive Tutors [e.g. 1]. A goal of this paper is to gain insight about the efficacy of this scaffolding approach.


Goal: Determine how one variable you choose affects the boiling point of ice

EXPERIMENT: Collect data to help you test your hypothesis. ... **more**

My Hypothesis

If I change the amount of ice so that it decreases, the time the ice takes to melt decreases.

amount of heat: Low
amount of ice: 300 grams
container cover: cover
size of the container: Small

It looks like you did great at designing a controlled experiment, but let me remind you to **collect data to help you test your hypothesis.**

Trial Data

Trial Number	Has Cover	Container Size	Heat Level	Liquid Amount	Melting Temp(°C)	Boiling Temp(°C)	Time(min) Melting	Time(min) Boiling
	Independent Variables							
3	true	Small	High	300 grams	0	100	6.25	38.75
4	true	Small	Medium	300 grams	0	100	7.5	47.5
5	true	Small	Low	300 grams	0	100	16.25	102.5

Figure 1. Pedagogical Agent Rex automatically provides support to students as they experiment with a Phase Change simulation in the Inq-ITS learning environment.

3. PARTICIPANTS AND PROCEDURE

We collected data from 299 eighth grade students as they engaged in inquiry within Inq-ITS. These students attended three different schools in suburban Central Massachusetts. Students at each school had the same teacher, and were separated into class sections. Some had prior experience conducting inquiry in Inq-ITS, and for others, this was their first experience.

These data were collected as part of a study to determine the impacts of automated scaffolding on acquisition and transfer of data collection skills across science topics. In this study, students were assigned 5 Phase Change inquiry activities, and two weeks later, 5 Free Fall activities. Students were allotted approximately two class periods per science topic to complete the activities. Due to time constraints, some students did not finish all the activities in each science topic.

Recall that in each activity, students formulated hypotheses, collected data and analyzed data. In the first 4 Phase Change activities, all students had scaffolding available as they formulated hypotheses. However, some students were randomly chosen to have data collection scaffolds available, whereas others did not. In the scaffolding condition, Rex (Figure 1) provided feedback to the students when they were evaluated as not demonstrating good data collection behavior. Students who were in the no-scaffolding condition received no feedback on their data collection. In the "analyze data" inquiry task, no students received scaffolding.

Both groups then completed a fifth Phase Change activity with no scaffolding. This enabled us to measure immediate impacts of scaffolding on skill acquisition within the same science topic. Approximately two weeks later, all students engaged in inquiry within the Free Fall activities. Students did not receive any feedback on their data collection within these activities. These activities were used to determine the impacts of scaffolding on transfer of skill across science topics.

4. EVALUATING THE DEMONSTRATION OF DATA COLLECTION SKILL

Within this work, we used automated methods for evaluating data collection skills [13, 16, 17]. This evaluation was used both to trigger scaffolding, and to provide observables of student performance for building Bayesian Knowledge Tracing models. Specifically, we aim to assess two process skills associated with productive data collection, designing controlled experiments and testing stated hypotheses [12]. These are demonstrated as students collect data using the simulation in the “experiment” stage of inquiry. Briefly, students design controlled experiments when they generate data that make it possible to determine what the effects of independent variables (factors) are on outcomes. They test stated hypotheses when they generate data that can support or refute an explicitly stated hypothesis. These skills are separable; students may test their hypotheses with confounded designs, or may design controlled experiments for a hypothesis not explicitly stated. Since these are process skills, students are assessed based on the actions they take while collecting data.

We evaluate whether students demonstrate these skills by combining predictions made by data-mined detectors [13], with knowledge-engineered rules to handle specific edge cases. This process works briefly as follows. Detectors were constructed by applying machine learning to predict labels of student skill. These labels were generated using text replay tagging on log files [18] from students’ interactions within the Phase Change activities. In this process, a human coder labels whether or not students are demonstrating the inquiry skills by viewing a “chunk” of student actions (the text replay) that has been formatted to highlight relevant information to make that coding easier. These labels can be used as “ground truth” for whether or not students demonstrate a skill, and subsequently for building and validating detectors that replicate human judgment.

To validate our detectors, we tested their predictive performance against *held-out test sets* of student data, data not used to construct the detectors. It is important to note that the students considered in this paper were not used to build the detectors. Performance was measured using A' [19] and Kappa (κ). Briefly, A’ is the probability that when given two examples of students’ data collection, one labeled as demonstrating skill and one not, a detector will correctly label the two. A’ is identical to the Wilcoxon statistic, and approximates the area under the ROC curve [19]. A' of 0.5 indicates chance-level performance, 1.0 indicates perfect performance. Cohen’s Kappa (κ) determines the degree to which the detector matches raw human judgment, with κ = 0.0 indicating chance-level performance and κ = 1.0 indicating perfect performance.

Using this validation process, we demonstrated that our detectors of can be used to evaluate students’ inquiry in Phase Change when they complete their experimentation [17]. More specifically, the designing controlled experiments detectors work well when students have run the simulation at least three times (thus collecting three pieces of data) in their experimentation. For data collections of this type, the detectors can distinguish a student who has designed controlled experiments when they have completed their data collection from a student who has not A’ = 94% of the time. They also could identify the correct class extremely well, κ = .75. The testing stated hypotheses detector also predicted quite well, without the limitation on the number of trials collected by the student, A’ = .91, κ = .70.

We also found that these detectors could also be used as-is to drive scaffolding in Phase Change [17], *before* students finished collecting their data. The designing controlled experiments detector could successfully be applied by the student’s third data collection with the simulation, and the testing stated hypotheses detector could be applied in as few as two simulation runs.

Finally, these detectors have been shown to generalize to evaluate skill within the Free Fall activities [11], a different science topic from which they were built (Phase Change), and an entirely different cohort of students. Under student-level stratification, the designing controlled experiments detector could distinguish a student who designed controlled experiments from one who did not A’ = 90% of the time, and highly agreed with a human coder’s ratings, κ = .65. Performance for the testing stated hypotheses detector was also high, A’ = .91, κ = .62.

As mentioned, though performance of these detectors is quite good for evaluation of data collection skill and for driving scaffolding, there are edge cases where the detectors did not perform as well. In particular, the designing controlled experiments detector cannot be applied when students collect only 1 or 2 pieces of data with the simulation. The testing stated hypothesis detector cannot be applied when the student collects only a single trial. In these cases, which are well-defined, we authored simple knowledge engineered rules to evaluate students’ data collection for a single trial [20] and two trials [21, 22].

Thus overall, combining data mining and knowledge engineering enabled successful evaluation of students’ data collection process skills. In the next section, we describe the data distilled from students’ usage of the Phase Change and Free Fall activities. These data are used to develop and test the BKT extensions.

5. DATASET FOR BKT MODELS

Students’ skill demonstration was evaluated by the detectors and knowledge engineered rules outlined in Section 4. A full profile of student performances was generated for each skill and each activity. These evaluations are the observations used to build BKT models of latent skill.

Certain students and evaluations were removed. First, we only consider students’ first opportunity to demonstrate skill prior to receiving scaffolding. More specifically, students can continue to collect data after they receive scaffolding, and be re-evaluated. These additional evaluations are not included in the data set. We do this to control for the possibility that specific scaffolds in our multi-level scaffolding approach may differentially impact learning. Thus, we look for the overall effects of scaffolding. Second, we removed 12 students who did not complete both the Phase Change and Free Fall activities due to absence. The final dataset contained 5878 unique evaluations of 287 students’ inquiry, 2939 evaluations for each data collection skill.

6. EXTENSIONS TO BKT

We amalgamated students’ performances across activities within a Bayesian Knowledge-Tracing framework [1]. BKT is a two-state Hidden Markov Model that estimates the probability a student possesses latent skill (L_n) after *n* observable practice opportunities ($Prac_n$). In our work, latent skill is knowing how to perform the data collection skills, and a practice opportunity is an evaluation of whether skill was demonstrated during data collection in an inquiry activity. A practice opportunity begins when students enter the "experiment" task in an inquiry activity. An opportunity ends when a student switches from the "experiment" task to the 'analyze data" task (see Section 2.1). As mentioned, the detectors / knowledge engineered rules evaluate students' actions, and these evaluations act as the observables. A student is evaluated as not having demonstrated skill ($Prac_n$ = 0) if one of two cases occurs. The first is if they are evaluated as not demonstrating a skill when they signal completion of data collection (e.g. attempt to switch to the "analyze data" task). The second is if, while collecting data, the system believes the student does not know either skill and provides scaffolding. This approach to address scaffolding's impact on student correctness is similar to others [e.g. 4].

The classic BKT model [1] is characterized by four parameters, *G*, *S*, L_0, and *T*. The Guess parameter (*G*) is the probability the student will demonstrate the skill despite not knowing it. Conversely, the Slip parameter (*S*) is the probability the student will not demonstrate the skill even though they know it. L_0 is the initial probability of knowing the skill before any practice. Finally, *T* is the probability of learning the skill between practice attempts. From these values, the likelihood of knowing a skill P(L_n) is computed as follows:

$$P(L_n) = P(L_{n-1}|Prac_n) + \left(1 - P(L_{n-1}|Prac_n)\right) * T, \text{ where}$$

$$P(L_{n-1}|Prac_n = 1) = \frac{P(L_{n-1}) * (1-S)}{P(L_{n-1}) * (1-S) + \left(1 - P(L_{n-1})\right) * G}$$

$$P(L_{n-1}|Prac_n = 0) = \frac{P(L_{n-1}) * S}{P(L_{n-1}) * S + \left(1 - P(L_{n-1})\right) * (1-G)}$$

This classic BKT model [1] carries a few assumptions. First, the model assumes that a students' latent knowledge of a skill is binary; either the student knows the skill or does not. The model also assumes one set of parameters per skill and that the parameters are the same for all students. Finally, the classic model assumes that students do not forget a skill once they know it.

Relevant to this work, the classic BKT model does not take into account whether students received any scaffolding from the learning environment [6] and does not account for the topic in which skills are demonstrated [8]. The same skill in different topics would either be treated as two separate skills (assuming no transfer), or as having no differences between topics (assuming complete transfer). Both of these assumptions are thought to be questionable [10, 23]. Below, we describe our approach to incorporate both of these factors.

6.1 Taking Scaffolding into Account

We introduce scaffolding into BKT as an observable, *Scaffolded*$_n$ = {True, False}, because it can directly be seen if our pedagogical agent provided help to students as they collected data. A similar approach was taken by [6] to develop the Bayesian Evaluation and Assessment model. In their domain, reading, this scaffolding observable was true if a student received help just before reading a word (each word was treated as a skill). The observable was linked to all four BKT model parameters, meaning that scaffolding could have an impact on initial knowledge (L_0), guess (*G*), slip (*S*) and whether or not students learn between practice opportunities (*T*). As a result, their BKT model contained 8 parameters to account for scaffolding.

Unlike [6], we instead chose to condition *only* the learning rate (*T*), for three reasons. First, the increase in the number of parameters could result in overfitting, especially since the classic BKT model is already known to be overparametrized [24]. Second, though the additional parameters may facilitate model interpretation, it is unclear whether conditioning all the classic BKT parameters on scaffolding improves predictive performance. In particular, [6] found no increase in predictive performance when accounting for scaffolding. Finally, the immediate effects of scaffolding on performance may not be relevant because we only look at first practice opportunities (thus looking at overall effects of scaffolding), and because there is a time delay between data collection performance attempts. In particular, students attend to a different inquiry task, analyzing data, after their data collection (see Section 2.1 for more details).

In our extension, conditioning learning on whether students receive scaffolding yields two learning rate parameters, *T_scaffolded* and *T_unscaffolded*. Thus, this model tries to account for the differential impacts scaffolding may have on whether or not students learn a skill (e.g. the latent variable knowledge transitions from "doesn't know" to "know" after practicing). Mathematically, the original equation for computing P(L_n) is conditionalized to account for the observable as follows:

$$P(L_n|Scaffolded_n = True) = P(L_{n-1}|Prac_n) + \left(1 - P(L_{n-1}|Prac_n)\right) * P(T_{scaff})$$

$$P(L_n|Scaffolded_n = False) = P(L_{n-1}|Prac_n) + \left(1 - P(L_{n-1}|Prac_n)\right) * P(T_{unscaff})$$

6.2 Taking Science Topic (Context) into Account

We also developed BKT extensions to the take into account the science topic in which students demonstrate their inquiry skills. Recall that students first practiced inquiry in Phase Change activities (possibly scaffolded or unscaffolded) and then practiced inquiry in unscaffolded Free Fall activities, a different science topic. As mentioned, modeling the change in science topic is of important since the degree to which inquiry skills transfer across topics is unclear [15]. We hypothesize that incorporating the change of science topic into our BKT framework may improve models' predictive performance.

We incorporate changing of science topics in two ways. First, we hypothesized that there may be a differential effect in learning between topics. For example, practice in Phase Change may prepare students to learn (and subsequently demonstrate) skills in Free Fall, called "preparation for future learning" [23]. To model differential learning between topics, we again break out the learning rate (*T*), this time for each topic: *T_PhCh*, *T_FF*. A new observable is also added for the current science topic, *Topic*$_n$ = {PhaseChange, FreeFall}. The result is a "BKT learn rate topic" model with a modification to the P(L_n) equation similar to the "scaffolded BKT model" described previously.

Our second model for incorporating the change of science topics posits that students may not understand that the skills are applicable across topics. We model this notion by adding in a linear degradation factor, $k \in (0,1)$, to potentially offset the likelihood students know the skill P(L_n) when the science topic switches. If k = 1 this implies there is no effect on students' knowledge when the topic switches. When k = 0, students will be presumed to not know the skill when the topic switches. One benefit of this approach is that it relaxes the assumption of skill independence if we had chosen to fit separate classic BKT models per skill, per science topic. Instead, k captures the potential for partial transfer of skill between science topics [cf. 10]. We also add an observable *Topic_Switch*$_n$ = {True, False} to address when the science topic changes from Phase Change to Free Fall (just before the student's first opportunity to practice in Free Fall). The corresponding P(L_n) modification for the "BKT skill degradation model" is:

$$P(L_n|Topic\_Switch_n = True) = k * [P(L_{n-1}|Prac_n) + (1 - P(L_{n-1}|Prac_n)) * T]$$

$$P(L_n|Topic\_Switch_n = False) = P(L_{n-1}|Prac_n) + (1 - P(L_{n-1}|Prac_n)) * T$$

Note that the degradation parameter k is different than modeling "forgetting" in the BKT framework [cf. 1, 8] in two ways. First, we note that the factor is applied to both conditional expressions in the P(L_n) equation, not just P(L_{n-1}|$Prac_n$) as done when modeling forgetting. Second, in these earlier approaches forgetting is modeled at each practice opportunity, whereas our factor is applied at a single point, when the science topic switches.

6.3 Combining Models

The above models introduce three new potential observables to the BKT framework relevant to our learning environment: *Scaffolded?* = {True, False}, *Topic* = {PhaseChange, FreeFall}, and *Topic_Switch?* = {True, False}. The models above individually incorporate the observables by conditioning the learning rate parameter, *T*, on them, or by adding a multiplicative reduction factor, *k*, to the computation of P(L_n). As part of this work, we also combined the extensions described above into larger models. The most complicated model incorporated all observables and contained seven parameters: (L_0, *G*, *S*, *T_Scaff_PhCh*, *T_Unscaff_PhCh*, *T_Unscaff_FF*, *k*). We next describe our process for fitting these models.

6.4 Model Fitting

As in [3], [13], we use brute force search to find the best fitting parameters. This method has been found to produce comparable or better model parameters than other methods [25]. In this approach, all potential parameter combinations in the search space are tried at a grain-size of 0.01. The best parameter set yields the lowest sum of squares residual (SSR) between the likelihood that the student would demonstrate skill, P(*Show_Skill*$_n$), and the actual data. This likelihood is computed as follows [1]:

$$P(Show\_skill_n) = P(L_{n-1}) * (1 - S) + (1 - P(L_{n-1})) * G$$

Once this set has been found, another brute force search around those parameters is run at a grain-size of 0.001 to find a tighter fit. We bound *G* to be less than 0.3 and *S* to be less than 0.1 [cf. 25]; all other parameters can be assigned values in (0.0, 1.0).

When fitting our models, we found the brute force search to be realistically tractable only up to fitting 5 parameter models. To fit the combined models with more parameters, we used a two-stage process. First, we fit a classic BKT model with four parameters (L_0, *G*, *S*, *T*). Then, we fit a combined model using fixed values for *G* and *S* from the classic model. These parameters were fixed because we believe the extended models described above will have the most impact on estimates of learning between practice opportunities and initial knowledge, not on guessing and slipping.

7. RESULTS

We determine if extending the classic BKT model to include scaffolding and changing of science topics will 1) improve predictions of future student performance in our learning environment, and 2) yield insights about the effectiveness of our scaffolding approach, and the transferability of the inquiry skills.

To address predictive performance, we determined if the new models' predictions of skill demonstration P(*Show_Skill*$_n$), aggregated from evidence over times {1…*n*-1}, can predict actual student performance at time *n* better than the classic BKT model. We train and test our models' performance by conducting six-fold student-level cross-validation, stratifying by both learning condition (having scaffolding available in Phase Change or not) and class section. Cross-validating in this way helps ensure that each fold equally represents learning conditions, and students from each class section/school. This increases assurance that models can be applied to new students.

As in [13], model goodness was determined using A' [19]. This is an appropriate metric to use when the predicted value is binary (either students demonstrated skill in *Prac*$_n$ or they did not), and the predictors for each model are real-valued, e.g. P(*Show_skill*$_n$). As a reminder, a model with A' of 0.5 predicts at chance level and a model with A' of 1.0 predicts perfectly.

Two variants on A' for student performance data are computed as follows. First, we compute overall A' values of each model collapsing over students as we did in [13]. Second, we compute the A' values of each model per student [3], and report the average per-student A'. These approaches have different strengths and weaknesses [cf. 3, 13, 25]. Collapsing over students is straightforward and enables comparison of models' broad consistency in predicting skill demonstration. In other words, this approach can show, in general, whether or not high likelihoods of demonstration of skill predicted by the model correspond with actual demonstration of skill. In addition, collapsing can be used when there is not enough within variance for each student to produce a meaningful per student A' [cf. 13]. Collapsing over students, however, provides weaker estimates of predicting an individual student's learning and performance than the A' per student metric [3, 25]. Collapsing may also yield estimates that are biased towards students who practiced more with the system since they contribute more data [25].

Only used students who had variation in their evaluations were used when computing A' per student. In other words, a student was not considered if they were evaluated correct on all practice opportunities or incorrect on all practice opportunities. This was necessary because A' is undefined unless there is at least one 'positive', and at least one 'negative' evaluation for a student [19]. As a result, 175 students remained for designing controlled experiments and 132 students for testing stated hypotheses.

We ascertain whether any BKT model variant outperforms the classic model by comparing A' values computed under the cross-validation scheme described. These results are described next.

7.1 Models' Overall Predictive Capability

As shown in Table 1, all of the models show strong consistency, meaning that high estimates of skill demonstration are associated with actual demonstration of skill. This is evidenced by collapsed A' values ranging from .817 to .837 for the designing controlled experiments skill, and collapsed A' values ranging from .840 to .853 for the testing stated hypotheses skill. Recall that these high collapsed A' values do not reflect the models' ability to predict individual student trajectories [25], because they factor out the student term. The model with the highest A' = .837 for predicting future performance of the designing controlled experiments skill 1) conditioned the learning rate on whether the student received scaffolding (T_Scaffolded extension), and 2) incorporated skill degradation when switching between science topics (kLn_TopicSwitch extension). This represents a small increase in

Table 1. BKT model variant performance predicting whether students will demonstrate skill in their next practice attempt in the learning environment. The A' values were computed under six-fold student-level cross-validation Overall, the best model for both skills is the one in which the learning rate is conditioned on whether or not the student received scaffolding during Phase Change (*T_Scaffolded*).

BKT Model Variant			**Designing Controlled Experiments**		**Testing Stated Hypotheses**	
T_Scaffolded	T_Topic	kLn_TopicSwitch	A' per student avg[a]	A' collapsed[b]	A' per student avg[a]	A' collapsed[c]
X			.685	.827	.656	.846
	X		.633	.818	.610	.840
		X	.641	.825	.612	.844
X	X		.678	.829	.648	.848
	X	X	.630	.826	.601	.845
X		X	.680	.837	.638	.852
X	X	X	.676	.836	.645	.853
		Classic BKT:	.635	.817	.613	.841

[a] $N = 287$ students; [b] $N = 175$ students; [c] $N = 132$ students

performance over the classic BKT model (A' = .817). The model with the highest A' = .853 for predicting future performance of testing stated hypotheses was the full model that incorporated all three extensions. This again was a small improvement over the classic BKT model (A' = .841).

In terms of predicting individual student performance, some of the models performed reasonably well. As a baseline, the Classic BKT model for designing controlled experiments had a per-student average A' = .635. For testing stated hypotheses, the Classic BKT model had a per-student average A' = .613. These values, though above chance A' (.5), are somewhat low.

When incorporating some of the BKT variants, the per-student average A' increased. In particular, BKT variants that leveraged conditioning on scaffolding (T_Scaffolded model) performed better than the Classic BKT model (Table 1). For example, the best BKT model variant for both skills incorporated only scaffolding. The per-student average A' of this model for designing controlled experiments was .685, a jump over the Classic BKT model. The per-student average A' for testing stated hypotheses was .656, and again, outperformed the Classic BKT model. These A' values are on par with the extended BKT models developed in [6] that incorporated scaffolding.

7.2 Model Interpretation

Like [6], we interpreted the models' parameters to understand what they reveal about the impacts of scaffolding and the learning and transfer of scientific inquiry skills between Physical Science topics. Since the full models with 7 parameters had A' performance on par with the other best performing models, we chose to interpret their parameters. The parameter averages and standard deviations for each skill model across all six folds are presented in Table 2. We focus on interpreting the new parameters we added to the model.

In Phase Change, the learning rate when students were scaffolded is much higher than the learning rate without scaffolding, *T_Scaff_PhCh* = .638 vs. *T_UnScaff_PhCh* = .190 for designing controlled experiments, and *T_Scaff_PhCh* = .823 vs. *T_UnScaff_PhCh* = .158 for testing stated hypotheses. These values indicate that scaffolding students' inquiry appears to have a positive effect on whether students learn the skills [6].

The learning rate for the Free Fall activities, which were unscaffolded and practiced after the Phase Change activities, was comparatively lower for each skill, *T_UnScaff_FF* = .094 for designing controlled experiments, and *T_UnScaff_FF* = .089 for testing stated hypotheses. The meaning of these values is more difficult to discern because all students had prior opportunity to practice in Phase Change before attempting the Free Fall tasks. It could be that the unscaffolded Free Fall activities, like the unscaffolded Phase Change activities, are less effective for helping students acquire these inquiry skills. However, it could also be that the lower learning rates reflect that many students already mastered the skills in Phase Change and thus these new activities afforded no additional learning opportunities. We believe the latter to be the case because 1) more than 85% of students demonstrated each skill in their first Free Fall practice opportunity (data not presented in this paper), and 2) the initial likelihood of knowing the skills (L_0) was high.

Finally, the skill degradation parameter *k*, which captures the degree of skill transfer between science topic (0 is no transfer, 1 is full transfer), was high for both skills. For designing controlled experiments, *k* = .973 and for testing stated hypotheses, *k* = .961. These high values suggest that skill transfers from Phase Change to Free Fall within our learning environment [cf. 15]. We elaborate on this finding in more detail in the next section.

8. DISCUSSION AND CONCLUSIONS

In the classic Bayesian Knowledge Tracing framework [1], scaffolding and the tutor context, the nature of the activities in which skills are applied, are not taken into account when predicting students' future performance. Similar to others' prior work [6-8] we explored here whether extending the BKT framework to incorporate these factors improves prediction of students' skill demonstration. This work was conducted to predict students' acquisition of two data collection inquiry skills, designing controlled experiments and testing stated hypotheses [cf. 12, 13], in performance-based inquiry tasks across two Physical Science topics, Phase Change and Free Fall. Specifically, we added three extensions to the BKT model: 1) conditioning the learning rate on whether or not students were scaffolded; 2) conditioning the learning rate depending on the topic in which students practiced inquiry (Phase Change or Free Fall); and 3) adding a degradation parameter to potentially lower the likelihood

Table 2. Means and standard deviations of the parameter values for full BKT model variant across all six folds.

	Full BKT Model Parameters						
Skill	*L₀*	*G*	*S*	*T_UnScaff_PhCh*	*T_Scaff_PhCh*	*T_UnScaff_FF*	*k*
Designing Controlled Experiments	.470 (0.014)	.196 (0.029)	.050 (0.006)	.190 (0.018)	.638 (0.035)	.094 (0.010)	.973 (0.006)
Testing Stated Hypotheses	.602 (0.026)	.198 (0.023)	.042 (0.007)	.158 (0.026)	.823 (0.057)	.089 (0.011)	.961 (0.009)

of a student knowing a skill when the science topic changed. Overall, we found that BKT can track development of both skills, in accordance with our prior work [13], and that our extensions led to improvements in prediction and model interpretability.

In comparing our BKT extension that incorporates scaffolding, our approach is closest to the one taken in [6]. Our model assumes that scaffolding will *only* impact learning, whereas [6] captures that scaffolding may differentially impact learning *and* immediate performance. Our modeling choice was motivated in part by parsimony given that BKT is already overparametrized [24], a possibility hypothesized in [6], and by the delay between performance attempts of the skills in our learning environment. Unlike [6], we found that taking scaffolding into account improved the ability to predict individual student learning and performance over the classic BKT model, possibly due to increased parsimony. We also teased apart the effects of scaffolding on our models' predictive abilities overall (collapsing over students) and on predicting individual student performance.

When interpreting the parameters of the extended model, we found that scaffolding appears to have a positive impact on learning, as in [6]. We do note, though, that we did not tease out the differential impacts of specific scaffolds in our multi-level scaffolding approach. It is possible that specific scaffolds trigger different degrees of learning. One possible way to incorporate this is to condition learning rate on the different kinds of scaffolds, not just whether or not students received scaffolding in general.

We also incorporated parameters to account for the possible effects of demonstrating inquiry skill within different science topics (Phase Change and Free Fall). This modeling was inspired by the empirical question of whether inquiry skills are tied to the science topic in which they are learned [15], or if they transfer across topics [9, 10]. Though incorporating these parameters did not increase the predictive performance of our models, they do provide possible insights to inquiry learning. In particular, the model parameters suggest that the data collection skills of interest transfer across science topics, which supports earlier findings [e.g. 20, 26, 14]. There are limits to how certain we can be about this interpretation, though. First, in our study design, we only randomized whether students received scaffolding in Phase Change, and then measured transfer to Free Fall. A stronger approach to increase parameter interpretability would be to also randomize the science topic order. Second, it is possible that the implied transfer of skill may be due to the structural similarities of the activities [9] across Physical Science tasks. In the future, it will be beneficial to conduct a similar study across different science topic areas, like Life and Earth Science [12], with different activity structures to tease apart these possible effects.

This paper offers three contributions. First, to our knowledge, this work is the first application of BKT to track the development of inquiry process skills across science topics. This work strengthens our earlier findings in using BKT for a single group of students and single topic [13], because we cross-validated our models with students from multiple schools who engaged in two science topics. Second, we extended BKT by incorporating scaffolding. Though this extension is similar to others' [e.g. 6, 8], it enabled a "discovery with models" analysis [cf. 27] that shed light on the potential relationships between performance in the environment, scaffolding, and transfer of inquiry skills [15]. Furthermore, conditioning the BKT learning rate on whether students received scaffolding improved prediction of individual students' trajectories over the classic model. Finally, we incorporated a form of tutor context (the science topic in which skills were demonstrated) directly in the BKT model, unlike [8], which addressed context by selecting subsets of training and testing data. By adding these additional parameters, we discerned that the data collection skills transferred across the two science topics by interpreting the extended BKT model.

In closing, we note that this work focuses primarily on validation and interpretation of skill *within our learning environment.* In our prior work [13], we also showed that BKT models not only had this internal reliability, but were also moderately predictive of other measures of inquiry. In the future, we will determine if our model extensions can also improve external validation, thus realizing the full potential of using our learning environment to estimate and track authentic inquiry skills.

9. ACKNOWLEDGMENTS

This research is funded by the National Science Foundation (NSF-DRL#0733286, NSF-DRL#1008649, and NSF-DGE#0742503) and the U.S. Department of Education (R305A090170 and R305A120778). Any opinions expressed are those of the authors and do not necessarily reflect those of the funding agencies. Special thanks are also given to Joseph Beck for advice and feedback on this work.

10. REFERENCES

1 Corbett, A.T. and Anderson, J.R. Knowledge-Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction*, 4 (1995), 253-278.

2 Pardos, Z.A. and Heffernan, N.T. Modeling Individualization in a Bayesian Networks Implementation of Knowledge Tracing. In *Proceedings of the 18th International Conference on User Modeling, Adaptation and Personalization* (Big Island, HI 2010), 255-266.

3 Baker, R.S.J.d, Corbett, A.T., Gowda, S.M. et al. 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* (Big Island of Hawaii, HI 2010), 52-63.

4 Pardos, Z.A. and Heffernan, N.T. KT-IDEM: Introducing Item Difficulty to the Knowledge Tracing Model. In *Proceedings of*

the 19th International Conference on User Modeling, Adaptation and Personalization (Girona, Spain 2011), 243-254.

5 Koedinger, K.R., Pavlik, P.I., Stamper, J., Nixon, T., and Ritter, S. Avoiding Problem Selection Thrashing with Conjunctive Knowledge Tracing. In *Proceedings of the 3rd International Conference on Educational Data Mining (EDM 2010)* (Pittsburgh, PA 2010), 91-100.

6 Beck, J.E., Chang, K., Mostow, J., and Corbett, A. Does Help Help? Introducing the Bayesian Evaluation and Assessment Methodology. In *Proceedings of the 9th International Conference on Intelligent Tutoring Systems* (Montreal, QC 2008), 383-394.

7 Jonsson, A., Johns, J., Mehranian, H. et al. Evaluating the Feasbility of Learning Student Models from Data. In *AAAI05 Workshop on Educational Data Mining* (Pittsburgh, PA 2005), 1-6.

8 Yudelson, M., Medvedeva, O., and Crowley, R. A Multifactor Approach to Student Model Evaluation. *User Modeling and User-Adapted Interaction*, 18 (2008), 349-382.

9 Thorndike, E.L. and Woodworth, R.S. The Influence of Improvement in One Mental Function Upon the Efficacy of Other Functions. *Psychological Review*, 8 (1901), 247-261.

10 Singley, M. and Anderson, J.R. *The Transfer of Cognitive Skill*. Harvard University Press, Cambridge, MA, 1989.

11 NATIONAL RESEARCH COUNCIL. *National Science Education Standards*. National Science Education Standards, Washington, D.C., 1996.

12 Gobert, J., Sao Pedro, M., Baker, R., Toto, E., and Montalvo, O. Leveraging educational data mining for real time performance assessment of scientific inquiry skills within microworlds. *Journal of Educational Data Mining*, 4, 1 (2012), 111-143.

13 Sao Pedro, M.A., Baker, R.S.J.d., Gobert, J.D., Montalvo, O., and Nakama, A. Leveraging Machine-Learned Detectors of Systematic Inquiry Behavior to Estimate and Predict Transfer of Inquiry Skill. *User Modeling and User-Adapted Interaction*, 23 (2013), 1-39.

14 Sao Pedro, M., Gobert, J., and Baker, R. Assessing the Learning and Transfer of Data Collection Inquiry Skills Using Educational Data Mining on Students' Log Files. In *Paper presented at The Annual Meeting of the American Educational Research Association.* (Vancouver, BC, Canada 2012), Retrieved April 15, 2012, from the AERA Online Paper Repository.

15 van Joolingen, W.R., de Jong, T., and Dimitrakopoulout, A. Issues in Computer Supported Inquiry Learning in Science. *Journal of Computer Assisted Learning*, 23, 2 (2007), 111-119.

16 Sao Pedro, M.A., Baker, R.S.J.d., and Gobert, J.D.. In *Proceedings of the 3rd Conference on Learning Analytics and Knowledge* (Leuven, Belgium 2013), 190-194.

17 Sao Pedro, M., Baker, R., and Gobert, J. Improving Construct Validity Yields Better Models of Systematic Inquiry, Even with Less Information. In *Proceedings of the 20th Conference on User Modeling, Adaptation, and Personalization (UMAP 2012)* (Montreal, QC, Canada 2012), 249-260.

18 Baker, R.S.J.d., Corbett, A.T., and Wagner, A.Z. Human Classification of Low-Fidelity Replays of Student Actions. In *Proceedings of the Educational Data Mining Workshop held at the 8th International Conference on Intelligent Tutoring Systems* (Jhongli, Taiwan 2006), 29-36.

19 Hanley, J.A. and McNeil, B.J. The Meaning and Use of the Area under a Receiver Operating Characteristic (ROC) Curve. *Radiology*, 143 (1982), 29-36.

20 Kuhn, D., Schauble, L., and M., Garcia-Mila. Cross-Domain Development of Scientific Reasoning. *Cognition and Instruction*, 9 (1992), 285-327.

21 Chen, Z. and Klahr, D. All Other Things Being Equal: Acquisition and Transfer of the Control of Variables Strategy. *Child Development*, 70, 5 (1999), 1098-1120.

22 Koedinger, K., Suthers, D., and Forbus, K. Component-Based Construction of a Science Learning Space. *International Journal of Artificial Intelligence in Education (IJAIED)*, 10 (1998), 292-313.

23 Bransford, J.D. and Schwartz, D. L. Rethinking Transfer: A Simple Proposal with Multiple Implications. In *Review of Research in Education, 24*. American Educational Research Association, Washington, D.C., 1999.

24 Beck, J.E. and Chang, K. Identifiability: A Fundamental Problem of Student Modeling. In *Proceedings of the 11th International Conference on User Modeling* (Corfu, Greece 2007), 137-146.

25 Pardos, Z.A., Gowda, S.M., Baker, R.S.J.d., and Heffernan, N.T. The Sum is Greater than the Parts: Ensembling Models of Student Knowledge in Educational Software. *ACM SIGKDD Explorations*, 13, 2 (2012), 37-44.

26 Glaser, R., Schauble, L., Raghavan, K., and Zeitz, C. Scientific Reasoning Across Different Domains. In DeCorte, E. et al., eds., *Computer-based Learning Environments and Problem-Solving* (Heidelberg, Germany, 1992), 345-371.

27 Baker, R.S.J.d. and Yacef, K. The State of Educational Data Mining in 2009: A Review and Future Visions. *Journal of Educational Data Mining*, 1, 1 (2009), 3-17.

Applying Three Models of Learning to Individual Student Log Data

Brett van de Sande
Arizona State University
PO Box 878809
Tempe, AZ 85287
bvds@asu.edu

ABSTRACT

Normally, when considering a model of learning, one compares the model to some measure of learning that has been aggregated over students. What happens if one is interested in individual differences? For instance, different students may have received different help, or may have behaved differently. In that case, one is interested in comparing the model to the individual learner. In this study, we investigate three models of learning and compare them to student log data with the goal of seeing which model best describes individual student learning of a particular skill. The log data is from students who used the Andes intelligent tutor system for an entire semester of introductory physics. We discover that, in this context, the ”best fitting model” is not necessarily the ”correct model” in the usual sense.

Keywords

data mining, models of student learning

1. INTRODUCTION

Most Knowledge Component (KC) [15] based models of learning are constructed in a similar manner, following Corbett and Anderson [8]. First, some measure of learning is selected (*e.g.* correct/incorrect on first try) for the j-th opportunity for that student to apply a given KC. This measure of learning is then aggregated over students (*e.g.* fraction of students correct) as a function of j. Finally, aggregated measure is then compared to some model (*e.g.* Bayesian Knowledge Tracing) with model parameters chosen to optimize the model's fit to the data. In principle, given sufficient student log data, one could uniquely determine which of several competing models best matches the data.

One drawback with this approach is that it does not take into account individual learner differences or the actual behaviors of students or tutors as they are learning. Thus, a number of authors have extended their models to include individual student proficiency and actual help received by the student. For instance, in the Cordillera natural language tutoring system for physics [16], the student may have been asked what the next step was or were told what the next stop was; this was used as input for an associated model. An overview of these models can be found in [7].

If one is primarily interested in the effectiveness of help given to an individual student or the effectiveness (for learning) of a particular strategy or behavior of a student, then it may make sense to fit a model of learning to the log data of each student individually. Given sufficient student log data, can we still talk about a particular model fitting the student log data well? That is the central question of this paper. To start our investigation, we will compare three different models of learning using data from students taking introductory physics and examine whether there is empirical support for using one model over the others. In fact, using Akaike Information Criteria (AIC), we obtain results that seem to favor two models over the third, but note that fitting the models to individual students can make the determination ambiguous.

1.1 Correct/Incorrect steps

Our stated goal is to determine student learning for an individual student as they progress through a course. What observable quantities should be used to determine student mastery? One possible observable is “correct/incorrect steps,” whether the student correctly applies a given skill at a particular problem-solving step without any preceding errors or hints. There are other observables that may give us clues on mastery: for instance, how much time a student takes to complete a step that involves a given skill. However, other such observables typically need some additional theoretical interpretation. *Exempli gratia*, What is the relation between time taken and mastery? Baker, Goldstein, and Heffernan [3] develop a model of learning based on a Hidden Markov model approach. They start with a set of 25 additional observables (including “time to complete a step”) and construct their model and use correct/incorrect steps to calibrate the additional observables and determine which are significant. Naturally, it is desirable to eventually include various other observables in any determination of student learning. However, in the present investigation, we will focus on correct/incorrect steps.

Next, we need to define precisely what we mean by a step. A student attempts some number of *steps* when solving a problem using an intelligent tutor system (ITS). Usually, a step

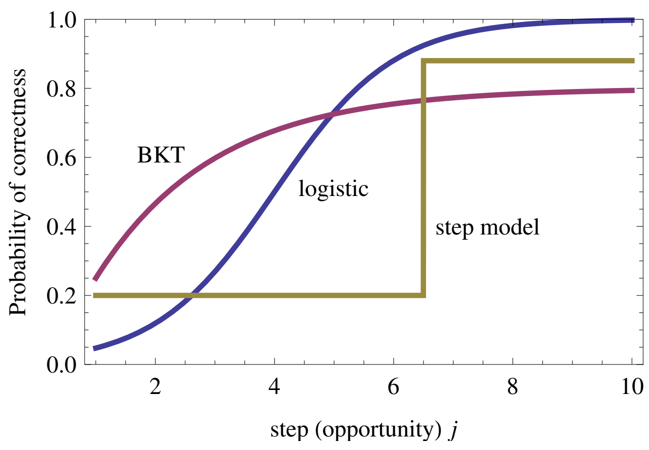


Figure 1: Functional form of the three models of student learning.

is associated with creating/modifying a single user interface object (writing an equation, drawing a vector, defining a quantity, *et cetera*) and is a distinct part of the problem solution (that is, help-giving dialogs are not considered to be steps). A student may attempt a particular problem-solving step, delete the object, and later attempt that solution step again. A step is an *opportunity* to learn a given Knowledge Component (KC) [15] if the student must apply that skill to complete the step.

Andes is a model-tracing tutor [2], which means that the ITS contains a number of "model solutions" to each problem and each step of the model solution has one or more KCs assigned to it. As a student solves a problem, Andes tries to match each student attempt at a step to a corresponding model solution step and, if that match is successful, assigns the corresponding KCs to that step attempt. For some common errors, Andes has a number of error detectors that infer what solution step the student was attempting to work on. In that case, KCs can assigned to that attempt. However, there are many errors where the associated KCs cannot be determined. In the log analysis, if a step attempt does not have any KCs assigned to it, we use the following heuristic to determine the associated KCs: First, we look at any subsequent attempts associated with the same user interface element and see if they have any KCs associated with them. If that fails, then we look for the next attempt having the same *type* of user interface element (equation, vector, *et cetera*) that has some KCs associated with it.

For each KC and student, we select all attempted steps that involve application of that KC and mark each step as "correct" if the student completes that step correctly without any preceding errors or requests for help; otherwise, we mark the step as "incorrect." If each incorrect/correct step is marked with a 0/1, then a single student's performance on a single KC can be expressed as a bit sequence, *exempli gratia* 00101011. We will label steps with $j \in \{1, \ldots, n\}$.

2. THREE MODELS OF LEARNING

Ultimately, we are interested in determining when a student has mastered a particular KC and, by inference, the effectiveness of any help given by the tutor. Thus, a useful model of learning should have the the following properties:

1. Be compatible with actual student behavior. That is, its functional form should fit well with student data. We will explore this question in Section 3.
2. Give the probability that learning has occurred at a given step.
3. Assuming learning has occurred at a given step, the model should give a prediction for the associated increase in performance and the rate of errors after learning.

We will consider three candidate models: the Bayesian Knowledge Tracing (BKT) model, the logistic function, and the "step model;" see Fig. 1.

The first model is the Bayesian Knowledge Tracing (BKT) model [8]. The hidden Markov model form of BKT is often fit to student performance data [4]. One can show that this model, in functional form, is an exponential function with three model parameters [13]:

$$P_{\mathrm{BKT}}(j) = 1 - P(S) - Ae^{-\beta j} \,. \tag{1}$$

One central assumption of BKT is that, given that learning has not already occurred, mastery is *equally probable* on each step. This assumption of equal probability does not match well with our goal of determining empirically the steps where learning has actually occurred for an individual student, criterion 2. On the other hand, this model does provide the final error rate $P(S)$ (the initial error rate is ambiguous), so criterion 3 is partially satisfied.

A number of models of learning based on logistic regression have been studied [6, 10, 7]. These models involve fitting data for multiple students and multiple KCs and may involve other observables such as the number of prior successes/failures a student has had for a given skill. However, in this investigation, we are interested in fitting to the correct/incorrect bit sequence for a single student and a single KC and a logistic regression model takes on a relatively simple form

$$\log\left(\frac{P_{\mathrm{logistic}}(j)}{1-P_{\mathrm{logistic}}(j)}\right) = b(j-L) \tag{2}$$

which can be written as:

$$P_{\mathrm{logistic}}(j) = \frac{1}{1+\exp\left(-b(j-L)\right)} \,. \tag{3}$$

It is natural to associate L with the moment of learning. However, the finite slope of $P_{\mathrm{logistic}}(j)$ means that learning may occur in a range of roughly $1/b$ steps before and after L. For $P_{\mathrm{logistic}}(j)$, the gain in performance is always 1 and the final error rate is always 0. Thus, although this model makes a prediction for when the skill is learned, criterion 2, it does not predict a gain in performance, criterion 3.

The third model is the "step model" which assumes that learning occurs all at once; this corresponds to the "eureka learning" discussed by [3]. It is defined as:

$$P_{\mathrm{step}}(j) = \left\{ \begin{array}{cc} g, & j < L \\ 1-s, & j \geq L \end{array} \right. \tag{4}$$

where L is the step where the student first shows mastery of the KC, g is the "guess rate," the probability that the student gets a step correct by accident, and s is the "slip rate," the chance that the student makes an error after learning the skill. These are analogous to the guess and slip parameters of BKT [8]. The associated gain in performance is $1 - g - s$ and the error rate after learning is simply s in this model. Thus, this model satisfies criteria 2 and 3.

3. MODEL SELECTION USING AIC

The BKT and logistic function models are widely used and we have introduced the step model $P_{\text{step}}(j)$ as an alternative. How well do these models match actual student behavior? Since we will use the step model in subsequent work, it would be reassuring to know whether it describes the student data as well (or better than) the other two models. We will use the Akaike Information Criterion (AIC) for this purpose [1, 5]. AIC is defined as

$$\text{AIC} = -2\log(\mathcal{L}) + 2K \tag{5}$$

where $\mathcal{L}$ is the maximized value of the likelihood function and K is the number of parameters in the model. AIC is an estimate of the expected relative "distance" between a given model and the true model (assumed to be complicated) that actually generated the observed data. It is valid in limit of many data points, $n \to \infty$, with leading corrections of order $1/n$.

A related method for choosing between models is the Bayesian Information Criterion (BIC) introduced by Schwarz [11]. BIC is defined as

$$\text{AIC} = -2\log(\mathcal{L}) + K\log(n) \tag{6}$$

where n is the number of data points. Burnham & Anderson [5, Sections 6.3 & 6.4] explain that BIC is more appropriate in cases where the "true" model that actually created the data is relatively simple (few parameters). If the true model is contained in the set of models being considered, then BIC will correctly identify the true model in the $n \to \infty$ limit. For BIC to have this property, the true model must stay fixed as n increases. The authors argue that, while BIC may be appropriate in some of the physical sciences and engineering, in the biological and social sciences, medicine, and other "noisy" sciences, the assumptions that underlie BIC are generally not met. In particular, as the sample size increases, it is typical that the underlying "true" model also becomes more complicated. This is certainly true in educational datamining: datasets are generally increased by adding data from new schools, or different years and one generally expects noticeable variation of student behavior from school to school or from year to year. In such cases, one safely can say that the "true" model is complicated (because people are complicated) and becomes more complicated as a dataset is increased in size. Although most authors quote both AIC and BIC values, there is good reason to believe that AIC is generally more appropriate for educational datamining work.

3.1 Method

We examined log data from 12 students taking an intensive introductory physics course at St. Anselm College during summer 2011. The course covered the same content as

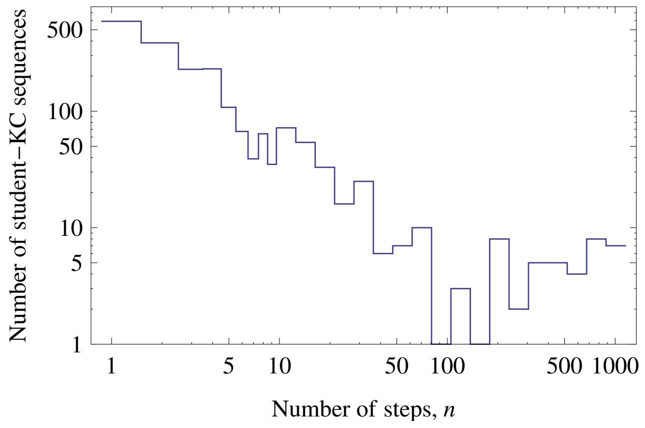


Figure 2: Histogram of number of distinct student-KC sequences in student dataset $\mathcal{A}$ having a given number of steps n.

a normal two-semester introductory course. Log data was recorded as students solved homework problems while using the Andes intelligent tutor homework system [17]. 231 hours of log data were recorded. Each student problem-solving step is assigned one or more KCs using the heuristic described in Section 1.1. The dataset contains a total of 2017 distinct student-KC sequences covering a total of 245 distinct KCs. We will refer to this dataset as student dataset $\mathcal{A}$. See Figure 2 for a histogram of the number of student-KC sequences having a given number of steps.

Most KCs are associated with physics or relevant math skills while others are associated with Andes conventions or user-interface actions (such as, notation for defining a variable). The student-KC sequences with the largest number of steps are associated with user-interface related skills, since these skills are exercised throughout the entire course.

One of the most remarkable properties of the distribution in Fig. 2 is the large number of student-KC sequences containing just a few steps. The presence of many student-KC sequences with just one or two steps may indicate that the default cognitive model associated with this tutor system may be sub-optimal; to date, there has not been any attempt to improve on the cognitive model of Andes with, say, Learning Factors Analysis [6]. Another contributing factor is the way that introductory physics is taught in most institutions, with relatively little repetition of similar problems. This is quite different than, for instance, a typical middle school math curriculum where there are a large number of similar problems in a homework assignment.

3.2 Analysis

Since the goodness of fit criterion, AIC, is valid in the limit of many steps, we include in this analysis only student-KC sequences that contain 10 or more steps, reducing the number of student-KC sequences to 267, covering 38 distinct KCs. We determine the correctness of each step (Section 1.1), constructing a bit sequence, *exempli gratia* 001001101, for each student-KC sequence. This bit sequence is then fit to each of the three models, P_{step}, P_{logistic}, and P_{BKT} by maximiz-

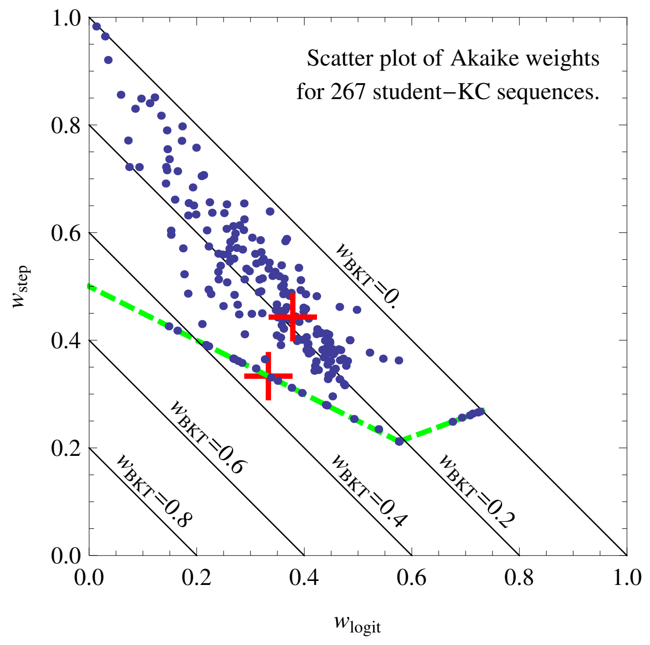


Figure 3: Scatter plot of Akaike weights for the three models, $P_{\rm step}$, $P_{\rm logistic}$, and $P_{\rm BKT}$, when fit to student-KC sequences from an introductory physics course. The point where all models are equal, $w_{\rm step} = w_{\rm logistic} = w_{\rm BKT} = 1/3$, is marked with the lower cross. The average of the weights is marked with the upper cross. The dashed line on the left represents points where $w_{\rm step} = w_{\rm BKT}$. Finally, the dashed line on the right marks data with bit sequences of the form $00\cdots011\cdots1$.

ing the associated log likelihood. For $P_{\rm logistic}$, and $P_{\rm BKT}$, the fits were calculated using the Differential Evolution algorithm [12] provided by *Mathematica*. For $P_{\rm step}$, the best fit, as a function of s and g, can be found analytically; one can then find the best fit, as a function of L, by conducting an exhaustive search. Next, we calculate the AIC score for each fit. Finally, we calculate the Akaike weights, $w_{\rm logistic}$, $w_{\rm step}$, and $w_{\rm BKT}$ for each student-KC sequence [5]. The weights are normalized so that

$$1 = w_{\rm logistic} + w_{\rm step} + w_{\rm BKT} \ . \qquad (7)$$

The Akaike weight represents the relative probability that a particular model in a given set of models is closest to the model that has actually generated the data.

A scatter plot of the weights is shown in Fig. 3. If all three models described the data equally well, then we would expect points to be scattered evenly about the center point $w_{\rm logistic} = w_{\rm step} = w_{\rm BKT} = 1/3$. Instead, we see the step model (average weight 0.44) weakly favored over the logistic model (average weight 0.37) and strongly favored over BKT (average weight 0.18). Indeed, we find no data points where $w_{\rm step} < w_{\rm BKT}$, although there is a noticeable accumulation of points along the line $w_{\rm step} = w_{\rm BKT}$.

Note that data in the form of incorrect steps then correct steps, *exempli gratia* $00\cdots011\cdots1$, is fit perfectly by both the $P_{\rm step}$ and $P_{\rm logistic}$ models. In this case, since $P_{\rm logistic}$ has one fewer parameter than $P_{\rm step}$, it is favored by AIC by a constant factor and $w_{\rm step} = {\rm e}^{-1} w_{\rm logistic}$. This case is plotted as the increasing dashed line in Fig. 3.

Since the student-KC sequences contain an average of about $n = 16$ steps, it is surprising that we find that AIC so strongly discriminates between the models. Perhaps, though, this is due to some finite n correction: recall that AIC is only strictly valid in the $n \to \infty$ limit.

3.3 Random data

To further investigate the observed strong discrimination between the three models, we constructed an artificial dataset containing random bit sequences (each step has 50% probability of being "correct") of length $n \in \{10, 20, 30, 40, 50\}$, with 10,000 sequences for each n. This dataset corresponds to a model of the form

$$P_{\rm random}(j) = 1/2 \ . \qquad (8)$$

We then repeated our analysis of the three models using this dataset and AIC as our selection criterion. Note that all three models, with a suitable choice of parameters, can be made equal to $P_{\rm random}$ itself.

As mentioned earlier, for data that is generated by a simple model (and $P_{\rm random}$ is about as simple as one can get) and the "true" model is included among the set of models, BIC is the more appropriate criterion for model selection [5, Sections 6.3 & 6.4]. However, for our results, the only difference between AIC and BIC is that BIC favors $P_{\rm logistic}$ more strongly over $P_{\rm step}$ and $P_{\rm BKT}$. Thus, using BIC would shift the weights so that $w_{\rm logistic}$ would increase somewhat over the other two weights. However, in order to maintain consistency with our experimental results, Fig. 3, we used AIC for the random data as well; see Fig. 4. This use of AIC versus BIC does not affect our conclusions.

For data generated by $P_{\rm random}$, one expects that all three models should perform equally well since all three can equal (with suitable choice of parameters) the known correct model $P_{\rm random}$. Thus, we would expect a scatter plot of the Akaike weights to center around $w_{\rm logistic} = w_{\rm step} = w_{\rm BKT} = 1/3$. Instead, we find that $P_{\rm step}$ is still highly favored over the other two; see Fig. 4. This bias seems to persist as we increase n.

Since we know that AIC (or BIC) is only strictly valid in the asymptotic limit $n \to \infty$, it is useful to see if the large differences persist as n is increased. If we average over the 10,000 weights and plot the average weight as a function of n, we find that the differences between the weights persist in the $n \to \infty$ limit; see Fig. 5. If we fit the average weights to a constant plus $1/n$; the fits are:

$$\begin{aligned} \langle w_{\rm step} \rangle &= 0.58 - \frac{1.50}{n} & (9) \\ \langle w_{\rm logistic} \rangle &= 0.24 + \frac{1.2}{n} & (10) \\ \langle w_{\rm BKT} \rangle &= 0.17 + \frac{0.30}{n} \ . & (11) \end{aligned}$$

This shows that AIC, in the asymptotic limit $n \to \infty$, still favors $P_{\rm step}$ over the other two models when used to evaluate

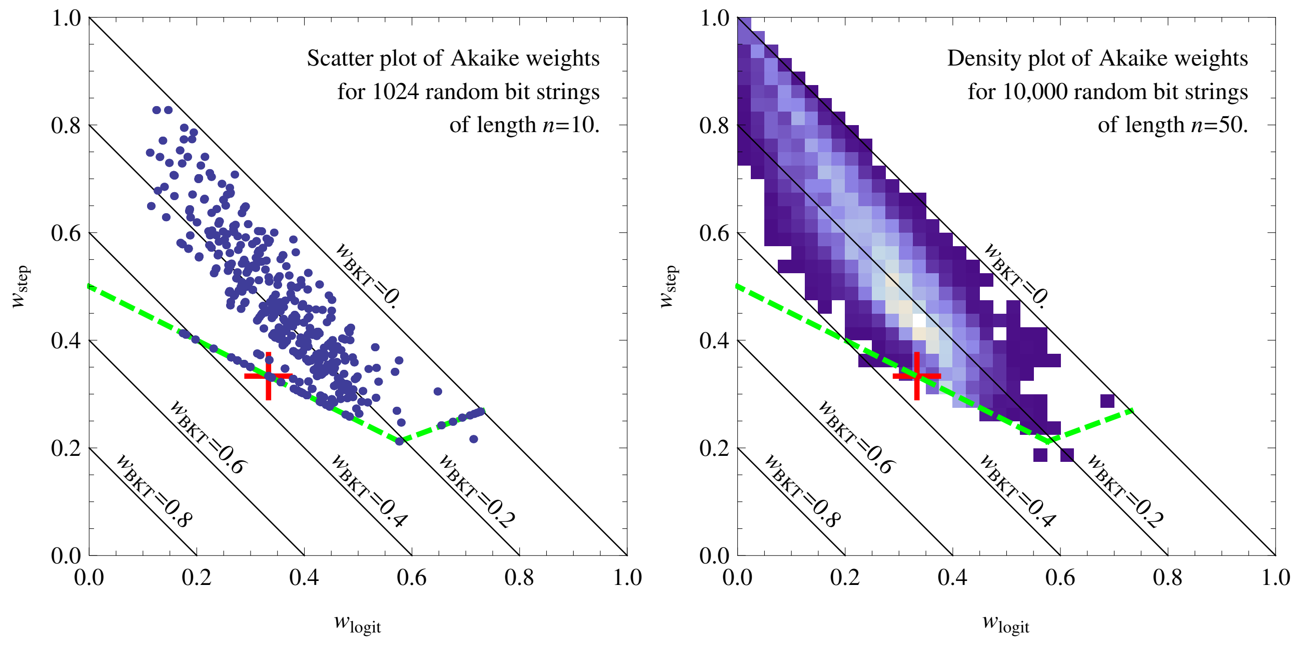


Figure 4: Akaike weights for the three models, $P_{\rm step}$, $P_{\rm logistic}$, and $P_{\rm BKT}$, when fit to randomly generated data. The point where $w_{\rm step} = w_{\rm logistic} = w_{\rm BKT} = 1/3$ is marked with a cross. For these datasets, each model should perform equally well, since, with an appropriate choice of parameters, they all can be made equal to the model that was used to generate the data.

randomly generated data.

If we repeat this analysis with BIC, we would still find that the weights converge to a constant value with $1/n$ leading errors. The only difference is that the logistic model has a larger weight than the other two. The differences between the weights of the three models still persist in the $n \to \infty$ limit.

3.4 Conclusions

In conclusion, we obtain some surprising results when we compare the three models, $P_{\rm step}$, $P_{\rm logistic}$, and $P_{\rm BKT}$, using individual student data. We see that AIC weakly favors the step model over the logistic model in a fashion that one might expect. However, in an unexpected fashion, we see that both are strongly favored over the BKT model. We see that this effect persists for randomly generated data and is not due to an insufficient number of opportunities (finite n effect).

Moreover, for any bit sequence, $P_{\rm BKT}$ *never* fits the data better than $P_{\rm step}$. Since both models have three parameters, this result holds for any maximum likelihood-based criterion, including both AIC and BIC. We don't have an analytic proof for this result, but the numerical evidence (see Fig. 5) is quite strong. In other words, even if one uses $P_{\rm BKT}$ (for some set of model parameters) to generate a bit sequence, one can always adjust the parameters in $P_{\rm step}$ so that it fits the bit sequence as well as, or better than, $P_{\rm BKT}$.

What does this mean? Let us think more carefully about maximum likelihood. If one uses a model to generate a single bit sequence, we cannot determine the exact probability function (the probability as a function of j) that generated it. At best, one can only talk about the probability that given a function *may* have generated that sequence. On the other hand, if one uses a particular probability function to generate a collection of infinitely many sequences, then we know the exact probability for each step. Therefore, given the collection of many sequences, one can uniquely determine the probability function that generated that collection. If that function comes from a particular model $\mathcal{A}$ (for some choice of model parameters), then we can safely conclude that model $\mathcal{A}$ is the correct model.

In other words, when we fit individual student data to a model (fitting model parameters separately for each student), then we can make no statements about what model is "correct" in the sense that it may have generated the data. We can only talk about a model being a good fit in the sense that it is "close" to the data. On the other hand, if we aggregate data from many students and fit to a model (finding the best fit model parameters), then we can talk about a model being correct in the usual sense that it may have generated the data.

If we are interested in determining the effectiveness of help given or of a particular student behavior, we are more concerned about being "close" to the student data than finding the correct theory of learning, so the fact that the step model fits the data better than the logistic function and the BKT model is of practical value when analyzing student log data. However, one should not then conclude that the step function is a better model of student learning, in the usual sense. The better fit does not predict anything about the nature of learning.

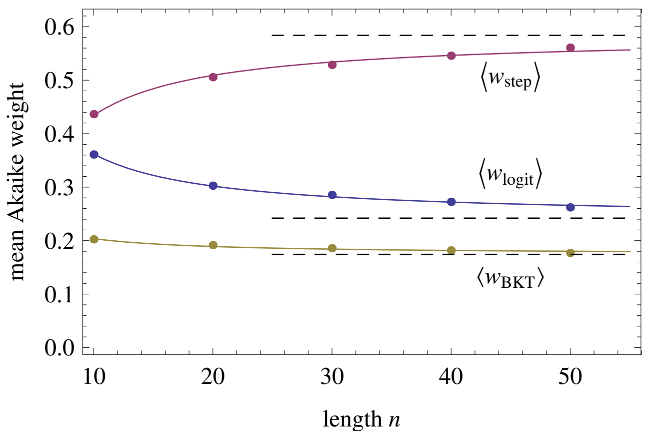


Figure 5: Mean Akaike weights for the three models, P_{step}, P_{logistic}, and P_{BKT}, when fit to randomly generated data of length n. (Each mean is calculated by averaging over 10,000 random bit sequences.) Also shown is a fit to a function of the form $a + b/n$ and a dashed line marking the asymptotic value a. Note that the large differences between the weights persist in the $n \to \infty$ limit.

Our results suggest that the step model may be useful for modeling the learning of an individual student. However, the step model assumes that learning a skill occurs in a single step. Is this how people actually learn? Certainly, everyone has experienced "eureka learning" at some point in their lives. However, it is unclear how well this describes the acquisition of other skills, especially since many KCs are implicit and people are not consciously aware that they even know them [9]. Certainly, if the student performance bit sequence is of the form $00\ldots011\ldots1$, it seems safe to assume that learning occurred all in one step, corresponding to the first 1 in the sequence. However, it is possible that the transition from unmastered to mastery occurs over some number of opportunities and the bit sequence of steps takes on a more complicated form. In a companion paper [14], we introduce a method (based on AIC) that can describe gradual mastery, even though the step model itself assumes all-at-once learning. In that approach, for a given bit sequence, one speaks about the *probability* that learning occurred at a particular step.

Finally, we see that the scatter plot of Akaike weights for student data is remarkably similar to the scatter plots for the random model. This suggests that the student data has a high degree of randomness, and, in general, that study of the random model may be quite useful for better understanding the student data.

4. ACKNOWLEDGMENTS

Funding for this research was provided by the Pittsburgh Science of Learning Center which is funded by the National Science Foundation award No. SBE-0836012.

5. REFERENCES

[1] H. Akaike. A new look at the statistical model identification. *IEEE Trans. Autom. Control*, 19(6):716–723, Dec. 1974.

[2] J. R. Anderson and R. Pelletier. A development system for model-tracing tutors. In L. Birnbaum, editor, *Proceedings of the International Conference of the Learning Sciences*, pages 1–8, Evanston, IL, 1991.

[3] R. S. J. D. Baker, A. B. Goldstein, and N. T. Heffernan. Detecting learning moment-by-moment. *Int. J. Artif. Intell. Ed.*, 21(1-2):5–25, Jan. 2011.

[4] J. Beck and K.-m. Chang. Identifiability: A fundamental problem of student modeling. In C. Conati, K. McCoy, and G. Paliouras, editors, *User Modeling 2007*, volume 4511 of *Lecture Notes in Computer Science*, pages 137–146. Springer Berlin / Heidelberg, 2007.

[5] K. P. Burnham and D. R. Anderson. *Model Selection and Multi-Model Inference: A Practical Information-Theoretic Approach*. Springer, July 2002.

[6] H. Cen, Kenneth Koedinger, and B. Junker. Learning factors analysis - a general method for cognitive model evaluation and improvement. In *Proceedings of the 8th international conference on Intelligent Tutoring Systems*, pages 164–175, Jhongli, Taiwan, June 2006. Springer-Verlag Berlin, Heidelberg.

[7] M. Chi, K. Koedinger, G. Gordon, P. Jordan, and K. VanLehn. Instructional factors analysis: A cognitive model for multiple instructional interventions. In *Proceedings of the 4th International Conference on Educational Data Mining*, Eindhoven, the Netherlands, June 2011.

[8] A. T. Corbett and J. R. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Model. User-Adapt. Interact.*, 4(4):253–278, 1995.

[9] 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 Sci.*, 36(5):757âĂŞ798, 2012.

[10] P. I. Pavlik, H. Cen, and K. R. Koedinger. Performance factors analysis âĂŞA 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*, page 531âĂŞ538, Amsterdam, The Netherlands, The Netherlands, 2009. IOS Press.

[11] G. Schwarz. Estimating the dimension of a model. *Ann. Stat.*, 6(2):461–464, Mar. 1978.

[12] R. Storn and K. Price. Differential evolution &ndash; a simple and efficient heuristic for global optimization over continuous spaces. *J. Global Optim.*, 11(4):341–359, Dec. 1997.

[13] B. van de Sande. Properties of the bayesian knowledge tracing model. Under review, 2012.

[14] B. van de Sande. Measuring the moment of learning with an information-theoretic approach. Under review, 2013.

[15] K. VanLehn. The behavior of tutoring systems. *Int. J. Artif. Intell. Ed.*, 16(3):227–265, Jan. 2006.

[16] K. Vanlehn, P. Jordan, and D. Litman. Developing pedagogically effective tutorial dialogue tactics: Experiments and a testbed. In *Proceedings of SLaTE Workshop on Speech and Language Technology in Education*, pages 17–20, Farmington, Pennsylvania USA, Oct. 2007.

[17] K. Vanlehn, C. Lynch, K. Schulze, J. A. Shapiro, R. Shelby, L. Taylor, D. Treacy, A. Weinstein, and M. Wintersgill. The andes physics tutoring system: Lessons learned. *Int. J. Artif. Intell. Ed.*, 15(3):147–204, Aug. 2005.

Evaluating Topic-Word Review Analysis for Understanding Student Peer Review Performance

Wenting Xiong
University of Pittsburgh
Pittsburgh, PA, 15260
wex12@cs.pitt.edu

Diane Litman
University of Pittsburgh
Pittsburgh, PA, 15260
litman@cs.pitt.edu

ABSTRACT
Topic modeling is widely used for content analysis of textual documents. While the mined topic terms are considered as a semantic abstraction of the original text, few people evaluate the accuracy of humans' interpretation of them in the context of an application based on the topic terms. Previously, we proposed RevExplore, an interactive peer-review analytic tool that supports teachers in making sense of large volumes of student peer reviews. To better evaluate the functionality of RevExplore, in this paper we take a closer look at its Natural Language Processing component which automatically compares two groups of reviews at the topic-word level. We employ a user study to evaluate our topic extraction method, as well as the topic-word analysis approach in the context of educational peer-review analysis. Our results show that the proposed method is better than a baseline in terms of capturing student reviewing/writing performance. While users generally identify student writing/reviewing performance correctly, participants who have prior teaching or peer-review experience tend to have better performance on our review exploration tasks, as well as higher satisfaction towards the proposed review analysis approach.

Keywords
Educational peer reviews, text analysis, topic modeling, user study

1. INTRODUCTION
Peer review is a popular educational approach for helping students improve their writing performance. It provides different perspectives and valuable feedback on what is compelling and what is problematic. Ideally, from analyzing student peer reviews, instructors may not only learn about student writing issues by reading student feedback, but may also evaluate student reviewing performance by checking if comments are given for important issues in a good manner. However, due to the large amount of reviews, teachers seldom read the comments carefully if at all. Instructors whom we have interviewed have complained that peer reviews are time consuming to read and difficult to interpret. Interpreting them requires synthesizing opinions from multiple parties while making comparisons and contrasts across multiple students at the same time.

Nowadays, some existing web-based peer-review systems can help teachers set up peer review assignments and even grade student papers based on peer ratings, though no software yet has the intelligence to support teachers' comprehension of the textual review comments. Previously [14], we took our first step to address this issue and designed an interactive analytic interface (RevExplore) on top of SWoRD [4], a web-based peer-review reciprocal system that has been used by over 12,000 students over the last 8 years. Before deploying RevExplore as a SWoRD plugin to the public, we would like to evaluate its functionality carefully, especially its natural language processing (NLP) component that automatically abstracts and compares review content at the topic-word level.

For this purpose, we carry out a user study to examine the idea of analyzing peer reviews by comparing them in groups based on their topic words. In particular, we investigate the analytic power of topic words in the context of assessing student writing/review performance by mining peer reviews. In this study, we not only show that our proposed topic-word extraction method can better enable users to identify student writing/reviewing issues than a baseline, but also demonstrate that the utility of our topic-word approach depends on various factors.

2. RELATED WORK
There is increasing interest in research on computer-supported peer reviews both from the students' perspective for improving learning and from the teachers' perspective for informing decision making. From the students' perspective, prior studies of automatically assessing student peer-review performance either focus on detecting important feedback features [3, 15], or aim to assess the overall peer-review helpfulness [13]. From the teachers' perspective, Goldin and Ashley [6] use Bayesian networks to model computer-supported peer review which yields pedagogically useful information about student learning and about grading schema. In contrast with their work, we are interested in the educational contents (textual peer reviews) rather than the interaction between students during the peer review activities. Further-

more, our RevExplore involves humans in the loop: it allows teachers to interactively explore peer-review data at the student level first, and then drill down to particular groups of students for automated analysis of their peer reviews afterwards.

With respect to data mining of educational textual contents, the general goal is to summarize or analyze the textual contents to provide feedback to teachers, either about student learning activities, or about the utility of teaching materials. For understanding student learning activities, word-based content analysis, where the words are either learned through topic modeling or crafted manually, has been widely used for categorizing educational data such as online discussion threads [10, 9] and student-tutor interactions [7]. As peer review provides students learning opportunities during both paper writing and reviewing, the textual reviews are valuable in reflecting both student writing and reviewing performance. Therefore we hypothesize that extending data mining techniques to textual peer reviews can provide useful feedback to instructors regarding both student *writing* and *reviewing* performance.

Several NLP techniques can be used for word-based content analysis. One is the frequency-based method, which considers the content of a target corpus in terms of its most frequent words. A famous application of this method is to generate word-clouds, which is a popular web2.0 tool for supporting impression formation over textural data. For example, it has been used to compare political speeches from different people[1] In our study, we consider this method as a baseline (denoted as *Freq*) for evaluating our proposed topic-word extraction method, which is to automatically learn the salient words of a target corpus through a topic-signature approach to topic modeling (denoted as *TopicS*). Topic signature modeling assumes a single topic of the target corpus when comparing it against a background corpus. And this topic can be represented as a set of words based on statistical analysis of the word distribution in both corpora [8]. Another kind of topic modeling is based on graphical models, such as LDA [1]. LDA considers each document as a mixture over an underlying set of topic probabilities. While it has been widely studied for many NLP tasks from sentiment analysis to text summarization, we did not employ it in RevExplore for several reasons. First, the learned topic model changes with parameter settings (the number of topics and the hyperparameters) which are quite task dependent. Furthermore, the learned topics are generally difficult to interpret [16], and hard to evaluate. Although various automatic metrics were proposed, they do not always agree with human judgements in end-applications [2].

3. TOPIC WORDS IN REVEXPLORE

RevExplore [14] utilizes data visualization in combination with NLP techniques to help instructors interactively make sense of peer review data, which was almost impractical before. It has a student performance overview and a review comparison detail-view. In the overview, RevExplore visualizes the overall peer-review information at the student level, which allows instructors to effectively identify points of interest during their initial data exploration. In the detail-view, RevExplore automatically abstracts the semantic information of peer reviews at the topic-word level with the original texts visible on demand. To create the detail-view, we adapt existing natural language processing techniques to the peer-review domain for supporting automated analytics.

3.1 Preprocessing – domain word masking

Because peer reviews frequently refer to the content of the papers that they comment on, it is necessary to reduce the influence of such "paper topic" words on the extraction of "review topic" words from the peer reviews, otherwise the "paper topic" will dominate the computation of "review topic" words. Therefore, as a preprocessing step, we first compute the "paper topic" words of the writing assignment using TopicS[2], a java implementation of the topic signature acquisition algorithm [8]. TopicS computes the topic words from a topic relevant (target) corpus against a topic irrelevant (background) corpus based on word distribution using chi-square statistics (which will be explained later in this section). For computing the "paper topic" words, we use all student papers as the target corpus and use 5000 documents from the English Gigaword Corpus as the background corpus (the default setting of TopicS). Based on our intuition, we set the chi-square cutoff to be 10 ($p = .0016$), yielding about 500 topic words. As these words depend on the domain of the writing assignment, we denote them as domain words for the rest of the paper.

To prevent analysts from being distracted too much by domain words when analyzing peer reviews, before computing the "review topic" words using any extraction method, we apply domain word masking to each peer review by replacing all occurrences of each domain word (e.g. "war", "african", "americans", "women", "democracy", "rights", "states") with a dummy term "domainwords".

3.2 Comparison-oriented topic signatures

The topic signature algorithm [8] assumes that a target corpus has a single topic, and it computes the topic words for the target corpus with respect to a general background corpus. For each word, the algorithm computes a *likelihood ratio* [5] which tests the hypothesis of the word being a topic word of the target corpus versus the hypothesis that the word is not a topic word. The -2 log likelihood ratio has a chi-square distribution, which allows us to test the significance of each word to the topic of the target corpus when compared against the background corpus. In our work, we use the existing software TopicS (as mentioned before) for extracting topic signatures.

In RevExplore overview, when two groups of students are chosen for further review comparison in the detail-view, there is an implicit assumption of a topic difference between their corresponding review groups. Furthermore, the topic to be mined changes dynamically in accordance with the change of the analytic goals, which are specified through different grouping of reviews. To capture these assumptions when using TopicS to extract the topic words for a particular review group, we take its reviews as the target corpus and use all of

[1]http://www.tagcrowd.com/blog/2011/03/05/state-of-the-union-2002-vs-2011/

[2]TopicS was developed by Anni Louis for evaluating automated text summarization [12].

the reviews as the background corpus. In this way, we tailor the computation of the topic words to the desired analytic property of the target review group. We denote this adapted method as *TopicS*. For *TopicS*, we set the significance cutoff as 6.635, corresponding to a p value of .01. For a typical review group of our study, the number of the extracted topic words is about 20.

In our user study, we compare *TopicS* with the frequency-based method (*Freq*), and we expect that our *TopicS* can outperform *Freq* in helping users achieve better task performance. Note that both methods are performed after the domain-word masking.

4. DATA

Our peer-review corpus consists of 1405 free-text review comments and 24 student papers, which were collected in a college level history class [11]. The peer review was done through SWoRD [4], a web-based peer-review reciprocal system, as follows:

Assignment creation: The teacher first created the writing assignment in SWoRD and provided a peer-review rubric which required students to assess a paper's quality on three separate dimensions (Logic, Flow and Insight), by giving a numeric rating on a scale of 1-7 in addition to textual comments.[3] For instance, the teacher created the following guidance for commenting on the "Logic" dimension: "*Provide specific comments about the logic of the author's argument. If points were just made without support, describe which ones they were. If the support provided doesn't make logical sense, explain what that is. If some obvious counter-argument was not considered, ...*" Teacher guidance for numerically rating the logical arguments of the paper was also given. For this history assignment, a rating of 7 ("Excellent") was described as "*All arguments strongly supported and no logical flaws in the arguments.*". A rating of 1 ("Disastrous") was described as "*No support presented for any arguments, or obvious flaws in all arguments.*". Textual review examples for the Flow dimension are provided in Figure 1 of Section 6.

Paper writing & peer review: In the next phase, 24 students submitted their papers online through SWoRD and then reviewed 6 peers' papers. The peer review was done in a "double blind" manner and each paper was reviewed by about 6 peers. As students were required to submit reviews on each dimension separately, SWoRD automatically associates the reviewing dimension with every numerical rating and textual comment. In addition, students also received reviews from one content expert and another writing expert, who reviewed in the same way as the peers did, yielding a final 1405 review comments[4] that we use in this study. In our user study, we will group students based on their writing performance as determined by the numerical peer review ratings. In particular, the average of peers' paper ratings received by each student (**ratingW**) measures the overall quality of a student's *writing performance*.

Backward evaluation: Finally, peer feedback was rated backwards regarding review helpfulness on a scale of 1-7, by the students who received the reviews. For our analysis, we will aggregate the helpfulness ratings for each reviewer, and use the average rating (**ratingR**) as a measure of a student's *reviewing performance*. As this step was not mandatory, the ratingR is only available for 12 students regarding their reviewing performance. (Experts' reviews were excluded in this backward evaluation.)

[3]While reviewing dimensions and associated rubrics are typically created by the teacher, teachers can also use a library provided by SWoRD.

[4]A single peer review can have multiple comments.

5. PRE-DEFINED REVIEW GROUPINGS

Different groupings of the peer reviews allow users to specify different goals in their review analysis tasks. In our user study, we look at two groupings of reviews based on existing review ratings to investigate student writing and reviewing performance. We use them as examples to examine how topic words extracted from a group of reviews can reflect the group's properties. As the instructor specified a different reviewing focus for each dimension, we consider the analysis of reviews on different dimensions as different tasks.

5.1 By paper author's average rating

To investigate student *writing performance*, we split students into high and low performance groups, based on a median split of students' **ratingW**. Then we create "high" and "low" groups of reviews accordingly, based on the group membership of the student who *received* the reviews. The hypothesis is that students who are highly rated have different writing issues compared with those who have lower ratings, and that such differences are reflected in the peer reviews that students receive.

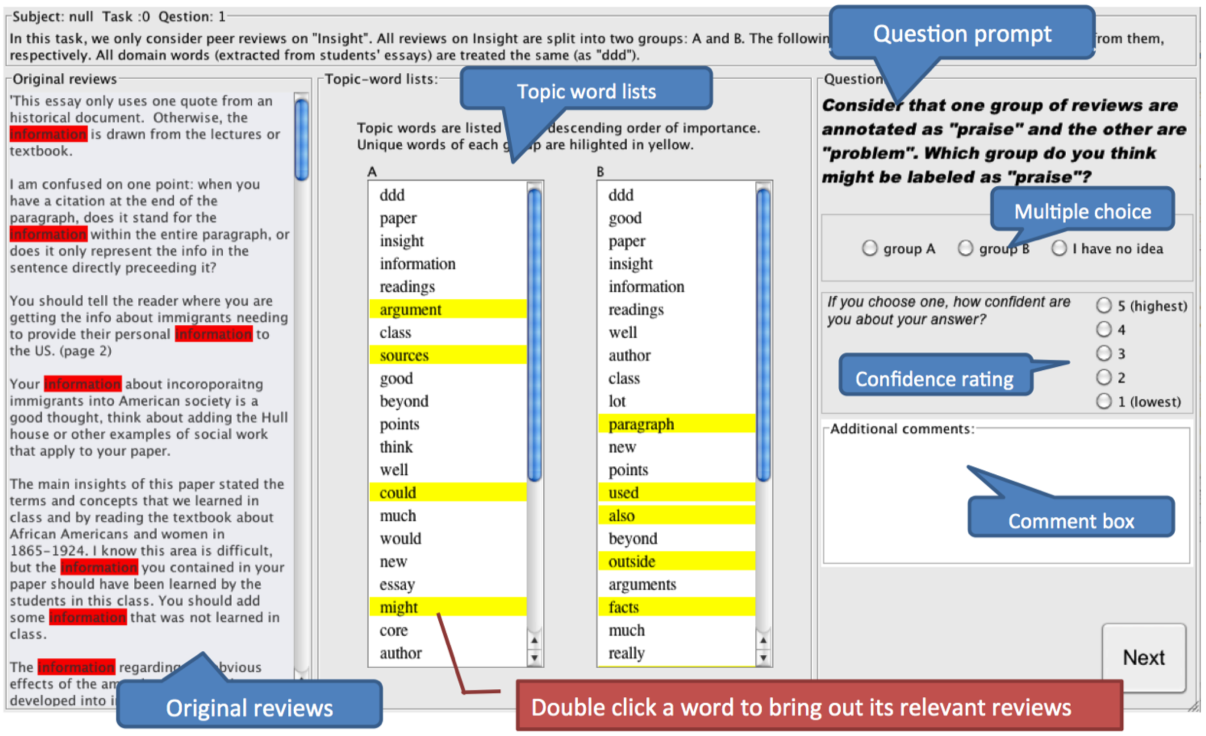
5.2 By reviewer's average helpfulness rating

Similarly, we investigate student *reviewing performance* by splitting students into "high" and "low" groups based on a median split of their **ratingR**. Then we create the "high" and "low" review groups accordingly, based on the group membership of the student who *wrote* the reviews. We hypothesize that review topic words can reveal reviewing issues distinguishing helpful and less-helpful reviews.

6. EXPERIMENT SETUP

We examine whether RevExplore is a useful analytic tool by evaluating the topic-word analytic approach in the context of educational peer-review analysis. In particular, we compare the effectiveness of **two** topic-word extraction methods (*TopicS* and *Freq*) quantitatively using six real peer-review analysis tasks – **two** review groupings (ratingW and ratingR) across **three** reviewing dimensions (Flow, Logic, and Insight). We denote the three factors as *Method*, *Split* and *Dim* respectively. We conduct a formative user study using a $2 \times 2 \times 3$ within-subject design, in which every subject goes through all experimental conditions in random order. In this paper, we analyze users' task performance and user satisfaction both qualitatively and quantitatively.

For task performance analysis, we are interested in three research questions: 1) whether humans can identify the review groups based on their topic words; 2) whether it is feasible to identify any pattern of student peer review performance by comparing peer reviews in groups based on their topic words; 3) whether topic words learned by *TopicS* are more informative than those by *Freq*. Thus we accordingly asked



Subject: null Task :0 Qestion: 1

In this task, we only consider peer reviews on "Insight". All reviews on Insight are split into two groups: A and B. The followin rom them, respectively. All domain words (extracted from students' essays) are treated the same (as "ddd").

Original reviews

'This essay only uses one quote from an historical document. Otherwise, the information is drawn from the lectures or textbook.

I am confused on one point: when you have a citation at the end of the paragraph, does it stand for the information within the entire paragraph, or does it only represent the info in the sentence directly preceeding it?

You should tell the reader where you are getting the info about immigrants needing to provide their personal information to the US. (page 2)

Your information about incoroporaitng immigrants into American society is a good thought, think about adding the Hull house or other examples of social work that apply to your paper.

The main insights of this paper stated the terms and concepts that we learned in class and by reading the textbook about African Americans and women in 1865–1924. I know this area is difficult, but the information you contained in your paper should have been learned by the students in this class. You should add some information that was not learned in class.

The information regarding bvious effects of the am developed into i

Topic-word lists:

Topic words are listed descending order of importance. Unique words of each g p are hilighted in yellow.

A	B
ddd	ddd
paper	good
insight	paper
information	insight
readings	information
argument	readings
class	well
sources	author
good	class
beyond	lot
points	paragraph
think	new
well	points
could	used
much	also
would	beyond
new	outside
essay	arguments
might	facts
core	much
author	really

Question

Consider that one group of reviews are annotated as "praise" and the other are "problem". Which group do you think might be labeled as "praise"?

○ group A ○ group B ○ I have no idea

If you choose one, how confident are you about your answer?

○ 5 (highest)
○ 4
○ 3
○ 2
○ 1 (lowest)

Additional comments:

Figure 1: Interface annotation for peer review analysis user study.

three questions in the user study, e.g. for analyzing student writing performance:

- Q1: *Considering students who received the reviews, which group do you think might be labeled as "high" in terms of their writing performance?*
- Q2: *Within one minute or two, can you figure out how one group of reviews focus on different issues/aspects/ scope compared to the other?*
- Q3: *Comparing the two topic extraction methods, given the correct labels, which method is more helpful in discovering the group difference of reviewing focus and content?*

For user satisfaction analysis, we would like to know how the utility of the proposed idea is affected by user background information, especially participant prior experience of peer review and teaching. We examine these factors in terms of both users' task performance and their reported satisfaction (subjective ratings) in an exit survey.

Participants: All 46 participants are students recruited from a university campus, who are from various academic backgrounds including English, Linguistics, Psychology, Education, Computer Science, etc. Although the tool is designed for instructors, it is quite difficult to recruit a significant number and thus we recruit students instead. Note that some students do have teaching experience and are experienced SWoRD users. To understand whether such background plays a significant role in the use of RevExplore, we record user background information especially regarding demographic factors that depict participants' prior experience in peer review and teaching: whether they have peer-review experience before ($expPR$), whether they used SWoRD before ($expSWoRD$), whether they were a TA before ($expTA$), and whether they have graded any writing assignment before ($expGW$). Participant distribution over these factors is presented in Table 1. Although we also look at other demographic factors such as age, gender, major, etc., due to the space limit, we do not report them in this paper.

Table 1: User distribution over demographic factors that are related to peer-review and teaching.

Factor	Frequency_no	Frequency_yes
$expPR$	23	23
$expSWoRD$	29	17
$expTA$	23	23
$expGW$	29	17

Procedure: Before being exposed to the analysis tasks, participants were first given instructions about the peer-review assignment, including both the paper topics and the reviewing rubrics. We also provided a warm up example to demonstrate how to analyze peer reviews through our user study interface. Figure 1 is a screenshot of the interface, which consists of three parts: the left pane displays the original reviews in lowercase after removing non-ascii characters; the middle pane shows the topic words extracted from the two groups of reviews; the right pane shows the analysis

Table 2: Descriptive statistics of user satisfaction. Higher rating means more positive opinion except for Q_textRef and Q_textImp. One sample t-test test value = 3 (neutral). Significant items are highlighted in bold ($p < 0.05$).

Question	Content	Mean	Std.Error	Sig.(2-tailed)
Q_easyness	Is it easy to make sense of reviews by comparing topic words?	2.85	.140	.291
Q_listDiff	Do the two lists of topic words from the two review groups look semantically different to you?	3.52	.123	.000
Q_layout	What do you think of the list-layout of topic words for comparison purpose?	3.57	.154	.001
Q_reviewDiff	What do you think of topic comparison in helping you identify the differences in the peer reviews?	3.54	.145	.000
Q_largeData	What do you think of topic comparison in helping you make sense of large amount of peer reviews?	3.93	.177	.000
Q_approach	How do you like the idea of exploring peer reviews by comparing them in groups using their topic words? (comparing to reading the textual reviews?)	3.46	.180	.015
Q_textRef	How often did you refer to the original reviews to make sense of the topic words?	1.96	.189	.000
Q_textImp	How important is the original reviews for you to analyze the group differences?	2.93	.171	.705

questions. During the study, if participants feel that some topic word is hard to interpret, they can double click the word on the list to bring out its related reviews in the original review pane. The overall length of the user study was about an hour.

During the user study, the participants completed all tasks in random order. For each task, we computed the same number of topic words for the high and low review groups using *Freq* and *TopicS*, and randomly picked one extraction method for a participant to examine first. For a given method, we presented the corresponding topic words in two lists, one for each group. And we asked participants the same questions Q1 and Q2 regarding the group differences without revealing the group labels. In order to exclude the impact of revealing the group labels for examining the first method, when switching to the second extraction method, we randomly layout the two list of topic words computed by the second method and then asked Q1 and Q2 again. After participants visited both methods, we allowed them to revisit the topic words computed by both methods with correct group labels attached, asking them to vote on which method generated more informative words in terms of identifying the different review focus between the two groups (Q3). In Q1, participants needed to provide their prediction or check "I have no idea"; in Q2, participants needed to answer either yes or "no", and they could also articulate what patterns they found in free text; in Q3, participants needed to vote for the better method or check "no preference".

After the user study, the participants took an exit survey to rate the utility of the two methods as well as the topic-word analytics in general for analyzing students' peer reviews. There are eight subjective questions in the survey. Participants gave their opinions in a scale of 5 points, with 3 being neutral. Survey questions and the descriptive statistics of user satisfaction are presented in Table 2.

7. EXPERIMENT RESULTS

The statistics of the task performance are summarized in Table 3. For measuring task performance, we use the following scheme to code participants' answers to the three questions:

$$Answer1 = \begin{cases} 1 & \text{if the answer is correct,} \\ -1 & \text{if the answer is incorrect,} \\ 0 & \text{if "I have no idea".} \end{cases} \tag{1}$$

$$Answer2 = \begin{cases} 1 & \text{if yes,} \\ 0 & \text{if "no".} \end{cases} \tag{2}$$

$$Answer3 = \begin{cases} 1 & \text{if vote for } \textit{TopicS,} \\ -1 & \text{if vote for } \textit{Freq,} \\ 0 & \text{if "no preference".} \end{cases} \tag{3}$$

For each question, we compare participants' answers to random guess using a one sample t-test to check if using the topic words (extracted by either method) is generally meaningful for our peer-review analysis tasks. As the table shows, in general, the proposed approach is better than random guess (the corresponding test mean is: 0, 0.5, 0), and the proposed topic extraction method (*TopicS*) yields better task performance than the baseline (*Freq*). However, we also notice that the task performance varies with the analysis tasks. This motivates us to further examine the effects of *Split* and *Dim*, as well as their interaction with *Method*, which is discussed later.

To analyze user satisfaction, we compare participants' rating of each survey item to the neutral state (3-point) using a one sample t-test. As Table 2 shows, despite that participants generally think the analysis task is neither easy or difficult,

Table 3: Summary of estimates of the variables across all different conditions, with higher mean bolded between the two extraction methods. It shows that *TopicS* generally yields higher mean compared with *Freq*, except for predicting the label of topic words when reviews were grouped by ratingR for Logic.

			Q1		Q2		Q3	
			Answer1		*Answer2*		*Answer3*	
Dim	Split	Method	Mean	Std.Error	Mean	Std.Error	Mean	Std.Error
Flow	ratingR	*Freq*	-.217	.135	.457	.074	.217	.109
		TopicS	**.413**	.127	**.565**	.074		
	ratingW	*Freq*	-.043	.139	.217	.074	.652	.109
		TopicS	**.022**	.144	**.783**	.061		
Insight	ratingR	*Freq*	.043	.132	.522	.074	.370	.130
		TopicS	**.326**	.128	**.543**	.074		
	ratingW	*Freq*	.109	.133	.283	.067	.565	.115
		TopicS	**.391**	.134	**.717**	.067		
Logic	ratingR	*Freq*	**.391**	.118	.304	.069	.283	.138
		TopicS	-.174	.133	**.587**	.073		
	ratingW	*Freq*	.109	.140	.391	.073	.261	.137
		TopicS	**.152**	.135	**.522**	.074		
Together		*Freq*	.07	.190	.36	.482	.96	2.068
		TopicS	**.19**	.923	**.62**	.486		

Table 4: Summary of Type III F-tests significance of fixed effects of *Method*, *Split*, *Dim* and their interactions on all variables of all three questions. Results are presented in p-value, with significant ones highlighted with "*" ($p < .05$).

	Q1		Q2	Q3
Source	*Answer1*	*Correct*	*Answer2*	*Answer3*
Dim	.907	.000*	.387	.289
Split	.000*	.297	.789	.055
Method	.196	.039*	.000*	na
Dim*Split	.015*	.533	.912	.226
Dim*Method	.008*	.001*	.364	na
Split*Method	.333	.863	.003*	na
Dim*Split*Method	.001*	.040*	.004*	na

they did express positive opinions towards the effectiveness of the topic word extraction methods (Q_listDiff), the list-layout of the topic words (Q_layout), and the usefulness of the topic-word based comparison approach for peer review analysis (Q_reviewDiff, Q_largeData and Q_approach). In addition, though participants rarely refer to the full review text (Q_textRef), they have neutral opinion towards the importance of having access to the full review text during the tasks (Q_textImp).

7.1 Task performance analysis

To further understand the impact of grouping and dimension on the utility of the topic word extraction methods, we use a mixed linear model to analyze the main effects of *Split*, *Dim*, *Method* as well as their interactions. Here we refer to the results of Type III F-tests[5] as recommended in SPSS, for Type III F-tests measure the effect of the target factor in question while controlling all else in the model. A summary of the observed significant effects in our analysis is outlined in Table 4.

Can we identify the review groups by their topic words? To answer our first research question, we took a further look at the correct cases using an indicator variable "Correct" which codes correct cases as 1 and codes both incorrect and "I have no idea" as 0. When using the linear mixed model to analyze the fixed effects on *Correct* (as summarized in Table 4), we found that *Method* and *Dim* are significant ($p < .05$), while *Split* is not. This indicates that *TopicS* can generate more informative topic words than *Freq*, regardless of how we group the reviews, though some reviewing dimensions are naturally more difficult for capturing group properties using the topic words. We also observed significant interaction effects between *Method* and *Dim*, and among all three factors. This implies that how much better *TopicS* is compared to *Freq* is affected by how we set up the review groups for comparison (related to both *Split* and *Dim*), which corresponds to the specific investigation goals of the analysis tasks.

[5]Type III F-tests compute sum of square as the partial sum of squares for each effect in the linear mixed model.

Do topic words reveal patterns in writing and reviewing performance? When tested on *Answer2* using the linear mixed model, only *Method* is found to be significant ($F(1, 530.831) = 40.015$, $p < .001$). An interaction exists between *Method* and *Split* ($F(1, 530.831) = 8.644$, $p = .003$), and among all factors ($F(1, 349.122) = 5.677$, $p = .004$). This tells that using the proposed *TopicS* is more likely to identify review patterns that are different between groups, regardless of which dimension the reviews are on. However, the utility of topic words is also influenced by the grouping, where *TopicS* typically outperformed *Freq* when used for analyzing writing performance, especially on Flow and Insight (as shown in Table 3).

Does the proposed approach extract more informative topic words? The analysis on the fixed effects of *Method* above already showed that *TopicS* can better support users in peer review analysis. Table 3 also shows that *TopicS* is preferred to *Freq* across all tasks. And further analysis with a mixed model (Table 4) shows that such preference is not influenced by either *Split* or*Dim*.

7.2 User background analysis

With respect to user background differences, we focus on demographic factors that are related to peer-review and teaching. We investigate expPR, expSWoRD, expTA and expGW by analyzing both user satisfaction and user-study task performance.

7.2.1 Measured on user satisfaction

For each survey question, we use oneway ANOVA to examine the ratings against each background factor as a binary independent variable. Results are summarized in Table 5.

Table 5: Oneway ANOVA analysis of user-background factors (binary) on user satisfaction. Factors that are significant ($p < 0.05$, highlighted with "*") or in trend are denoted by the mean value of the "yes" group.

Question	expPR	expSWoRD	expTA	expGW
Q_easyness	3.78*	3.24*		
Q_listDiff				
Q_layout				
Q_reviewDiff		4.0*		
Q_largeData		4.35		
Q_approach			3.83*	3.88
Q_textRef		1.29*		1.47*
Q_textImp		2.41*		2.47*

With respect to participants' peer-review experience, students who did peer review before ($expPR = yes$) generally think the review analysis tasks much easier than students who never did it before ($p = .033$). In particular, SWoRD users feel the proposed approach more useful than non-SWoRD users ($expSWoRD = yes$) in helping them identify the peer review differences ($p = .014$). With respect to teaching experience, it is important to note that students who have teaching experience ($expTA = yes$) like our idea of exploring peer reviews by comparing them in groups using their topic words ($p = .039$). Their feedback can somehow approximate instructors opinions towards RevExplore, which suggests the usefulness of the proposed idea for instructors to examine their peer review data in real life. While participants generally have a neutral attitude to the importance of their access to the original review in full text, students who have used SWoRD ($expSWoRD = yes$) or graded writing assignments ($expGW = yes$) before rely on this information much less than the others ($p = .006$, $p = .048$, respectively), and they think it less important than the others as well ($p = .018$, $p = .037$, respectively). This indirectly reflects the effectiveness of our topic-word approach for peer review analysis.

7.2.2 Measured on task performance

To investigate how user background factors influence the task performance, we look at all participants' task performance across all conditions, considering *expPR*, *expSWoRD*, *expTA* and *expGW* as between-subjects effects and *Method*, *Split* and *Dim* as within-subjects effects. In this setting, we use the repeated-measures linear model provided by SPSS to run a Mixed Model ANOVA. First, we look for any main effect caused by the between-subjects factors; second, we examine the interactions between within- and between-subjects factors which show up in the within-subjects section of the repeated-measures analysis.

First of all, there is no significant interaction or main effect of the between-subjects factors observed on participants' answers to any of the review analysis questions. This means that users' prior experience in teaching and peer-review does not directly influence their task performance, which indirectly validates our using college students as the user study subjects.

However, user background factors do exert impact on the utility of topic-word analytics, as these factors qualify the effects of the within-subjects factors, especially *Method*, as summarized in Table 6. It is interesting to see that none of *expPR*, *expSWoRD*, *expTA* or *expGW* interacts with Method by itself alone, but in pairs. For identifying review groups (Q1), *expTA* occurs in both interactions (*Method*expSWoRD *expTA* and *Mehtod*expPR*expTA*), while the other between-subjects factor is about peer review. When peering into the group differences, participants who have both teaching and peer-review experience tend to have better performance (based on modified population marginal mean). For *Method*expSWoRD*expTA*, SWoRD users who have TA experience exhibit better performance when using *TopicS* than using *Freq*, though such difference was not observed when we examined *Mehtod*expPR*expTA*. With respect to *Dim* (examining *Dim*expPR* on Q1), peer-review novels achieved their best performance on Logic, while participants who have peer-review experience did best on Insight. For both groups Flow is the most difficult dimension. Furthermore, to which extend *TopicS* is better than *Freq* is influenced by the interaction between *Dim* and user's peer-review experience (*expPR*/*expSWoRD*).

In addition, we also observed that the main effects of the within-subjects factors given the presence of the between-subjects effects generally follow the pattern of Table 4 (which does not consider between-subjects effects), thus we do not discuss them here again.

Table 6: Summary of Mixed Model ANOVA of within-subjects effects, including interactions between user-background factors (between-subjects effects) and Method, Split, Dim (within-subjects effects). Significant results are presented in p-value ($p \leq .05$).

	Q1		Q2	Q3
Source	*Answer1*	*Correct*	*Answer2*	*Answer3*
Dim*expPR	$F(2,66) = 3.1$, $p = .050$	$F(2,66) = 3.6$, $p = .032$		
Method*expSWoRD*expTA	$F(1,33) = 7.4$, $p = .001$	$F(1,33) = 6.1$, $p = .019$		na
Mehtod*expPR*expTA	$F(1,33) = 9.6$, $p = .004$	$F(1,33) = 4.2$, $p = .049$		na
Mehtod*expTA*expGW			$F(1,33) = 6.1$, $p = .019$	na
Dim*Method*expPR	$F(1,66) = 3.4$, $p = .040$			na
Dim*Method*SWoRD	$F(2,66) = 4.0$, $P = .022$	$F(2,66) = 5.5$, $p = .006$		na

8. CONCLUSIONS

In this paper we evaluate the topic-word analytics for analyzing educational peer reviews with a user study. The user study shows that student peer reviews can be used to examine student writing and reviewing performance based on peer review topic words, and that the proposed comparison-oriented topic-word extraction method (*TopicS*) suits our analytic tasks best compared with the frequency based method (*Freq*). However, the utility of the learned topic words is influenced by the analytic goals (specified through review grouping) and dimensions, as well as users' prior experience in teaching and peer-review. Analysis of user satisfaction shows that participants who have teaching experience significantly favor our approach more than the others, which suggests the usefulness of the proposed approach in supporting instructors for analyzing student peer reviews in the real-world. Even though we did not include manual digestion of original peer reviews as a baseline, we indirectly compare it with our topic-word approach in the exit survey (Q_approach).

In the future, we would like to evaluate the proposed approach in the context of RevExplore, which allows users to specify analytic goals at runtime. Finally we hope to integrate RevExplore into SWoRD as part of the teacher dashboard to support interactive review content analytics.

9. ACKNOWLEDGEMENTS

This work is supported by Andrew Mellon Predoctoral Fellowship 2012-2013 and IES award R305A120370.

10. REFERENCES

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[2] J. Boyd-Graber, J. Chang, S. Gerrish, C. Wang, and D. Blei. Reading tea leaves: How humans interpret topic models. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*, 2009.

[3] K. Cho. Machine classification of peer comments in physics. In *Proceedings of the First International Conference on Educational Data Mining*, pages 192–196, 2008.

[4] K. Cho and C. D. Schunn. Scaffolded writing and rewriting in the discipline: A web-based reciprocal peer review system. *Computers and Education*, 48(3):409–426, 2007.

[5] T. Dunning. Accurate methods for the statistics of surprise and coincidence. *Comput. Linguist.*, 19(1):61–74, 1993.

[6] I. Goldin and K. Ashley. Peering inside peer review with bayesian models. In *Artificial Intelligence in Education*, pages 90–97. Springer, 2011.

[7] B. Lehman, W. L. Cade, and A. Olney. Off topic conversation in expert tutoring: Waste of time or learning opportunity. In *EDM'10*, pages 101–110, 2010.

[8] C.-Y. Lin and E. Hovy. The automated acquisition of topic signatures for text summarization. In *Proceedings of the 18th conference on Computational linguistics*, volume 1 of *COLING '00*, pages 495–501, Stroudsburg, PA, USA, 2000. Association for Computational Linguistics.

[9] F.-R. Lin, L.-S. Hsieh, and F.-T. Chuang. Discovering genres of online discussion threads via text mining. *Computers & Education*, 52(2):481–495, 2009.

[10] N. Ming and E. Baumer. Using text mining to characterize online discussion facilitation. *Journal of Asynchronous Learning Networks*, 2011.

[11] M. M. Nelson and C. D. Schunn. The nature of feedback: how different types of peer feedback affect writing performance. *Instructional Science*, 37:375–401, 2009.

[12] A. Nenkova and A. Louis. Can you summarize this? identifying correlates of input difficulty for generic multi-document summarization. In *Proceedings of Association for Computational Linguistics*, 2008.

[13] W. Xiong and D. Litman. Automatically predicting peer-review helpfulness. In *Proceedings 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011.

[14] W. Xiong, D. Litman, J. Wang, and C. Schunn. An interactive analytic tool for peer-review exploration. In *Proceedings of the 7th Workshop on the Innovative Use of NLP for Building Educational Applications*, pages 174–179, 2012.

[15] W. Xiong and D. J. Litman. Identifying problem localization in peer-review feedback. In *Proceedings of Tenth International Conference on Intelligent Tutoring Systems*, volume 6095, pages 429–431, 2010.

[16] Z. Zhai, B. Liu, H. Xu, and P. Jia. Constrained lda for grouping product features in opinion mining. *Advances in Knowledge Discovery and Data Mining*, pages 448–459, 2011.

Mining Social Deliberation in Online Communication

If You Were Me and I Were You

Xiaoxi Xu, Tom Murray, Beverly Park Woolf and David Smith
School of Computer Science
University of Massachusetts
Amherst, MA
{xiaoxi,tmurray,bev,dasmith}@cs.umass.edu

ABSTRACT

Social deliberative skills are collaborative life-skills. These skills are crucial for communicating in any collaborative processes where participants have heterogeneous opinions and perspectives driven by different assumptions, beliefs, and goals. In this paper, we describe models using lexical, discourse, and gender demographic features to identify whether or not participants demonstrate social deliberative skills from various online dialogues. We evaluate our models using three different corpora with participants of different educational and motivational levels. We propose a protocol about how to use these features to build models that achieve the best in-domain performance and identify the most useful features for building robust models in cross-domain applications. We also reveal lexical and discourse characteristics of social deliberative skills.

Keywords

Social deliberative skills, collaborative problem solving, collaborative learning, collaborative knowledge-building, discourse analysis, applied machine learning, feature engineering

1. INTRODUCTION

Learning is often depicted as a social process that includes collaborative knowledge-building and problem solving. In this "situated" perspective on learning, learners often must negotiate differing perspectives or goals to build knowledge or solve problems collaboratively. Previous research [25] has shown that certain communication skills, such as listening with empathy and perceiving and responding to other's emotions, part of the collective intelligence of groups, can improve group performance on a wide variety of tasks, such as brainstorming, making collective moral judgments, and negotiating over limited resources. In this research, we focus explicitly on identifying similar skills that are called for to handle diverse opinions and perspectives. For example, do participants attentively listen to each other's opinions? Do they make a good faith effort to understand perspectives other than their own? These skills, including cognitive empathy, affective empathy, and reciprocal role-taking, are part of what we call *social deliberative skills* [14].

Social deliberative skills are at the overlap of cognitive skills and social/emotional skills. Specifically, a participant should present rational arguments with supporting evidence in order that his view be taken seriously and valued. Similarly, one has to turn down the volume of his own thoughts to attentively listen to other's opinions and has to intentionally switch the channel from "me" to "you" to be able to understand or even appreciate another's perspectives. Indeed, this "cognitive empathy" of "*if you were me and I were you*", the soul of social deliberative skills, is needed in any sphere of human interaction, from collaborative learning, to marriage, to workplace relationship, and to world affairs. The ultimate goal of our research is to support social deliberative skills in online communication. In this research, we explore the possibility of automatically assessing and predicting the occurrence of social deliberative skills.

Creating computational models for assessing social deliberative skills has profound implication on several fronts: it (1) supports more efficient analysis for research purposes into online communication and collaboration in social processes; (2) provides assessment measures for evaluating the quality and properties of group dialogues; and (3) provides tools for informing facilitators to adjust skill support and intervention efforts [13]. Previous research in learning science has extensively focused on creating educational software that supports cognitive skills in collaborative environments, such as inquiry skills, metacognition and self-regulated learning skills, and reflective reasoning skills [24, 3, 5, 19, 11]. Research in these areas has provided a deep theoretical context for studying the cognitive aspect of social deliberative skills. A burgeoning body of research has begun to study the social relational aspect of collaborative processes, such as influence [20] and up-taking [21]. This line of research has mainly used structural features of social interactions, such as reply structure, linking notes in a conceptual framework, as well as spatial and temporal proximity to address the questions of who are the central actors in discussions and whose ideas receive the most development. But, collaboration interactions generally take place in the form of natural language. It is reasonable to suppose that language-level features, including lexical features (i.e., what is said) and discourse features (i.e., how it is said) could provide crucial insights into the characteristics of social deliberative skills that are called for in collaborative problem solving and communication in general.

In this paper, we create computational models for assessing social deliberative skills in online communication. The online dialogues that we study are from participants ranging from undergraduate students of multiple disciplines, to highly-educated academic professionals, to members of the general public. We first analyze these online dialogues through a variety of lexical, discourse, and gender demographic features and then create machine learning classifiers to recognize social deliberative skills. To the best of our knowledge, this research is the first work that implements the state-of-the-art conceptual framework of social deliberation. This paper makes four contributions: (1) development of an automatic system for predicting social deliberation, (2) discovery of which type of features or feature combinations are the best for building social deliberative classifiers and under which conditions, (3) discovery of which type of features are the best for building a *robust* social deliberative classifier across domain changes, and (4) identification of language characteristics of social deliberation. These contributions lie at the intersection of machine learning, education, computational social science, and communication studies.

The rest of the paper is organized as follows. In Section 2, we introduce the concept of social deliberative skills. In Section 3, we describe three experimental domains. Section 4 introduces experimental design and methodology. We discuss experimental results in Section 5 and conclude with future work in Section 6.

2. SOCIAL DELIBERATIVE SKILLS

Social deliberative skills involve the application of cognitively-oriented higher-order skills to thinking about the perspectives of others and, consequently, of the self as well. In other words, social deliberative skills require that a speaker reflect not only upon a purely *objective* idea (e.g., a topic) but also upon *my* ideas, *your* ideas, *our* ideas, and *their* ideas. Tracing the origins of this phrase also describes its meaning: to live with others (social) and to balance (deliberative) differences (skills). Our prior research [14] has defined a theoretical framework for social deliberative skills, which includes a group of high-order communication skills that are essential for different tasks and stages of communication that involves a disequilibrium of diverse perspectives. These component skills include social perspective seeking (i.e., social inquiry), social perspective monitoring (i.e., self-reflection, weighting opinions, meta-dialogue, meta-topic, and referencing sources for supporting claims), as well as cognitive empathy and reciprocal role-taking (i.e., appreciation, apology, inter-subjectivity, and perspective taking). Here is an example of "perspective taking" from authentic dialogue in our corpora: *I can't help but imagine what that is like, for her and for her family.* As an another example, the following statement is about "self reflection": *I am probably extremely bias because I am under 21 years old and in college. I wonder if as a 45 year old I will feel differently.*

Social deliberative skills can also be seen as a composite skill [15], which, though less precise can serve as a general marker of social deliberation, for use in evaluation and real-time feedback in intervention. In this study, we focus on creating computational models to assess whether participants of online dialogues demonstrate the use of *composite social deliberative skill* (or social deliberative behavior, SBD).

3. CORPORA

Problem solving and negotiating with others at some level are a regular part of our lives. These actions represent typical everyday communication situations where social deliberative behavior is needed. In this study, we examined three online corpora, two of which involve participants in discussions addressing separate ill-defined problems and one of which involves participants in a negotiation.

In the first domain, **civic deliberation**, posts were collected from a civic engagement online discussion forum at *e-democracy.org*. Thirty two participants discussed ethnic issues and suggesting ways to alleviate tensions about their multi-racial community. These participants were self-selected with an implicit goal to improve community relations. Participants were mostly level-headed and demonstrate social deliberative behavior (SDB) repeatedly. In this domain, SDB occupied 57% of the total 396 annotated segments [1], see Table 1.

In the second domain, **college dialogues**, posts were collected from college students participating in computer-mediated discussions. Ninety undergraduate students from a variety of disciplines discussed about controversial topics. The topics included "should the legal drinking age be lowered in Massachusetts?" and "what are the pros and cons of using FaceBook or other social networking software as part of high school curriculum?" These discussions were part of experimental trials with the goal of assessing online educational software tools that support SDB. In contrast to participants from the other two domains that were self-motivated to be deliberative, participants in this group received class credit and were encouraged to participate. In this domain, SDB occupied only 32% of the total 1783 annotated segments.

In the third domain, **professional community negotiation**, email exchanges were collected from sixteen geographically dispersed faculty participants who did not know each other and who were from two academic communities. These faculty members negotiated about a proper solution to a conference scheduling conflict. An emerging theme in this dialogue was the tension between democratic decision-making versus top down fiat decision-making by those in authority. Participants were highly educated academic professionals, most of whom encouraged democratic decision making about relocating the conference, which partly led to a more deliberative dialogue. In this domain, SDB occupied 53% of the total 438 annotated segments.

In order to provide training data for machine learning models to automatically assess whether or not participants perform social deliberation, two independent trained human judges had annotated the three corpora based on the social deliberative skill scheme [2]. We achieved good inter-rater re-

[1]Posts were segmented manually at speech act boundaries, and there are typically 3-5 segments per post.

[2]We developed a hand coding scheme containing over 50

Table 1: Data statistics with various domains

Domain	Social deliberative behavior	Other speech acts	Total segment count	Participant count
Civic deliberation	225 (57%)	171 (43%)	396	32
Professional community negotiation	231 (53%)	207 (47%)	438	16
College dialogues	565(32%)	1218(68%)	1783	90
All	1021(39%)	1596(61%)	2617	138

liability scores for both composite and component social deliberative skills as measured using Cohen's Kappa statistics across domains. Note that the social deliberative behavior (or the composite social deliberative skill) is an aggregate over component social deliberative skills. The inter-rater reliability scores of social deliberative behavior for the civic deliberation domain, college dialogues domain, and professional community negotiation domain were 73.5%, 64.3%, and 68.4%, respectively.

4. EXPERIMENTAL DESIGN AND METHODOLOGY

The goal of experiments in this section is to address the following two research questions. First, which type of features (i.e., lexical, discourse, and gender demographic features) or feature combinations are the best for building social deliberative classifiers for each domain? Second, which type of features are the best for building a *robust* social deliberative classifier across domain changes? To this end, we designed two experimental scenarios.

- **Scenario 1: In-domain analysis for each domain**
- **Scenario 2: Cross-domain analysis for each pair of domains**

In both scenarios, we study *feature effects* on prediction performance of machine learning models. Specifically, we build machine learning models using different feature sets and their possible combinations to see which leads to the best prediction performance. We have three types of features (i.e., lexical, discourse, and gender demographic), so we evaluate a group of 6 possible feature configurations.

These two scenarios differ in the following way. The first scenario allows features comparison for each domain and provides the basis for evaluating cross-domain performance. The second scenario offers a systematic view of how machine learning models built with different feature sets perform across domains. We have three domains, so we will evaluate all 6 possible combinations of domain pairs.

With respect to performance measures, we use accuracy (% of correct identification of social deliberative behavior (SDB)), precision (% correct of identified as SDB), recall (% labeled as "SDB" that were predicted to be "SDB"), and F_2 measure (the harmonic mean of precision and recall that weights recall twice as high as precision). Recall is more valued than precision in this study for two reasons. First, the social deliberative skill scheme is expanding, so the SDB considered in this research is by no means complete. The second reason is relevant to our planned applications. Our first planned application to real-time deliberation is through a Facilitators Dashboard. The Dashboard will alert facilitators to potentially important patterns and metrics in the dialogues they are monitoring, in order to help them decide when and how to perform interventions. Because facilitators can intelligently filter out dubious analysis, our algorithms should err on the side of identifying all important patterns, at the risk of including some false positives.

annotations on social deliberative skills and other speech acts.

4.1 Features

Computational understanding of social deliberation is an unexplored research territory. In choosing features for this study, we recognize that we lack sufficient knowledge of what features might be predictive of social deliberative behavior. Therefore, to explore possible features, we turned to the literature of social, psychology, and psycholinguistic studies. This research is the first to use lexical, discourse, and gender demographic features to characterize linguistic patterns of social deliberative behavior.

4.1.1 Lexical features – LIWC

LIWC, Linguistic Inquiry Word Count [18], is a lexicon based linguistic system. It was created by analyzing the utterances of over 24,000 participants totaling over 168 million words. LIWC produces groups of words from 82 language dimensions through a word counting approach. These 82 groups fall into 10 general categories: *linguistic processes*, *social processes*, *affective processes*, *cognitive processes*, *perceptual processes*, *biological processes*, *relativity*, *personal concerns*, *spoken categories*, and *punctuation.*

LIWC has gained a trusted reputation for tracking linguistic features that are indicative of social and psychological phenomena. For example, when investigating gender differences in linguistic styles using LIWC features, researchers in [1] found significant differences between genders for the use of self references, but not for the use of social words and positive and negative emotion words. In [23], LIWC features helped find the roles that emotional and informational supports play in participants' commitment in online health support groups. In another study [8], LIWC helped identify the communication characteristics of terrorists and authoritarian regimes. Given a wealth of evidence of the effectiveness of LIWC features in decoding people's communication and interaction styles from the language they use, we expect that LIWC features can contribute to demystifying the link between language and social deliberation.

4.1.2 Discourse features – Coh-Metrix

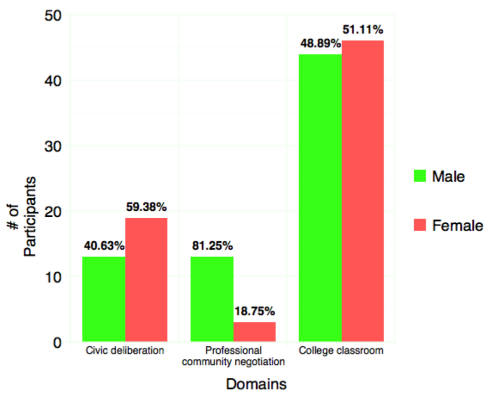


Figure 1: Gender distributions in various domains

Coh-Metrix [7] is a discourse model aimed at better understanding of discourse comprehension, communication breakdowns and misalignments. It was initially developed to explore cognitive constructs of cohesion in written text. Cohesion here refers to the linguistic features that explicitly link words, propositions, and events in a text, which in turn facilitate a reader's coherent mental representation of a text [7]. Coh-Metrix tracks word-level features that are similar to LIWC, but also incorporates modules and algorithms that assess collocations of words. Specifically, Coh-Metrix produces approximately 100 measurements that fall under 8 categories: *narrativity*, *referential cohesion*, *syntactic simplicity*, *word concreteness*, *causal cohesion*, *logical cohesion*, *verb cohesion*, and *temporal cohesion*.

Much like LIWC, Coh-Metrix has been widely used as a computational psycholinguistic tool for predicting complex phenomena, such as affect states, personality, deception, and even physical and mental health outcomes [9, 12, 2, 4]. Given that Coh-Metirx provides a platform for a systematic and deeper analysis of discourse contents, we believe that it can uncover subtle linguistic characteristics relevant to social deliberative behavior.

4.1.3 *Demographic feature – gender*

Previous research [25] has revealed a formula for successful teams in group environments (e.g., business, classroom, or at home). The formula indicates: *People willing to listen and empathize + people with social sensitivity (i.e., perceive and respond to other's emotions) = smart effective teams able to achieve in any environment.* That research concludes by noting that adding women to a team helps improve group performance. This is because women were found to score higher on average on social sensitivity. Motivated by theses research findings, we decided to incorporate gender as a factor in our analysis. In Figure 1, we show gender distributions in different domains. In future research we will collect other demographic data, such as education level, age, and political orientation, and test their predictive power of social deliberative behavior.

4.2 Machine Learning Models

In this study, we face the problems of small training data and high dimension feature space. In choosing machine learning models to identify social delineative behavior, we prefer a model that meets the following requirements. First, the model is able to select important features automatically during learning. Second, the model performs well with a low ratio of training data size to the number of feature variables. Third, the learnt model is transparent and easy to interpret (i.e., "glass box" model).

As we show below, L_1 Regularized Logistic Regression (L_1RLR) is a model that satisfies our needs. L_1RLR performs feature selection and learning simultaneously. It formulates the learning problem as a trade-off between minimizing loss (i.e., achieving good accuracy on training data) and choosing a sparse model (i.e., improving generalization in prediction on unseen data, higher interpretability, and computational savings).

Before we describe L_1RLR, let us recall that the logistic loss function is defined as:

$$p(y|x;\mathbf{W}) = \frac{1}{1+\exp(-\mathbf{W}^T x)}$$

where x is the training data, y is the response variable, and $\mathbf{W}$ is the model we learn.

In L_1 Regularized Logistic Regression, we solve the following optimization problem:

$$\arg\max_{\mathbf{W}} \sum_i \log(p(y_i|x_i;\mathbf{W}) - \lambda * \Omega(\mathbf{W})$$

where $\Omega(\mathbf{W})$ is a regularization term used to penalize large weights. In L_1RLR, $\Omega(\mathbf{W})$ is the L_1 norm [22], which is also called least absolute shrinkage and selection operator (Lasso), described below:

$$\Omega(\mathbf{W}) = ||\mathbf{W}||_1 = \Sigma_i|\mathbf{W}_i|$$

Lasso produces a laplace (i.e., double exponential) prior that is "pointy" at zero, which allows feature shrinkage and selection. It is different from the classical L_2 norm [10], also referred to as ridge norm, because L_2 norm produces a Gaussian prior that is near zero and therefore imposes no sparsity. Previous research [16] has shown that L_1 regularization requires the number of training examples that grows logarithmically with the number of features to learn well.

In this study, we used the l_1 regularized dual averaging algorithm [26] for solving L_1 Regularized Logistic Regression. For results reported in Figure 2, we trained l_1RLR (i.e., λ=1, γ=2) [3] with various feature sets. For all in-domain experiments, we report average performance over 10-fold stratified cross-validation within the same domain. For cross-domain experiments, we report results following the training-validating-testing protocol. We trained and validated on the training corpus and tested on the testing corpus.

5. RESULTS AND DISCUSSIONS

Experimental results (Figure 2) reveal a number of interesting patterns. One of the most salient patterns is that imbalanced class/label distribution hurts predictive performance

[3]We also experimented with other values (0.01, 0.1, 10) of λ and found slightly worse performance than the results reported here.

(more on recall than precision), regardless of feature configurations. This can be seen in the third sub-column (i.e., the college dialogues domain) of in-domain results. This observation suggests that before creating a model, it is important to strategically solve the imbalanced data problem, either from the algorithm level (e.g., adjusting class weights or priors) or from the data level (e.g., up-sampling or down-sampling).

Other important yet subtle patterns are explained below. Note that, we use "performance" and "recall" interchangeably in this discussion because recall is the most valued among all the performance measures in this application, as explained earlier.

Gender's effect on predicting social deliberative behavior. The first row in Figure 2 shows that gender *alone* has no predictive power of social deliberative behavior. Specifically, the classification results in each domain reflect the bias of class distribution on training machine learning models toward predicting all data as coming from the majority class. For example, in the college dialogue domain, as shown in Table 1, the majority class is "other speech acts." Classifiers built with various feature configurations unanimously used this bias without any corrections from the gender feature to predict every instance as "other speech acts." This means that every cell in the confusion matrix [4] is zero except the false negative, and therefore recall and precision are zero. This pattern also applies to other domains. We speculate that because social deliberative behavior (as a composite skill) contains skills that greatly overlap cognitive and social/emotional skills, features correlated with only *emotional* related skills, such as social sensitivity, are not effective in predicting social deliberative behavior.

Different capacities of lexical and discourse features in different domains. First, we examine the performance of each feature alone, ignoring feature combinations. As can be seen from in-domain results, compared to LIWC features (70.7% at recall), Coh-Metrix features (83.6% at recall) have the best predictive power on the *civic deliberative domain*. The performance of the model built with LIWC features added on top of Coh-Metrix features has a slight ($< 1\%$) increase in this domain. In the *professional community negotiation domain*, compared to Coh-Metrix features (74.0 % at recall), LIWC features (90.0% at recall) have the upper hand. The performance of the model built with Coh-Metrix features added on top of LIWC features has a drastic ($> 15\%$) decrease in this domain. The *college dialogues domain* has similar patterns as the professional community negotiation domain. In other words, LIWC features are the most predictive for the college dialogues domain. These patterns suggest that lexical and discourse features have different capacities in different domains for the task of predicting social deliberative behavior.

Next, we look at feature combinations. For the *civic deliberation domain*, Coh-Metrix and LIWC features combined, among all 6 feature configurations, led to the best model in that domain. For the *professional community negotiation domain*, LIWC features alone, among all 6 feature configurations, led to the best model in that domain. For the *college dialogues domain*, LIWC and gender feature combined, led to the best model in that domain. This implies that determining which features or feature combinations to use and in which order has an impact on whether and when we will attain the best model. We will explore this point in the text below.

Features for building robust models. Now, we look at the feature effects on predictive performance for cross-domain analysis. The model built with LIWC features using the data from the *professional community negotiation domain* achieved the best cross-domain performance [5]. For example, this model, when applied to the *civic deliberation domain*, achieves 89.3% on cross-domain recall, which is even better than the best in-domain recall (83.6%) achieved by using Coh-Metrix features in this domain. In addition, this model, when applied to the *college dialogues domain*, achieves 86.9% on cross-domain recall, which is much better than the best in-domain recall (9.9%) achieved by using LIWC features in this domain. This observation concludes that LIWC features seem to be the most useful features for building robust models in cross-domain applications. Moreover, when averaging in-domain and cross-domain performance for each feature and feature combinations for each domain, we observe that LIWC features achieved the highest recall (88.8%), followed by Coh-Metrix features (84.5%).

Protocols for using linguistic features to predict social deliberative behavior. The results in Figure 2 imply a protocol about how to use lexical and discourse features to build a model (i.e., l_1RLR) in order to achieve the best in-domain performance. This protocol can be described as follows:

1. Use LIWC features to build a model, whose performance (i.e., recall) is denoted by $p(l)$.
2. Use Coh-Metrix features to build a model, whose performance is denoted by $p(c)$.
3. If $p(l) > p(c)$, the best performance is $p(l)$; otherwise combine LIWC and Coh-Metrix to build a model, whose performance, denoted as $p(lc)$, is the best performance.

This protocol is the most efficient way to find the *right* feature sets for building a model with the best predictive performance. This protocol also suggests that for certain domains LIWC features – features related to "what is said" – are sufficient to predict social deliberative behavior. In these domains, Coh-Metrix features – features related to "how it is said" – might be too overwhelming for the model to achieve good performance. For other domains, the LIWC features are not close enough for identifying the sophistication of social deliberative behavior, and combining with Coh-Metrix features can greatly help increase model performance. In a

[4]In a confusion matrix, each column represents the instances in a predicted class, while each row represents the instances in an actual class.

[5]We witnessed an unstable performance of combining gender with other features. For example, the cross-domain performance of the model built with LIWC and gender combined, compared to that of the model built with LIWC alone, decreases in the civic deliberation domain but increases in the college dialogues domain. Due to the unstable performance of the gender feature, we ignore it for the rest of this study.

broader sense, this protocol evaluated on different corpora provides evidence that determining the right feature set with the best model performance can be streamlined to improve work efficiency. The streamlined processes need to be designed by taking advantage of feature capacities.

Now, we examine the linguistic characteristics of social deliberative behavior. We learnt earlier that, when considering each feature alone, LIWC and Coh-Metrix features have different capacities in different domains for the task of predicting social deliberative behavior. Specifically, Coh-Metrix features are the most predictive in identifying social deliberative behavior in the civic deliberation domain; LIWC features are the most predictive in identifying social deliberative behavior in the professional community negotiation domain and produce the best model for cross-domain prediction tasks. In Table 2, we show the top 10 Coh-Metrix features learnt by L_1 regularized logistic regression built from the civic deliberation domain. Similarly, in Table 3, we show the top 10 LIWC features learnt by L_1 regularized logistic regression built from the professional negotiation domain. Below, we summarize the lexical characteristics of social deliberative behavior and the discourse characteristics of social deliberative behavior.

Lexical characteristics of social deliberative behavior. The *lexical* characteristics of social deliberative behavior, compared to that of "other speech acts," are as follows: shorter message, more dictionary words, fewer big words, fewer words per sentence, more adverbs, fewer pronouns, fewer punctuations, fewer cognitive processes words, fewer space words, fewer auxiliary verbs.

Discourse characteristics of social deliberative behavior. The *discourse* characteristics of social deliberative behavior, compared to that of "other speech acts," include more negative additive connectives, higher negation density, less lexical diversity, less narrativity, shorter message, more pronouns (especially more second person pronouns), fewer spatial motion words, lower word concreteness, fewer connectives.

Examining the linguistic patterns of social deliberative behavior, we found that the LIWC system and the Coh-Metrix system agreed on some features (e.g., a few spatial motion words) and produced incongruent results for others. For example, a few pronouns found by LIWC, whereas many pronouns found by Coh-Metrix. This incompatible finding suggests that social deliberative behavior may have different appearances in different domains. Therefore, in this study we found no conclusive linguistic characteristics of social deliberative behavior.

6. CONCLUSION AND FUTURE WORK

In this paper, we built machine learning models to identify social deliberative behavior from various online dialogues using lexical, discourse, and gender demographic features. We recognized the different capacities of lexical and discourse features in different domains and proposed a protocol about how to use them to build models that achieve the best in-domain performance. We also found that lexical features (i.e., LIWC) were the most useful features for building robust models in cross-domain applications.

Table 2: Top 10 Coh-Metrix features learnt by L_1 regularized logistic regression built from the civic deliberation domain

Coh-Metrix feature	*Interpretation*	*Weight*
CONLOGi	negative additive connectives	9.044
DENNEGi	negation density	8.582
LEXDIVVD	lexical diversity	- 8.122
PNar	narrativity	-7.774
READNW	total number of words	-7.317
PRO2i	second person pronouns	6.03
DENPRPi	pronouns	5.503
SPATlpi	spatial motion words	-4.889
WRDCacwm	word concreteness	-4.69
CONi	all connectives	-4.24

Table 3: Top 10 LIWC features learnt by L_1 regularized logistic regression built from the professional community negotiation domain

LIWC feature	*Interpretation*	*Weight*
WC	word counts	-0.043
Dic	dictionary words	0.037
Six|tr	big words	-0.011
WPS	words/sentence	-0.01
adverb	adverbs	0.009
pronoun	pronouns	-0.009
AllPct	total punctuations	-0.009
cogmech	cognitive processes	-0.007
space	space	-0.004
auxverb	auxiliary verbs	-0.004

In future work, we will include semantic features (e.g., name entity relations) in our models to predict social deliberative behavior. In addition, we will build models using interaction features and structure features to study whether mutual influences, such as linguistic style matching [17], and group dynamics are predictive of social deliberative behavior. We will also investigate whether combining language features and structure features for building models can lead to performance gains. Moreover, we will evaluate the proposed protocol on more data sets to test its external validity and to identify the characteristics of domains with which each feature type (i.e., lexical vs. discourse) works the best. Furthermore, we will create multi-task machine learning models with advanced regularizers (e.g., sparse group Lasso [6]) to simultaneously identify each component social deliberative skills from online dialogues. We hope these endeavors can increase our understanding of the nature of social deliberative behavior and thereby inform the design and development of educational tools to support social deliberative behavior in collaborative processes, from knowledge building, to problem solving, and to communication in general.

7. ACKNOWLEDGEMENT

We thank all anonymous reviewers for their thorough evaluation and constructive recommendations for improving this paper. This research is supported by the US National Science Foundation under grant #0968536 from the division of Social and Economic Sciences. Any opinions or conclusions expressed are those of the authors, not necessarily of the funders.

		In-domain			Cross-domain					
	Training corpus	*Civic deliberation*	*Professional community negotiation*	*College dialogues*	*Civic deliberation*	*Civic deliberation*	*Professional community negotiation*	*Professional community negotiation*	*College dialogues*	*College dialogues*
	Testing corpus	*Civic deliberation*	*Professional community negotiation*	*College dialogues*	*Professional community negotiation*	*College dialogues*	*Civic deliberation*	*College dialogues*	*Civic deliberation*	*Professional community negotiation*
Gender	*Accuracy*	56.8	52.7	68.3	52.7	31.7	56.8	31.7	43.2	47.3
	Precision	56.8	52.7	0.0	52.7	31.7	56.8	31.7	0.0	0.0
	Recall	100.0	100.0	0.0	100.0	100.0	100.0	100.0	0.0	0.0
	F2	86.8	84.8	0.0	84.8	69.9	86.8	69.9	0.0	0.0
LIWC	*Accuracy*	55.3	55.3	67.9	56.2	47.1	59.6	39.3	42.4	47.3
	Precision	58.9	54.6	46.1	56.9	33.4	59.6	32.7	46.8	50.0
	Recall	70.7	**90.0**	9.9	69.7	67.6	**89.3**	**86.9**	9.8	2.6
	F2	68.0	79.7	11.8	66.7	56.1	81.2	65.2	11.6	3.2
Cohmetrix	*Accuracy*	62.1	54.8	68.0	53.4	36.4	53.3	39.0	41.9	47.0
	Precision	61.4	55.3	44.7	53.5	30.7	58.3	30.2	36.8	42.9
	Recall	**83.6**	74.0	3.7	90.0	80.0	62.2	70.3	3.1	1.3
	F2	77.9	69.3	4.6	79.2	60.5	61.4	55.5	3.8	1.6
LIWC+Gender	*Accuracy*	52.3	54.6	68.0	56.2	47.1	60.9	38.3	42.4	48.0
	Precision	56.8	54.2	49.2	56.8	33.2	60.6	32.6	46.8	60.0
	Recall	67.1	89.6	10.6	70.6	66.6	88.9	88.9	9.8	1.3
	F2	64.8	79.3	12.6	67.3	55.4	81.3	66.1	11.6	1.6
Cohmetrix +Gender	*Accuracy*	62.4	55.5	68.2	53.9	35.8	53.8	38.8	40.9	46.2
	Precision	62.7	55.6	46.5	53.9	30.8	58.5	30.1	32.3	33.5
	Recall	83.6	77.5	3.5	87.0	77.9	64.0	70.3	4.0	1.3
	F2	78.3	71.8	4.3	77.5	59.6	62.8	55.5	4.8	1.6
LIWC +Cohmetrix	*Accuracy*	62.4	54.5	68.3	54.1	38.8	55.6	39.4	41.2	48.0
	Precision	63.3	55.1	48.9	54.0	30.9	60.0	30.3	35.7	66.1
	Recall	84.4	74.5	4.1	87.0	75.2	65.3	70.4	4.4	1.3
	F2	79.2	69.6	5.0	77.5	58.4	64.2	55.7	5.4	1.6
All	*Accuracy*	62.1	54.3	68.5	52.7	36.9	54.8	39.8	41.4	47.7
	Precision	62.3	55.0	48.2	53.1	30.6	59.7	31.6	38.7	42.9
	Recall	84.4	74.0	4.6	88.7	78.6	63.1	71.0	5.3	1.3
	F2	78.8	69.2	5.6	78.2	59.8	62.4	56.8	6.4	1.6

Figure 2: Predictve performance (in %) of L_1 regularzied logistic regression built using different feature configurations in different scenarios

8. REFERENCES

[1] C. Bell, P. McCarthy, and D. McNamara. Using liwc and coh-metrix to investigate gender differences in linguistic styles. In *Applied natural language processing and content analysis: Identification, investigation, and resolution*, pages 545–556. Hershey, PA: IGI Global, 2012.

[2] R. Campbell and J. Pennebaker. The secret life of pronouns flexibility in writing style and physical health. *Psychological Science*, 14(1):60–65, 2003.

[3] C. Conati, K. Vanlehn, et al. Toward computer-based support of meta-cognitive skills: A computational framework to coach self-explanation. *International Journal of Artificial Intelligence in Education (IJAIED)*, 11:389–415, 2000.

[4] S. D'Mello, N. Dowell, and A. Graesser. Cohesion relationships in tutorial dialogue as predictors of affective states. In *Proc. 2009 Conf. Artificial Intelligence in Education: Building Learning Systems That Care: From Knowledge Representation to Affective Modelling, V. Dimitrova, R. Mizoguchi, B. du Boulay, and AC Graesser, eds*, pages 9–16, 2009.

[5] D. C. Edelson, D. N. Gordin, and R. D. Pea. Addressing the challenges of inquiry-based learning through technology and curriculum design. *Journal of the Learning Sciences*, 8(3-4):391–450, 1999.

[6] J. Friedman, T. Hastie, and R. Tibshirani. A note on the group lasso and a sparse group lasso. *arXiv preprint arXiv:1001.0736*, 2010.

[7] A. Graesser and D. McNamara. Computational analyses of multilevel discourse comprehension. *Topics in Cognitive Science*, 3(2):371–398, 2010.

[8] J. Hancock, D. Beaver, C. Chung, J. Frazee, J. Pennebaker, A. Graesser, and Z. Cai. Social language processing: A framework for analyzing the communication of terrorists and authoritarian regimes. *Behavioral Sciences of Terrorism and Political Aggression*, 2(2):108–132, 2010.

[9] J. Hancock, L. Curry, S. Goorha, and M. Woodworth. On lying and being lied to: A linguistic analysis of deception in computer-mediated communication. *Discourse Processes*, 45(1):1–23, 2007.

[10] A. Hoerl and R. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[11] M. C. Linn. Designing the knowledge integration environment. *International Journal of Science Education*, 22(8):781–796, 2000.

[12] F. Mairesse, M. Walker, M. Mehl, and R. Moore. Using linguistic cues for the automatic recognition of personality in conversation and text. *Journal of Artificial Intelligence Research*, 30(1):457–500, 2007.

[13] T. Murray, L. Wing, B. Woolf, A. Wise, S. Wu, L. Clarke, L. Osterweil, and X. Xu. A prototype facilitators dashboard: Assessing and visualizing dialogue quality in online deliberations for education and work. In *Proceedings of the 12th International Conference on e-Learning, e-Business, Enterprise Information Systems, and e-Government*, 2013.

[14] T. Murray, B. Woolf, X. Xu, S. Shipe, S. Howard, and L. Wing. Supporting social deliberative skills in online classroom dialogues: preliminary results using automated text analysis. In *Intelligent Tutoring Systems*, pages 666–668. Springer, 2012.

[15] T. Murray, X. Xu, and B. Woolf. An exploration of text analysis methods to identify social deliberative skills. In *Proceedings of the 16th International Conference on Artificial Intelligence in Education*, 2013.

[16] A. Ng. Feature selection, l_1 vs. l_2 regularization, and rotational invariance. In *Proceedings of the twenty-first international conference on Machine learning*, page 78. ACM, 2004.

[17] K. G. Niederhoffer and J. W. Pennebaker. Linguistic style matching in social interaction. *Journal of Language and Social Psychology*, 21(4):337–360, 2002.

[18] J. Pennebaker, M. Francis, and R. Booth. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, 2001.

[19] B. J. Reiser, I. Tabak, W. A. Sandoval, B. K. Smith, F. Steinmuller, and A. J. Leone. Bguile: Strategic and conceptual scaffolds for scientific inquiry in biology classrooms. *Cognition and instruction: Twenty-five years of progress*, pages 263–305, 2001.

[20] T. Strzalkowski, G. A. Broadwell, J. Stromer-Galley, S. Shaikh, S. Taylor, and N. Webb. Modeling socio-cultural phenomena in discourse. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1038–1046. Association for Computational Linguistics, 2010.

[21] D. D. Suthers, N. Dwyer, R. Medina, and R. Vatrapu. A framework for conceptualizing, representing, and analyzing distributed interaction. *International Journal of Computer-Supported Collaborative Learning*, 5(1):5–42, 2010.

[22] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[23] Y. Wang, R. Kraut, and J. Levine. To stay or leave? the relationship of emotional and informational support to commitment in online health support groups. In *Proceedings of the ACM conference on computer-supported cooperative work*, 2011.

[24] B. Y. White, T. A. Shimoda, J. R. Frederiksen, et al. Enabling students to construct theories of collaborative inquiry and reflective learning: Computer support for metacognitive development. *International Journal of Artificial Intelligence in Education (IJAIED)*, 10:151–182, 1999.

[25] 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.

[26] L. Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *The Journal of Machine Learning Research*, 11:2543–2596, 2010.

Oral Presentations (Short Papers)

Paragraph Specific N-Gram Approaches to Automatically Assessing Essay Quality

Scott Crossley
Georgia State University
34 Peachtree Ave, Ste 1200
Atlanta, GA 30303
01+404-413-5179
scrossley@gsu.edu

Caleb DeFore
Georgia State University
34 Peachtree Ave, Ste 1200
Atlanta, GA 30303
01+404-413-5200
cdefore1@student.gsu.edu

Kris Kyle
Georgia State University
34 Peachtree Ave, Ste 1200
Atlanta, GA 30303
01+404-413-5200
kkyle3@student.gsu.edu

Jianmin Dai
Arizona State University
PO Box 872111
Tempe, AZ 85287
01+404-413-5200
Jianmin.Dai@asu.edu

Danielle S. McNamara
Arizona State University
PO Box 872111
Tempe, AZ 85287
01+404-413-5200
dsmcnamara1@gmail.com

ABSTRACT

In this paper, we describe an n-gram approach to automatically assess essay quality in student writing. Underlying this approach is the development of n-gram indices that examine rhetorical, syntactic, grammatical, and cohesion features of paragraph types (introduction, body, and conclusion paragraphs) and entire essays. For this study, we developed over 300 n-gram indices and assessed their potential to predict human ratings of essay quality. A combination of these n-gram indices explained over 30% of the variance in human ratings for essays in a training and testing corpus. The findings from this study indicate the strength of using n-gram indices to automatically assess writing quality. Such indices not only explain text-based factors that influence human judgments of essay quality, but also provide new methods for automatically assessing writing quality.

Keywords

Essay quality, computational linguistics, corpus linguistics, automatic feedback, intelligent tutoring systems.

1. INTRODUCTION

Academic success often depends on a student's writing proficiency [1]. Unfortunately, for many students, such proficiency is often difficult to attain and frequently remains elusive throughout schooling [5]. One major problem in the teaching of writing skills is that students have limited opportunities to write and receive feedback from teachers and peers. Such a problem is related to time constraints inside and outside of the classroom [5], which minimize opportunities for students and teachers to interact one on one. A potentially profitable approach to providing students with greater access to writing opportunities and ensuring that students receive feedback on their writing is through the use of automatic writing evaluation (AWE) systems that provide students with the opportunities to write essays and automatically receive feedback on the quality of their writing.

However, AWE systems often lack the sensitivity to respond to a number of features in student writing and, more specifically, to those features that relate to instructional efficacy [8]. Our goal in this study is to investigate the potential for n-gram indices related to paragraph types (i.e., introduction, body, and conclusion paragraphs) to predict human judgments of essay quality. We are interested in paragraph specific indices because developing writers need to focus on and learn strategies for building quality introduction, body, and conclusion paragraphs. If we can identify n-grams in quality essays that relate to paragraph building strategies and to human judgments of writing quality, then we can use these n-gram indices to assign automatic scores to essays. In addition, such indices may prove beneficial in providing automated formative feedback to users that directly link to instructional strategies (i.e., strategies for building stronger paragraphs).

1.1 The Writing Pal

The Writing Pal (W-Pal) is an intelligent tutoring system (ITS) that contains an AWE system in order to provide summative and formative feedback to users [4]. However, unlike strict AWE systems, W-Pal adopts a pedagogical focus by providing writing strategy instruction to users. Thus, unlike AWE systems, which focus on essay practice with some support instruction, W-Pal emphasizes strategy instruction and targeted strategy practice prior to whole-essay practice. The writing strategies cover the three phases of the writing process: prewriting, drafting, and revising. Each of the writing phases is further subdivided into instructional modules. These modules include *Freewriting* and *Planning* (prewriting); *Introduction Building*, *Body Building*, and *Conclusion Building* (drafting); and *Paraphrasing*, *Cohesion Building*, and *Revising* (revising). In W-Pal, students view lessons on each strategy, play practice games, and then write practice essays for each of the modules.

Essay writing is an essential component of W-Pal. As a result, the system includes an essay-writing interface, which allows students to compose essays. These essays are then analyzed by the W-Pal AWE system, which is used to provide automated formative and summative feedback to the participants based upon natural language input and hierarchical classification as compared to regression analyses. Such hierarchical classification affords the opportunity to provide feedback at different conceptual levels on a variety of linguistic and rhetorical features [2].

In general, the feedback in W-Pal focuses on the strategies taught in W-Pal lessons (e.g., *Conclusion Building*, P*araphrasing*, and *Cohesion Building*) and practice games and is primarily based on linguistic features reported by the AWE scoring model. For instance, if a student produces an essay that is too short, the system will provide feedback to the student suggesting the use of idea generation techniques such as those found in the freewriting module. If a student essay does not meet the paragraph threshold, the W-Pal feedback system will suggest techniques to plan and organize the essay more effectively including outlining and focusing on structural elements such as positions, arguments, and evidence (all elements taught in the instructional modules). Such feedback can be general (e.g., asking students to condense similar sentences, restructure sentences, and improve cohesion), but it can also be more specific and remind students to preview their thesis statements and arguments in the introduction paragraph, write concise topic sentences and present evidence in body paragraphs, and provide conclusion statements and restate the thesis in the concluding paragraph. The feedback system in W-Pal has proven effective in prior writing studies [6] demonstrating that essays revised using W-Pal feedback are scored significantly higher than their original drafts (as assessed by an automatic scoring algorithm).

However, in practice, the feedback provided by the W-Pal AWE system to W-Pal users can be repetitive and overly broad [6]. For instance, students often receive the same feedback from the AWE as they continue to submit drafts and revisions of papers over time. The repetition in the AWE systems is a product of the general nature of much of the feedback provided by the system and is a direct reflection of the specificity of many of the linguistic indices found in the NLP scoring algorithms used by W-Pal. These algorithms are often informed by linguistic features that, while predictive of essay quality, are not highly useful in providing feedback to users. For instance, the current algorithm includes many indices related to lexical sophistication and syntactic complexity, both of which are important indicators of essay quality [3]. However, feedback at such a fine-grain level of linguistic analysis (e.g., use more infrequent words or produce more sentences that include infinitive forms) is not very practical, helpful, or formative. As a result, much of the feedback given to W-Pal users is necessarily general in nature and could potentially hinder students' ability to utilize the feedback effectively.

2. METHODS

Our goal in this study is to develop paragraph specific n-gram indices to automatically assess the essay quality of student writers in the ITS W-Pal. The purpose of these indices is to provide potentially stronger links between the instructional modules in W-Pal and the automatic scores assigned to essays by the AWE system. If practical and specific elements of texts related to essay quality can be developed, then these elements, in turn, could also inform feedback mechanisms and potentially provide better connections between the instructional modules in W-Pal (i.e., Introduction Building, Body Building, and Conclusion Building) and formative feedback concerning these modules.

2.1 Corpus

The corpus we used to develop the n-gram indices comprised 1123 argumentative (persuasive) essays. Because our interest is in developing automated indices that are predictive across a broad range of prompts, grade levels, and temporal conditions, we selected a general corpus that contained 16 different prompts, three different grade levels (10th grade, 11th grade, and college freshman), and two different temporal conditions (essay that were untimed and essays that were written in 25-minute increments).

Not all the essays from this corpus were used to develop the n-gram indices. Only those essays that contained at least three paragraphs were selected to develop the n-gram indices. Such essays provide some evidence that the writer had produced an introduction, body, and conclusion paragraph affording the opportunity to examine paragraph specific n-grams. After removing all essays that contained fewer than 3 paragraphs, we were left with 971 essays. We used these essays to develop the n-gram indices. We used the essays in the entire corpus (N =1123) to train a regression model.

We tested the training regression model on a test set of argumentative essays that were not used in the developmental process. The essays were written by participant in a W-Pal study. They ranged in grade level from 9th to 12th (M = 10.2, SD = 1.0). Each participant wrote a pretest and a posttest essay (N = 128). The essays were written within the W-Pal essay-writing interface.

2.2 Human judgments

Each essay in the developmental corpus and the test set was scored independently by two expert raters using a 6-point rating scale developed for the Scholastic Aptitude Test. The rating scale was used to holistically assess the quality of the essays and had a minimum score of 1 and a maximum score of 6.

2.3 N-gram indices

To develop the n-gram indices, we first separated the paragraphs in all the essays that contained three or more paragraphs based on sequential positioning. All initial paragraphs were classified as introductory paragraphs; all middle paragraphs were classified as body paragraphs; and all final paragraphs were classified as conclusion paragraphs. Each paragraph was further classified as low quality (i.e., average essay score of 3 or less) or high quality (i.e., average essay score of 3.5 or greater).

The paragraphs for each position and quality rating were then analyzed using WordSmith [7] to identify key n-grams (unigrams, bigrams, and trigrams). Two expert raters then identified linguistic patterns among the key n-grams and used these linguistic patterns to classify the n-grams into linguistic groupings related to rhetorical, grammatical, syntactic, and cohesion features. N-grams were organized in the groupings based on strength of keyness. However, if a unigram was a keyword and that unigram was also included within a key bi-gram or tri-gram, the bi-gram or tri-gram was removed if it had a lower keyness value. The selected n-gram groupings are briefly discussed below.

2.3.1 Introductory Paragraphs

Twenty groupings of n-grams were identified for the introductory paragraphs. These groupings were based mostly on rhetorical features, but also include cohesion, syntactic, and grammatical features.

2.3.2 Body Paragraphs

Twenty-seven groupings of n-grams were identified for the body paragraphs. These groupings were based mostly on rhetorical features, but also include cohesion, syntactic, and grammatical features.

2.3.3 Conclusion Paragraphs

Twenty-five groupings of n-grams were identified for the conclusion paragraphs. These groupings were based mostly on rhetorical features, but also include cohesion and syntactic features.

2.4 Analyses

For each n-gram grouping, we calculated an incidence score and a proportion score for the n-grams in the grouping for each paragraph type (i.e., introduction, body, and conclusion paragraphs) and for the essay as a whole. We also combined all of the positive and all of the negative n-grams into separate indices and computed their incidence in the paragraph types and for the essays as a whole. These incidence and proportion scores became our automated indices for the subsequent regression analysis. Within each essay, all body paragraphs were pooled and treated as a single entity.

We used the essays in the entire corpus to create regression models to predict the human ratings for the essays. We first conducted correlations between the index scores and the human ratings of essay quality. We selected all those variables that demonstrated at least a small effect size ($r > .10$) and did not demonstrate strong multicollinearity with one another or with text length ($r < .899$). The model from this regression analysis was then extended to the essays in the testing corpus to examine how well the model predicted essay quality in an independent corpus.

3. Results

3.1 Multiple Regression All Essays

Of the 316 n-gram grouping indices calculated for this study, 163 of the indices demonstrated at least a small effect size with the human ratings of essay quality ($p < .001$) for all the essays in the corpus. Of these, four demonstrated strong correlations with text length and were removed. Lastly, six indices demonstrated strong multicollinearity with other indices and were removed, leaving 153 indices.

The linear regression using the selected variables yielded a significant model, $F(20, 1102) = 32.925$, $p < .001$, $r = .612$, $r^2 = .374$. Twenty variables were significant predictors in the regression. The remaining variables were not significant predictors and were either not included in the model or were removed in the steps of the model (in the case the index *Body all positive* grouping index). The regression model is presented in Table 1. We used the B weights and the constant from the regression analysis to assess the model on an independent data set (the 128 essays from the W-Pal efficacy study). The model for the test set yielded $r = .576$, $r^2 = .332$.

4. Discussion

This study demonstrates that n-gram indices related to rhetorical, grammatical, and cohesion feature of a text can be strongly predictive of human judgments of essay quality. These n-grams were calculated at the paragraph level and at the text level. The indices were tested on essays that contained as few as 1 to 2 paragraphs and on essays that contained only 3 or more paragraphs. The results of this study provide models of essay quality that could be implemented in an AWE system to provide increased accuracy of summative feedback (i.e., holistic scores). Because many of the n-gram indices are paragraph specific and many of them are related to rhetorical or cohesion patterns (as compared to syntactic and grammatical patterns), the indices are expected to provide more specific feedback to users within the W-Pal system that will be both more practical and more useful. The feedback that is based on these indices can be linked to instructional modules within the W-Pal system.

The regression model demonstrated that the combination of the 20 variables accounts for 37% of the variance in the human evaluations of overall writing quality. The most predictive indices were generally the combined n-gram indices that integrated all the

Table 4: Linear regression results for all essays

Entry	Variable Added/Removed	Correlation	R-Squared	*B*	SE	B
Entry 1	Body all positive	0.474	0.225	Removed	Removed	Removed
Entry 2	Body all positive proportion	0.510	0.260	0.766	0.174	0.199
Entry 3	Conclusion all positive	0.527	0.278	0.067	0.010	0.223
Entry 4	Conclusion all negative	0.548	0.300	-0.013	0.005	-0.087
Entry 5	Body adverbs positive proportion	0.562	0.316	0.397	0.074	0.132
Entry 6	Body connectives positive essay	0.568	0.322	0.017	0.004	0.130
Entry 7	Remove body all positive	0.566	0.321	-	-	-
Entry 8	Introduction stance negative	0.573	0.328	-0.089	0.028	-0.088
Entry 9	Conclusion all negative proportion	0.578	0.334	-0.823	0.210	-0.149
Entry 10	Introduction choice negative	0.583	0.339	-0.250	0.084	-0.079
Entry 11	Body general references positive essay	0.588	0.345	0.041	0.012	0.090
Entry 12	Body 3rd person negative	0.590	0.348	-0.019	0.007	-0.079
Entry 13	Introduction all negative proportion	0.593	0.351	-0.406	0.172	-0.112
Entry 14	Body casual positive	0.595	0.354	0.232	0.095	0.059
Entry 15	Body quantity positive essay	0.597	0.356	0.016	0.006	0.070
Entry 16	Introduction totality positive essay	0.599	0.359	-0.033	0.013	-0.075
Entry 17	Conclusion set membership positive essay	0.602	0.362	0.054	0.023	0.060
Entry 18	Body tense positive	0.604	0.364	0.041	0.017	0.063
Entry 19	Introduction 2nd person negative	0.606	0.367	0.038	0.015	0.071
Entry 20	Conclusion 1st person positive essay	0.608	0.369	-0.020	0.010	-0.052
Entry 21	Introduction conditionals negative	0.610	0.372	-0.067	0.032	-0.059
Entry 22	Introduction comparison positive essay	0.612	0.374	0.158	0.076	0.055

Notes: Estimated Constant Term is 2.563; *B* is unstandardized Beta; SE is standard error; B is standardized Beta

Note: Essay is n-gram count across the entire essay. All other n-gram counts across the paragraph types.

positive or negative n-grams for the paragraph type. For instance, positive body n-grams and positive and negative conclusion n-grams were the strongest predictors of essay quality (predicting almost 30% of the variance in the human ratings alone) followed by negative introduction n-grams. The remaining indices were more specific in nature and included six introduction n-gram indices (related to stance, choice, totality, 2nd person, conditionals, and comparison), seven body n-gram indices (related to adverbs, connectives, general reference, 3rd person, causality, quantity, and tense), and two conclusion n-gram indices (related to set membership and first person). The majority of these indices were measured at the paragraph level with 7 of the 20 indices measured across the text. Because this analysis included essays with only 1 or 2 paragraphs, we presume that conclusion n-gram indices were less predictive insomuch as many essays would not contain a second or third paragraph that would act as a conclusion.

From a linguistic perspective, this study has demonstrated that rhetorical features of paragraphs are important indicators of essay quality. The majority of the n-gram indices that loaded into our regression models were rhetorical in nature. For instance, the use of adverbs such as *yet, unfortunately,* and *completely* are important indicators of writing proficiency demonstrating that better writers use a greater number of such adverbs. High quality essays also contain fewer negative stance n-grams in the introduction (e.g., *I think, know, feel that*). Good writers also use more general reference terms such as *these* and *those*, indicating that referencing previous noun phrases is an important indicator of writing quality. Such an index may also relate to the cohesive properties of the text and, in support, this study also reports that other cohesive features loaded into our regression models. For instance, positive n-gram connectives (i.e., *however, and*) found in the body are significant predictors. Unlike rhetorical and cohesive n-gram indices, no syntactic indices loaded into our regression model and only one grammatical n-gram index loaded (positive body tense n-grams). Such a finding does not diminish the importance of syntactic and grammatical features in essay writing, but rather demonstrates that an n-gram approach likely does not capture the complexity needed to assess such features.

We envision that these n-gram indices could be used to provide formative feedback to users in an ITS. For instance, these n-gram indices directly overlap with instruction modules in W-Pal (i.e., introduction building, body building, and paragraph building) and would thus link with the writing strategies with which users become familiar during training. The indices are also much more paragraph specific than current feedback algorithms in W-Pal, which focus on general feedback concerning relevance to topic, essay structure, paragraph structure, and revising strategies. For example, the current feedback reminds users to attend to structural elements in paragraphs such as positions, arguments, and evidence. However, the feedback algorithms do not provide specific linguistic features to which to attend. We envision that the n-gram indices discussed in this study could provide useful and specific formative feedback to assist in student essay revision. For instance, users could be given specific feedback about their use of adverb, general reference, connective, quantity, and tense n-grams in their body paragraphs. Users could also receive direct and specific feedback on their use of set membership words and 1st persons in their conclusion. This feedback would be based on concrete linguistic features in the text and would provide rhetorical, cohesion, and grammatical information to the user that could be exploited during the revision process.

5. Conclusion

While strongly predictive, the n-gram indices investigated here should be examined in conjunction with more traditional linguistic indices that have demonstrated predictive power in explaining essay quality (i.e., lexical, syntactic, and cohesive features of text; [3]). Such an analysis would assess how predictive the n-gram indices are when combined with other variables. More importantly, the indices should be tested to examine the degree to which they are able to provide more direct and specific formative feedback and the effects of such feedback on essay revision and quality.

6. ACKNOWLEDGMENTS

This research was supported in part by the Institute for Education Sciences (IES R305A080589 and IES R305G20018-02). Ideas expressed in this material are those of the authors and do not necessarily reflect the views of the IES.

7. REFERENCES

[1] Kellogg, R. and Raulerson, B. 2007. Improving the writing skills of college students. *Psychonomic Bulletin and Review. 14*, 237-242.

[2] Crossley, S., Roscoe, R., and McNamara, D. (in press). Using natural language processing algorithms to detect changes in student writing in an intelligent tutoring system. Manuscript submitted to the *26th International Florida Artificial Intelligence Research Society Conference*.

[3] McNamara, D., Crossley, S., and Roscoe, R. 2013. Natural language processing in an intelligent writing strategy tutoring system. *Behavioral Research Methods, Instruments and Computers.* Advance online publication.

[4] McNamara, D., Raine, R., Roscoe, R., Crossley, S., Jackson, G., Dai, J., Cai, Z., Renner, A., Brandon, R., Weston, J., Dempsey, K., Carney, D., Sullivan, S., Kim, L., Rus, V., Floyd, R., McCarthy, P., and Graesser, A. 2012. The Writing-Pal: Natural language algorithms to support intelligent tutoring on writing strategies. In P. McCarthy & C. Boonthum-Denecke (Eds.), *Applied natural language processing and content analysis: Identification, investigation, and resolution* (pp. 298-311). Hershey, P.A.: IGI Global.

[5] National Commission on Writing. 2003. *The Neglected "R."* College Entrance Examination Board, New York.

[6] Roscoe, R., Kugler, D., Crossley, S., Weston, J., and McNamara, D. S. 2012. Developing pedagogically-guided threshold algorithms for intelligent automated essay feedback. In P. McCarthy & G. Youngblood (Eds.), *Proceedings of the 25th International Florida Artificial Intelligence Research Society Conference* (pp. 466-471). Menlo Park, CA: The AAAI Press.

[7] Scott, M. 2008. WordSmith Tools version 5, Liverpool: Lexical Analysis Software.

[8] Shermis, M.D., Burstein, J.C. and Bliss, L. 2004. The impact of automated essay scoring on high stakes writing assessments. *Paper presented at the annual meeting of the National Council on Measurement in Education*, April 2004, San Diego, CA

Degeneracy in Student Modeling with Dynamic Bayesian Networks in Intelligent Edu-Games

Alireza Davoodi
Department of Computer Science
The University of British Columbia
2366 Main Mall, Vancouver, BC, V6T1Z4, Canada
+1 (604) 8225108
davoodi@cs.ubc.ca

Cristina Conati
Department of Computer Science
The University of British Columbia
2366 Main Mall, Vancouver, BC, V6T1Z4, Canada
+1 (604) 8224632
conati@cs.ubc.ca

ABSTRACT

This paper investigates the issue of degeneracy in student modeling with Dynamic Bayesian Network in Prime Climb, an intelligent educational game for practicing number factorization. We discuss that maximizing the common measure of predictive accuracy (i.e. end accuracy) of the student model may not necessarily ensure trusted assessment of learning in the student and that, it could result in implausible inferences about the student. An approach which bounds the parameters of the model has been applied to avoid the issue of degeneracy in the student model to a high extent without significantly diminishing the predictive accuracy of the student model.

Keywords

Educational Games, Student Model, Dynamic Bayesian Networks, Predictive Accuracy, Model Degeneracy

1. INTRODUCTION

Assisting individuals to acquire desired knowledge and skills while engaging in a game, distinguishes digital educational games (henceforth edu-games) from traditional video games [1, 2]. Edu-games integrate game design methods with pedagogical techniques in order to more appropriately address the learning needs of the new generation, which highly regards “doing rather than knowing”. Adaptive edu-games as a sub-category of edu-games leverage a user model to track the evolution of knowledge in the students and support tailored interactions with the player and have been proposed as an alternative solution for the one-size-fits-all approach used in designing non-adaptive edu-games [2].

Prime Climb (PC) is an adaptive edu-game for students in grades 5 and 6 to practice number factorization concepts. It provides a test-bed for conducting research on adaptation in edu-games. Prime Climb uses Dynamic Bayesian Network (DBN) to construct a student model which maintains and provides an assessment of student’s knowledge on target skills (number factorization skills) during and at the end of the interaction. The model’s assessment of the student’s knowledge on the desired skills during the game play is leveraged by an intelligent pedagogical agent which applies a heuristic strategy to provide the student with personalized supports in the form of varying types of hints [3]. In addition, the model’s evaluation of the student’s knowledge on target skills at the end of the game, provides predictions of the student’s performance on related problems outside the game environment (for instance on a post test). Therefore, an accurate student model is the main component of a system which adapts to users and any issue which could decay the efficiency of the model should be appropriately avoided and resolved.

While most of the work on user modeling in educational systems has been on optimizing the predictive accuracy (predicting student’s performance on opportunities to practice skills) of the student models [5], there is limited work on educational implications and conceptual meaning imposed by the student model resulted from the predictive accuracy optimization process. This paper investigates the issue of degeneracy in the student model in PC and how it impacts the modeling. The issue of degeneracy is defined as a situation in which the parameters of a parametric student model are estimated such that the model has the highest performance (is at its global maximum given the performance and limitations of the optimization method) with respect to some standard measures of accuracy, yet it violates the conceptual assumptions (explained later in more details) underlying the process being modeled [6].

2. RELATED WORK

Difficulties in inferring student knowledge have been recently studied [4, 6, 8, 9, 10, 11] in an approach to educational user modeling called Knowledge Tracing (KT) [7]. Knowledge Tracing assumes a two-state learning model in which a skill is either in the learned or unlearned state. An unlearned skill might change to the state of learned at each opportunity the student practices the skill. In KT, it is also assumed that the student’s correct/incorrect performance in applying a skill is the direct consequence of the skill being in the learned/unlearned state; yet there is always the possibility of a student correctly applying a rule without knowing the corresponding skill. This is referred to as probability of guessing. Similarly, the likelihood of a student showing an incorrect performance on applying a rule while knowing the underlying skill is called the probability of slipping. One issue with KT, called Identifiability was addressed by Beck [4]. The issue of Identifiability refers to the existence of multiple equally good mappings from observable student’s performance to her corresponding latent level of knowledge while each mapping claims differently about the student performance and knowledge. To address this issues, Beck introduced the Dirichlet prior approach [4] in which a Dirichlet probability distribution is defined over the model’s parameters in a KT to bias the estimation of the model parameters toward the mean of the distribution. The Dirichlet prior approach was then extended and the Multiple Dirichlet Prior approach [8] and Weighted Dirichlet Prior [9] were proposed to further address the Identifiability issue in KT. Backer et al. [6] discussed that the Knowledge Tracing models may also suffer from the problem of degeneracy. A KT model is degenerate if it updates the probability of a student knowing some skills in such a way that it violates the conceptual assumptions (such as a student being more likely to make a correct answer if she does not have the corresponding knowledge than she does) underlying the process being modeled. Generally when the probability of slipping and guessing in KT are greater than 0.5 the model is said to be theoretically degenerate. It was also shown that the Dirichlet prior KT model (which was proposed to address the Identifiability problem in KT) also suffers from the degeneracy problem [4]. One straightforward approach to avoiding theoretical degeneration is bounding the Knowledge Tracing model parameters (probability of guessing and slipping) to take a value less than 0.5. This approach is called Bounded KT [4]. A KT model could be also empirically degenerate even if not theoretically degenerate. Two tests were also introduced to investigate empirical degeneracy in KT [4]. Baker et al. [7, 11, 12] also proposed an approach called Contextual Guess and Slip in Knowledge Tracing for contextually estimating the probabilities of guessing and slipping and showed that such model is less degenerated than standard KT which allows any value between 0 and 1 for guessing and slipping.

This paper builds on the previous works on issues with Knowledge Tracing, to investigate the issue of degeneracy in a student model which uses a Dynamic Bayesian Network and a causal structure to infer about the student's knowledge on skills in an adaptive edu-game called Prime Climb. The issues of degeneracy has been studied in Knowledge Tracing models which assume that learning different skills is independent from each other while in PC, based on guidance from a math expert, it is assumed that the factorization skills are not independent from each other. Moreover, In KT, at each time, the student has an opportunity to practice a single skill, while in Prime Climb, at least three skills are practiced simultaneously and consequently there are other model's parameters than probability of guessing and slipping in the student model in PC.

3. PRELIMINARIES/BACKGROUND

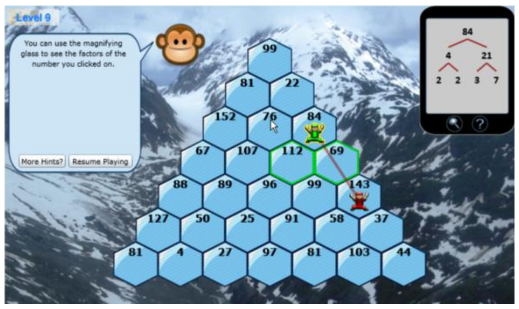


Figure 1: Prime Climb Edu-game

3.1 Prime Climb Edu-game:

In Prime Climb (Figure 1), the player and her partner climb a series of mountains (11 mountains) of numbers by pairing up the numbers which do not share a common factor. The main interaction of a player with Prime Climb consists of making a movement from a location on a mountain of numbers to another location on the mountain until she reaches the top of the mountain. Therefore at each movement, the student practices at least 3 skills: 1) Factorization of the number the player moves to. 2) Factorization of the number the partner is on and 3) The concept of common factor between the 2 numbers.

3.2 Student Model in Prime Climb:

Prime Climb is equipped with 11 probabilistic student models (one for each mountain) which use Dynamic Bayesian Network to model the evolution of student's factorization knowledge during the period of time that she interacts with Prime Climb. To this end the student model consists of time slices representing relevant temporal states in the process being modeled. Each time slice is created once a student makes a movement (climbs a mountain). The smallest student model in PC consists of 23 binary nodes (random variables) and the largest one contains 131 nodes.

<u>PC's Student Model Nodes</u>: In PC, each student model contains several binary nodes [5] such as:

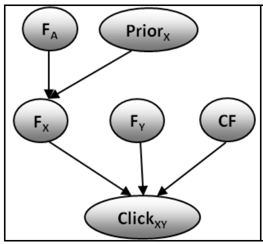
Factorization Nodes (F_X): Each factorization node, F_X, is a binary random variable which represents the probability that the student has mastered the factorization skill of number X.

Common Factor Node (CF): There is only one CF node. It is a binary random variable representing the probability that student has mastered the concept of common factor between numbers.

Prior_X Node: There is one Prior node for each none-root factorization node in the model. It shows the prior probability that the student knows the factorization of the number X to its factors.

Click Nodes (Click_{XY}): Once the player makes a move (i.e. moves to number X while the partner is on Y) a Click node is temporarily added as a child of the three random nodes F_X, F_Y and CF to make a causal structure. Therefore, these three nodes are conditionally dependent to each other given evidence on the Click node. Such causal structure allows apportion of blame for wrong movements [5]. Table 1 and Table 2 show the Conditional Probability Table (CPT) of the F_X and Click nodes respectively.

Table 1: Model Structure and CPT of F_X Factorization Node



F_A	Prior_X	$P(F_X = \text{Known})$
Known	Known	1
Known	Unknown	$Max \times \frac{P_k}{P}$
Unknown	Known	1
Unknown	Unknown	0

Table 2: CPT of Click (K: Known, U: Unknown, C: Correct)

	$F_X = K$		$F_X = U$		
	F_Y=K	F_Y=U	F_Y=K	F_Y=U	
$P(\text{Click}_{XY}=C)$	1-Slip	Edu-Guess	Edu-Guess	Guess	CF=K
	Guess	Guess	Guess	Guess	CF=U

<u>Model's Parameters in Prime Climb</u>: The parameters (guess, edu-guess, slip and max) are called "model's parameters" in the student model in Prime Climb:

Slip: The probability of making a wrong action on a problem step when the student has the corresponding knowledge.

Guess: The probability of making a correct action on a problem step when the student does not have the corresponding skill.

Edu-Guess: The probability of a student making a correct answer while the student does not completely master the required knowledge for making such correct action.

Max: A coefficient in the formula ($Max \times \frac{P_k}{P}$) used to calculate the probability of a student making a correct move proportional to number of its known parents. (P is number of parents of F_X and P_K is number of those parents which are known.

<u>End Accuracy of Student Model</u>: The student model in PC is evaluated based on the end accuracy (=predictive accuracy) of the model. The end accuracy is defined as the model's performance in accurate assessment of the student's factorization knowledge about some sample numbers appearing on a post-test at the end of the game and calculated using the following formula [13]:

$$End\ Accuracy = \frac{Sensitivity + Specificity}{2}$$

Before starting the game, the factorization and common factor nodes in the student model are initialized with prior probabilities that the student knows the number factorization and common factor concept. In Prime Climb, three types of prior probability are used which are defined as following:

Generic: The prior probability that a student knows number factorization and common factor skills is set to 0.5.
Population: The prior probability is calculated based on scores of a group of students on a pre-test which examines the knowledge of students on specific factorization skills.
User-specific: The prior probability is specific to each student based on her performance on the pre-test. If a student has correctly responded a number factorization question in the pre-test, the probability that the student knows the corresponding factorization skill is set to 0.9 otherwise it is set to 0.1.
Plausibility of Parameters in PC's Student Model: While the end accuracy is used to evaluate PC's student model, the model's parameters can also be directly evaluated based of "plausibility" criteria. One criterion is the impact of model's parameters (guess, edu-guess, slip and max) on performance of the adaptive interventions (hints) mechanism in PC. The performance of hinting mechanism in PC is calculated based on average of two F-Measures [12]: 1) Positive F-Measure: calculated using precision and recall of the hinting mechanism in identifying correct time points for providing hints and 2) Negative F-Measure: calculated using precision and recall of the hinting mechanism in identifying the time points in which hints should not be given to the users. According to such criteria, a set of model's parameters improving performance of the hinting mechanism while providing reasonable number of hints during game play is more plausible. For instance if a student makes 200 movements in total during the game, it is not plausible to receive over 100 hints (One hint for every two movement on average). Notice that the value of the model's parameters directly affects the hinting mechanism in PC.

4. DEGENERACY IN STUDENT MODEL

Student Model Optimization: The Prime Climb's original student model allows any value between 0 and 1 for the model's parameters (slip, guess, edu-guess and max). The values for the parameters are estimated such that the model's end accuracy is maximized. To this end, an exhaustive search procedure is applied which examines values between 0 and 1 in interval of 0.1 for each parameters and eventually selects the parameters combination maximizing the end accuracy. To this end, a Leave-One-Out Cross Validation approach was applied across 43 students who played Prime Climb. The optimal set of parameters and the mean end accuracy across the test folds for each prior probability type were computed and summarized in Table 3.

Table 3: Estimated Parameters and End Accuracy

Prior Probability	Guess	Edu-Guess	Max	Slip	End Accuracy (M/SD)
Population	0.5	0.3	0.2	0.4	0.77/0.14
Generic	0.2	0.6	0.8	0.6	0.70/0.15
User-specific	0.6	0.1	0.6	0.6	0.72/0.20

Degeneracy in Student Model: The degeneracy in student modeling in Prime Climb is defined as violation of the conceptual assumptions behind modeling of a student's knowledge on factorization skills during interaction with the game. The conceptual assumptions in PC student model are as following:

1) Correct evidence (action) on a skill must not decrease the probability of the student knowing the skill.
2) An incorrect action on a skill must not increase the probability of the student knowing the skill.

Any pattern in the student model violating the aforementioned assumptions is marked as model degeneration. We defined two tests to investigate model degeneracy in Prime Climb's student model. If the student model fails either of these two tests, the model is said to be degenerated:

Test 1 of degeneration in Prime Climb: If a student makes a correct/incorrect action on an opportunity to practice a skill, the probability of the student knowing the skill should not be less/greater than the probability of knowing the skill before making the action on the skill. Mathematically the following cases show failures in Test 1:

$$P(F_X = Known\ |Click = Correct) < P(F_X = Known)$$
$$P(F_Y = Known\ |Click = Correct) < P(F_Y = Known)$$
$$P(F_X = Known\ |Click = Wrong) > P(F_X = Known)$$
$$P(F_Y = Known\ |Click = Wrong) > P(F_Y = Known)$$
$$P(CF = Known\ |Click = Correct) < P(CF = Known)$$
$$P(CF = Known\ |Click = Wrong) > P(CF = Known)$$

Test 2 of degeneration in Prime Climb: Assume a dependency relationship between two skills S_1 and S_2 such that knowledge on S_1 implies knowledge on S_2 with a certain probability. If a student performs correctly/incorrectly on an opportunity to practice skill S_1, the probability that the student knows skill S_2 should not be less/greater than its values before making the action.

The original student model was checked for degeneracy using the Tests 1 and 2 of degeneration. Table 4 summarizes the mean number of failures across the 43 students who played PC.

Table 4: Failures in Test 1 and Test 2 in PC's Original Model

Prior Probabilities	Failures in Test 1 (M/SD)	Failures in Test 2 (M/SD)
Population	268.91/64.26	1.71/2.85
Generic	101.84/28.73	258.35/80.48
User-specific	339.17/75.11	138.86/62.04

As shown in Table 4, the original student model in Prime Climb suffers from degeneracy issue. Theoretically, based on the CPT of the Click node (See Table 2), it can be concluded that the following conditions (in Table 5) might cause specific patterns of degeneracy in the Prime Climb's student model.

Table 5: Conditions and Patterns of Degeneracy in PC

Conditions	Related Patterns of Degeneracy
Eduguess< Guess	$P(CF = Known\ |Click = Correct) < P(CF = Known)$ $P(CF = Known\ |Click = Wrong) > P(CF = Known)$
1-Slip < Guess	
1-Slip < Eduguess	$P(F_Y = Known\ |Click = Correct) < P(F_Y = Known)$ $P(F_Y = Known\ |Click = Wrong) > P(F_Y = Known)$ $P(F_X = Knwon\ |Click = Correct) < P(F_X = Known)$ $P(F_X = Known\ |Click = Wrong) > P(F_X = Known)$

Given the estimated parameters for the original presented in Table 3 and the degeneracy conditions in Table 4, different patterns of degeneracy can be observed in the Prime Climb's original model.

5. BOUNDED STUDENT MODEL

To alleviate the issue of degeneracy, the model's parameters are bounded to take values from outside the subspaces (conditions in Table 5) that cause specific patterns of degeneracy in PC. Such model is called Prime Climb's Bounded student model. Similar to the PC's original model, an exhaustive search approach is used to find a set of bounded model's parameters which maximizes the model's end accuracy. In this study we allow values greater than 0.5 for the model's parameters. The estimated parameters and the end accuracy of the bounded student model are shown in Table 6.

Table 6: The Estimated Parameters and End Accuracy

Prior Probability	Guess	Edu-Guess	Max	Slip	End Accuracy (M/SD)
Population	0.7	0.7	0.2	0.4	0.76/0.15
Generic	0.5	0.6	0.4	0.8	0.68/0.15
User-specific	0.3	0.3	0.6	0.8	0.70/0.18

<u>Comparison of the Models' Accuracy</u>: The results of a paired t-test showed no statistically significant difference between the end accuracy and AUC (Area under the ROC curve) of the original and bounded models in none of the prior probability type.

Table 7: AUC of the student models

AUC	Prior probability Types		
	Population	Generic	User-specific
Original	0.7345	0.6762	0.7860
Bounded	0.7375	0.6643	0.7449

<u>Comparison of Models' Degeneracy</u>: A paired t-test is used to compare the two models based on the average number of failures in the two tests of degeneracy. Table 8 shows the results. In all cases, the bounded model resulted in significantly lower number of failures in the both tests of degeneration and the p-value is less than 0.01 (except where indicated by *).

Table 8: Comparison of Failures in Degeneration Tests

Tests	Models	Population (Mean/SD)	Generic (Mean/SD)	User-specific (Mean/SD)
Test 1	Bounded	9.17 / 7.45	8.8/7.29	10.24/10.96
	Original	268.91 / 64.26	101.84/28.73	339.17/75.11
Test 2	Bounded	1.44 / 2.0	0.24/0.48	0.53/1.2
	Original	1.71 / 2.86*	258.35/80.48	138.86/62.04

<u>Comparison of the Models' Parameters Plausibility</u>: The plausibility of the estimated parameters in original and bounded models was compared based on performance (measured by F-Measure as described before) of the hinting method in PC. To this end, the performance of the hinting mechanism as well as average number of given hints are calculated. A paired t-test is used to compare the hinting procedure performance and number of hints across 43 students. The following tables show the comparison results. In all comparisons, the p-value is less than 0.01.

Table 9: Comparison of F-Measures of Hinting Mechanism

F-Measure	Population (Mean/SD)	Generic (Mean/SD)	User-specific (Mean/SD)
Original	0.24 / 0.2	0.3 / 0.24	1 / 0
Bounded	0.29 / 0.22	0.55 / 0.32	0.95 / 0.08

Table 10: Comparison of number of adaptive hints

#Hints	Population (Mean/SD)	Generic (Mean/SD)	User-specific (Mean/SD)
Original	112.5 / 56.62	82.95 / 27.37	139 / 39.36
Bounded	55.2 / 19.84	48.53 / 19.1	42 / 18.24

As shown in Table 10, the results of a paired t-test show that the bounded models resulted in a significantly lower number of hints ($p<0.01$ in all cases) while significantly higher performance for the hinting mechanism (except for the student model with user-specific prior probability type). Note that on average each student makes 164.5 movements while playing PC. Based on the results, the hinting mechanism provides 2, 1.7 and 3.3 times more hints in the original model than the bounded with population, generic and user-specific prior probability types respectively. This shows that in general, the bounded model provides more plausible model's parameters than the original student model.

6. CONCLUSIONS/FUTURE WORK

This paper discussed that optimizing the student model in Prime Climb does not ensure a trusted student modeling because the model might be degenerated. The issue of degeneracy and sources and patterns of degeneracy were described and one approach to addressing this issue called, bounded model was also introduced and compared with the original student model. It was shown that the bounded model has a comparable accuracy with the original model while it contains significantly fewer cases of degeneracy. The estimated parameters in the bounded model were also more plausible than the parameters in the original model. In the current bounded model, the model's parameters are estimated the same across all students. As for future work, we will consider more personalized model's parameters in bounded model to account for individual differences between users.

7. REFERENCES

[1] de Castell, S. & Jenson, J., 2007, Digital Games for Education: When Meanings Play. Intermedialities, 9, 45-54.

[2] Conati, C. and M. Klawe, 2002, Socially Intelligent Agents in Educational Games. In Socially Intelligent Agents - Creating Relationships with Computers and Robots. K. Dautenhahn, et al., Editors, Kluwer Academic Publishers.

[3] Conati C and Manske M.: Evaluating Adaptive Feedback in an Educational Computer Game, IVA 2009, 146-158

[4] Beck, J.E., 2007, Difficulties in inferring student knowledge from observations (and why you should care). Educational Data Mining

[5] Manske, M., Conati, C., Modelling Learning in an Educational Game. AIED 2005: 411-418

[6] Baker, R. S.J.d., Corbett, A.T., Aleven, V., 2008, More Accurate Student Modeling Through Contextual Estimation of Slip and Guess Probabilities in Bayesian Knowledge Tracing. Human-Computer Interaction Institute. Paper 6. http://repository.cmu.edu/hcii/6

[7] Corbett, A.T. and Anderson, J. R., 1995, Knowledge tracing: Modeling the acquisition of procedural knowledge, User Modeling and User Adapted Interaction, Volume 4, Number 4, 253-278

[8] Gong, Y., Beck. J. E., Ruiz, C., 2012, Modeling Multiple Distributions of Student Performances to Improve Predictive Accuracy. UMAP 2012: 102-113

[9] Rai, D., Gong, Y., Beck, J., 2009, Using Dirichlet priors to improve model parameter plausibility. EDM 2009: 141-150

[10] Baker, R. S.J.d., Corbett, A.T., Aleven, V., 2008, Improving Contextual Models of Guessing and Slipping with a Trucated Training Set. EDM 2008: 67-76

[11] Baker, R. S.J.d., Corbett, A.T., Gowda, S. M., Wagner, A.Z., MacLaren, B. A., Kauffman, L. R., Mitchell, A. P., Giguere, S., 2010, Contextual Slip and Prediction of Student Performance after Use of an Intelligent Tutor. UMAP 2010: 52-63

[12] Powers, David M W, 2011. "Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation". Journal of Machine Learning Technologies **2** (1): 37–63.

[13] Altman DG, Bland JM (1994). "Diagnostic tests. 1: Sensitivity and specificity". *BMJ* 308 (6943): 1552

Clustering and Visualizing Study State Sequences

Michel C. Desmarais
Polytechnique Montréal
michel.desmarais@polymtl.ca

François Lemieux
Polytechnique Montréal
francois.lemieux@polymtl.ca

ABSTRACT

This paper investigates means to visualize and classify patterns of study of a college math learning environment. We gathered logs of learner interactions with a drill and practice learning environment in college mathematics. Detailed logs of student usage was gathered for four months. Student activity sessions are extracted from the logs and clustered in three categories. Visualization of clusters allows a clear and intuitive interpretation of the activities within the clustered sessions. The three clusters are further used to visualize the global activity of the 69 participating students, which would otherwise be difficult to grasp without such means to extract patterns of use. The results reveal highly distinct patterns. In particular, they reveal an unexpected and substantial amount of navigation through exercises and notes without students actually trying the exercises themselves. This combination of clustering and visualization can prove useful to learning environments designers who need to better understand how their application software are used in practice by learners.

1. INTRODUCTION

The human eye is a powerful means to extract patterns from data, given the proper visualization tools. We borrow visualization techniques from social sciences to display state sequence diagrams and demonstrate their use in Educational Data Mining (EDM).

We combine the visualization tools with clustering techniques to better understand the patterns of use of a learning environment. Detailed user sessions of interaction with a drill and practice environment for college math are encoded as sequences of activities.

2. VISUALIZATION OF TEMPORAL SEQUENCES

Visualization of student interactions is one of the core topics of educational data mining and a few studies have introduced innovative visualization tools in the last decade [11; 10; 9].

This paper focuses on the visualization of temporal sequences of student activity. This type of data can be represented in two different forms:

(1) *Event sequences.* A given event occurs at a specific time. Events can be considered as having no duration, and the focus is more on the transition from one event to another. A majority of studies in EDM have studied such transition data as we see below.

(2) *State sequences.* Each student is engaged in a given activity, or state, for a specific time duration. For example, a student can be consulting notes, involved in problem solving, reading a scaffolded hint, etc.

Student state changes are triggered by events, as a transition from one state to another occurs after some event. Therefore, the two concepts are tightly related. But the type of analysis differ whether we focus on state sequences or the event sequences. Event sequences will often be represented as graphs, emphasizing the path between events and transition frequencies, whereas state sequences are often represented as a flow of states (activities) on a time line. The emphasis for state sequences is on the types and the duration of activities over a given period, instead of transition between states.

2.1 Event Sequences

Event sequences have been studied by a few researchers who were aiming to find patterns of student learning. Beal and Cohen have surveyed a number of these techniques [2]. In one of their study, they used Hidden Markov Models (HMM) to predict sequences of answer types of a math tutor. They showed that modeling the level of student engagement as a hidden factor of the HMM helped improve predictions [3]. Jeong et al. also showed the use of HMM to characterize student behavior in a "learning by teaching" paradigm [7].

Hadwin et al. [6] have also investigated activity event sequences to find patterns of study behavior. They used transition graph and graph theoretic statistics to characterize the student study patterns.

Köck and Paramythis [8] did an extensive study of event sequence analysis over the ANDES Tutor data [12]. They combined k-means clustering techniques with Discrete Markov Models to successfully extract and identify student problem-solving styles.

2.2 State Sequences

Instead of emphasizing the transition between states, temporal state sequences of student activity emphasize the time line perspective and the duration of activities. This type of representation has been used in sociology [1], but has not received much attention in Educational Data Mining.

The time line perspective representation is well suited for visualization of activities as a function of time. Each sequence of activity is represented as a single horizontal bar, and each activity is displayed by a segment on the bar with a given color. We will refer to this as *state sequences*. This type of visualization is shown in figure 1. Both types of diagrams will be explained in more details later. In the current study, we use the TraMineR package [5] available on the R statistical analysis platform.

3. LEARNER ACTIVITY SEQUENCES AND THEIR CLUSTERING

A prerequisite to effective visualization is that the data must be arranged in a meaningful and organized manner such that the patterns emerge naturally to the human eye.

Our solution to effective visualization of usage patterns by a large number of student is two staged: (1) we use a clustering technique to extract the main patterns and use state sequence visualization to characterize them intuitively (section 3.5), and (2) we display student usage as a function of these patterns (section 3.6). Let us first explain sequence data, the clustering algorithm and the results obtained, and finally show the global student picture.

3.1 The Drill and Practice Learning Environment

The web base drill and practice learning environment records detailed logs of user interactions with the application in the form of events that are processed to visualize activities.

The application was made available for four months in the summer of 2012 to newly enrolled university students who wanted to refresh, or enhance their knowledge of prerequisite mathematical concepts for all engineering programs at Polytechnique Montreal. The decision to do the exercises were entirely left at the discretion of the students and no marks or bonus were given for those who used the application. Furthermore, the exercises are presented with a button next to each of them, that, once clicked, immediately shows the right answer and lets the student assess for himself or herself if his/her answer is right. If deemed right, the exercise is marked as completed in the Results section of the application. The student can thereby assess his progress in terms of the proportion of exercises completed.

A total of 1030 exercises are available and they span over 10 mathematics topics such as basic algebra, logarithms and exponentials, trigonometry, calculus, and linear algebra. The notes section represents the equivalent of about 150 pages of a textbook.

3.2 Event and Activity Data

Detailed log data, such as answers to exercises, clicking on a hyperlink, and even scrolling with the mouse is logged as events with a time stamp. This allows to record almost as much as can be recorded on a web browser to assess the level of activity of a user.

These events can be considered as having no time duration, and need to be transformed into sequences of student states, which involves some pre-processing. This process involves the creation of pause events if no event occurs for more than 5 minutes, such as scrolling or navigation. An exception to this rule is if the following event is an answer to an exercise, since problem solving during exercises can last longer than 5 minutes. It also involves pre-processing to ensure that answers to exercises will not go unseen and allow for the distinction between active answer periods, and time spent on problem solving.

Once these adjustments are made to the event sequence, the next step consists in projecting the sequence of events over a time line. A time line represents a series of equal segments of time for which a state is given. If the granularity of the time line is 15 sec., for example, then the state at each segment of the time line is set to the most recent event in this 15 sec. interval. This may result in events that never get displayed if the time segment is longer then the time between events.

3.3 From Events to Activity States Sequences

As explained above, the projection of the events sequence to a state sequence is based on labelling a time segment based on the last event of the current sequence, or the last event of previous sequences if none occurred during the time segment.

A student sequence of activities is broken down per session. A session is defined as all activities contiguous in time that are no more than 1.5 hour apart. A time difference between events of over 1.5 hour creates a new sequence (session).

Seven types of activities are derived from the log of user events:

1. **Answer Ex.**: A click over the answer button ("Answer ex." event) occurred during the time step and is represented as the activity of that time step.
2. **Nav. Exerc.**: Student is browsing through the exercises but has not answered an exercise during the time segment.
3. **Nav. Notes**: Student is browsing through sections of the notes module.
4. **Pause**: No event occurred in the last 5 minutes. Pauses can last up to 1.5 hour (the maximum time after which a new session is created).
5. **Prblm. solv.**: Last event was an answer to an exercise, but no event was recorded during the time step and therefore we assume the student is in problem solving mode over the exercises shown on the page.
6. **Result**: Browsing a page that summarizes statistics on the exercises completed and the number of remaining exercises per main section.
7. **Start**: Activity on the login page.

For the purpose of this study, we ignore sessions that are shorter than 5 minutes. This leaves a total of 454 sessions of activity sequences by 69 students. Mean and median session duration are respectively 42 and 20 minutes with a minimum duration of 5 minutes and a maximum duration of 6.3 hours. Mean and median number of sessions per student are respectively 2 and 6.5, with a minimum of 1 and a maximum of 93 sessions. 24 students completed no exercise whereas one student completed all 1030 of them. Median number of attempts to exercises is 12 and mean is 174. An exercise can be attempted more than once if the student answers that he did not get the answer right. In this case, the exercise is shown with a "validate my answer" button. If the student answers his response is correct, the solution is displayed in place of the button.

3.4 Clustering Algorithm

As mentioned, there are 454 sessions by 69 students. To build an synthetic view of their usage patterns, we first extract types of state sequences from the data with a clustering algorithm.

Based on the well known Levenshtein distance, an agglomerative (bottom up) hierarchical method is used to aggregate the most similar sequences (Ward method of the R `cluster` package [4]). In short, the algorithm consists in pairing the most similar individual sequences, and in further pairing groups of sequences, ensuring that the mean distance between two clusters is minimal at every level.

We chose to create 3 clusters based on exploratory visualization of the different results.

3.5 Clusters of Activity Sequences

The 3 clusters of state sequences are shown in figure 1. Each horizontal line in a graph shows an activity state sequence (session). The time segments are 15 sec. each. The figures show 240 segments, which corresponds to a total of one hour. Longer sessions are truncated. There are 7 types of activities as described in section 3.2. For the sake of visibility, we have randomly sampled only 30 of the sequences of each type. The actual numbers of sessions per type are shown below.

The clustered sequences obtained can be characterized as follow:

- **Type 1 (N=135)**: Exploratory behavior. The students engage in a mixture of browsing through exercises and notes.
- **Type 2 (N=196)**: Short sessions comprising a variety of behaviors. Shorter sessions are aggregated due to the fact that the Levenshtein distance penalizes deletion and addition of sequence elements.
- **Type 3 (N=123)**: Exercise intensive sessions.

3.6 Activities per Student

Figure 2's top diagram shows the proportion of session types for each of the 69 students. The students are ordered according to the time they spent with the application. The y axis corresponds to the three session types. Darker cells indicate the dominant session(s) for that student.

The bottom diagram is a frequency plot of the number of sessions per student.

We can see that the more engaged students, defined as those who spend the most time with the applications, have sessions of all types (students 60 to 69). However, we notice that students 40 to 55, who have a usage time over the median, are not engaged in the same manner as the ones who had a predominance of type 3 sessions. They do not engage much in doing exercises, if at all, but do spend a substantial amount of time browsing through the application content.

Also noteworthy is that the student with very short sessions (type 2) have diverse patterns of activities. Most combine browsing and answer exercises, and browsing through notes, in a relatively short period of time compared to the others. They are essentially exploring the application. Unsurprisingly, almost all of the students who have only a single session of interaction with the learning environment have this type of behavior.

4. DISCUSSION

This paper introduces a technique to visualize student learning activities with a self-regulated drill and practice environment, and reports on an experiment that combines the visualization method with clustering and classification techniques to obtain a global view of student activities.

The clustering clearly reveals that very distinct study patterns emerge per session. Without the visualization of clusters as state diagrams, cluster interpretation would remain a difficult task. This is particularly the case because a single student often adopts different patterns of study sessions. Therefore, session study patterns do not necessarily discriminate between student themselves. However, we do notice a predominance of certain session types on a per student basis, in good part due to the correlation of session length with session patterns.

How can such visualization and clustering method be of use to the design of learning environments, or to the teachers? It might help them to better understand how students use the learning environment and, in return, make adjustments to the features of the learning environment to better suit the actual patterns of usage.

At least in our case, this study revealed that, although the most engaged students did use the application as intended, with a predominance of Type 3 sessions, many students did not use it as intended, or even as expected. Both authors were involved in the design, and we had expected much more back and forth between the study notes and exercises. Except for a few students, this did not happen very much. Instead, many sessions consisted of relatively long pauses with browsing periods through the notes and exercises, lasting sometimes over an hour without actually doing practice exercise. We intend to conduct interviews in later studies to better understand this behavior, but this current study helps us reveal the patterns to investigate further.

Acknowledgements

This study was funded by MATI (Maison des technologies de formation et d'apprentissage Roland-Giguère) under the project "Services d'apprentissage personnalisés".

5. REFERENCES

[1] A. Abbott and A. Tsay. Sequence analysis and optimal matching methods in sociology review and prospect. *Sociological Methods & Research*, 29(1):3–33, 2000.

[2] C. Beal and P. Cohen. Temporal data mining for educational applications. *PRICAI 2008: Trends in Artificial Intelligence*, pages 66–77, 2008.

[3] C. Beal, S. Mitra, and P. R. Cohen. Modeling learning patterns of students with a tutoring system using Hidden Markov Models. *Artificial intelligence in education: Building technology rich learning contexts that work*, 158:238, 2007.

[4] B. Everitt, S. Landau, and M. Leese. Cluster analysis. 4th. *Arnold, London*, 2001.

[5] A. Gabadinho, G. Ritschard, M. Studer, and N. S. Müller. Mining sequence data in R with the TraMineR package: A users guide for version 1.2. *Geneva: University of Geneva*, 2009.

[6] A. Hadwin, J. Nesbit, D. Jamieson-Noel, J. Code, and P. Winne. Examining trace data to explore self-regulated learning. *Metacognition and Learning*, 2:107–124, 2007.

[7] H. Jeong, A. Gupta, R. Roscoe, J. Wagster, G. Biswas, and D. Schwartz. Using hidden Markov models to characterize student behaviors in learning-by-teaching environments. In *Intelligent Tutoring Systems*, pages 614–625. Springer, 2008.

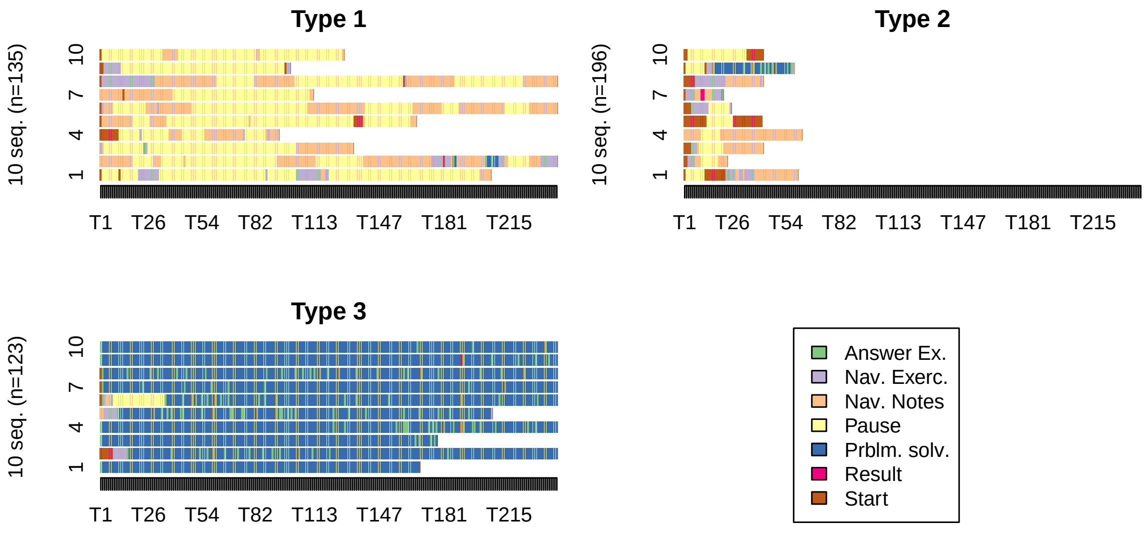


Figure 1: Random samples of 10 sessions for each type of cluster from various students. Type 1 corresponds to browsing through notes and exercises with frequent pauses and very little problem solving. Type 2 corresponds to short sessions of various behaviour. Type 3 are sessions focused on problem solving and answers to exercises.

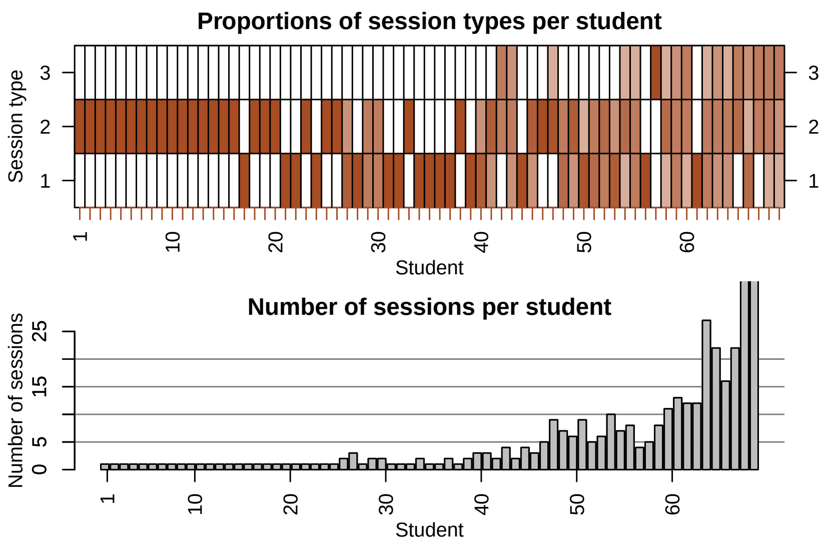


Figure 2: Type of sessions per student. Students are on the x-axis and ordered from shortest time of use (302 sec.) to the longest time (120 hrs.)—median time is 1.2 hr. and mean time is 6.1 hrs.

[8] M. Köck and A. Paramythis. Activity sequence modelling and dynamic clustering for personalized e-learning. *User Modeling and User-Adapted Interaction*, 21:51–97, 2011. 10.1007/s11257-010-9087-z.

[9] A. Merceron and K. Yacef. Tada-ed for educational data mining. *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning*, 7(1):267–287, 2005.

[10] J. Mostow, J. Beck, H. Cen, A. Cuneo, E. Gouvea, and C. Heiner. An educational data mining tool to browse tutor-student interactions: Time will tell. In *Proceedings of the Workshop on Educational Data Mining, National Conference on Artificial Intelligence*, pages 15–22, 2005.

[11] C. Romero, S. Gutiérrez, M. Freire, and S. Ventura. Mining and visualizing visited trails in web-based educational systems. *Educational Data Mining 2008*, page 182, 2008.

[12] K. Vanlehn, C. Lynch, K. Schulze, J. A. Shapiro, R. Shelby, L. Taylor, D. Treacy, A. Weinstein, and M. Wintersgill. The ANDES physics tutoring system: Lessons learned. *Int. J. Artif. Intell. Ed.*, 15(3):147–204, 2005.

Analyzing the Mental Health of Engineering Students using Classification and Regression

Melissa Deziel, Dayo Olawo, Lisa Truchon, Lukasz Golab
University of Waterloo, Canada
{madeziel, oaolawo, laatruch, lgolab}@uwaterloo.ca

ABSTRACT

In this paper, we describe a data mining study of the mental health of undergraduate Engineering students in a large Canadian university. We created a survey based on guidelines from the Canadian Mental Health Association, and applied classification and regression algorithms to the collected data. Our results reveal interesting relationships between various aspects of mental health and year of study (first and final year students have lower mental health scores than second-year students), academic program (students in competitive programs have lower overall mental health but higher self-actualization, whereas students in a program with a flexible curriculum had higher overall scores), and gender (female Engineering students tend to have lower scores).

Keywords

Mental health data mining, linear regression, rule mining

1. INTRODUCTION

Mental health affects all facets of daily life and therefore awareness is critical. In particular, the well-being of undergraduate students is imperative as it greatly affects their academic success. In this paper, we advocate the use of data mining to understand the factors affecting mental health. We describe a case study in which we applied regression and classification algorithms to mental health survey data collected in a large Canadian university. We focus on Engineering, a competitive and demanding discipline with a heavy gender bias towards male students.

We conducted an anonymous survey - both online and in person - in which we asked students to rate five aspects of their mental health, as defined by the Canadian Mental Health Association [4]. These five aspects are: Ability to Enjoy Life, Resilience, Balance, Emotional Flexibility and Self-Actualization. Our survey also included questions about potential academic influences on mental health such as year of study, academic program, gender, academic workload, and relationship status. We received over 300 responses in total.

We then applied linear regression and classification algorithms to identify which of the above external influences have the greatest effect on each aspect of mental health. Examining the regression coefficients and classification rules revealed interesting insights into the mental health of Engineering students. We found that the number of hours of homework was the best predictor of overall mental health, followed by year of study. In particular, first-year and final-year students tend to have lower mental health scores while second-year students have the highest scores. We also found that female students in all academic programs and years have lower overall mental health but higher Emotional Flexibility. Another interesting result was that students in highly competitive and challenging programs have lower overall mental health scores but higher Self-Actualization, whereas students in inter-disciplinary programs with a flexible curriculum tend to have higher scores.

The aim of this paper is to illustrate how data mining algorithms can be used to analyze mental health data and to encourage further work on applying machine learning to better understand mental well-being. We discuss related work in Section 2, our methodology in Section 3, and our results in Section 4. We conclude in Section 5 with recommendations arising from this study and suggestions for future work.

2. RELATED WORK

The importance of mental health and well-being in students is exemplified by the large number of studies on this topic. Past research has focused on using surveys to identify factors that affect mental health, but applying machine learning tools to such data has not received much attention. One recent example is Li et al. [3], which examined variables such as ethnicity, gender and age to classify the mental health of Chinese college students into three groups using regression. They found that the strongest predictor of adjustment and severe mental health problems was the level of satisfaction with one's major. In this paper, we consider a wider variety of education-related features, including gender, year of study, academic workload and academic program, and we examine five different aspects of mental health.

Another interesting example of mining mental health data is described in Diederich et al. [1], which used machine learning techniques to identfy mental health issues such as schizophrenia, and mania. Their thesis was that through analyzing data or language and conversations between psychiatrists and patients, they could develop more accurate diagnostic classification systems. The study used various methods including emotional classification and clustering algorithms.

In general, previous work on understanding the mental health of students has investigated factors such as gender and academic year. The Center for Addiction and Mental Health [6] found through surveys that females were more prone to report mental health issues than males, and final-year students were least likely to report these symptoms as compared to students in other years. The National Union of Students [5] conducted a survey of colleges and universities across Scotland found that examinations, concerns about future career prospects and finances were the major sources of stress. Zacaj [9] studied gender differences in

Engineering education, focusing on women's integration in the classroom, outlook for future careers, decision-making power and responsibility. Soet and Sevig [7] investigated the effect of ethnicity and sexual orientation on various mental health problems (depression, eating disorders, substance use, etc.). One insight from this study was that African American students were found to be less distressed than their counterparts. Finally, Trockel et al. [8] studied the academic performance of first-year college students and their health behaviour. Their results demonstrate that feelings of anxiety, depression, and time pressure negatively affected the performance of these students. However, participating in extracurricular activities alongside having a good support system positively affected the academic performance. In this paper, we use regression and classification to reach similar conclusions to those reported in previous work (including finding that women and first-year students in Engineering tend to have lower mental health and that the number of hours of homework is strongly correlated with mental health), and we present new insights into the mental health of Engineering students enabled by the use of regression and classification algorithms.

3. METHODOLOGY

The first part of our survey included questions about potential academic influences on mental health, summarized in Table 1. Previous work has focused on factors such as financial situation and career prospects; in this study, we focus mainly on academic factors. In particular, we hypothesize that students enrolled in competitive programs with a high workload and an imbalance of extra-curricular activities will have lower mental health scores.

Table 1: Attributes used in the first part of the survey

Attribute	**Possible Values**
Gender	Male, Female
Year of study	1,2,3,4
Engineering program	Environmental, Electrical, Computer, Systems Design, Mechanical, Chemical, Geological, Civil
Hours of class per week	0-40
Hours of homework per week	0-80
In a committed relationship of more than 6 months?	Yes, No
Recent relationship break-up?	Yes, No
Hours of extracurricular activities per week (sports, student politics, student clubs, etc.)	0-40

The second part asked the participants to rate, on a scale from zero to six, the following five aspects of their mental health.

1. Ability to Enjoy Life: characterized by enjoying the present and worrying less about the future or the past.

2. Resilience: ability to recover from adversity; a characteristic shared by those who cope well with stress and change.

3. Balance in various aspects of life, such as time spent alone or with others, work and play, sleep and wakefulness, and rest and exercise.

4. Emotional Flexibility: ability to reduce stress that is obtained from rigid emotional expectations.

5. Self-Actualization: ability to recognize one's abilities and the process of this recognition.

These five aspects were defined by the Canadian Mental Health Association [4]. In addition to considering each aspect separately, we also summed up the five scores to compute an overall mental health score for each respondent (on a scale from zero to 30).

We conducted the survey online and on campus during a seven-day period in the Fall semester, targeting undergraduate Engineering students from all programs. We received 312 responses, which corresponds to 5.6 percent of the Engineering student population, and discarded six due to missing responses and/or values out of bounds. 70 percent of the respondents were male. 71 percent were single.

We used the WEKA data mining toolkit [2] to analyze the survey results. First, we applied least-squares linear regression to predict the overall mental health score as well as the individual five component scores. We then computed separate linear regression models for selected Engineering programs (we ignored programs from which we received very few responses), and separate models for each year of study (across all programs), to see if certain programs or years face unique mental health challenges.

Next, we discretized the numeric attributes. Mental health scores for each category were converted into: Very Low (0), Low (1), Medium (2-3), High (4), Very High (5) and Excellent (6). Overall mental health scores were converted into: Very Low (0-5), Low (6-10), Medium (11-15), High (16-20), Very High (21-25) and Excellent (26-30). Hours of class and extracurricular activities per week were converted as follows: Low (0-15), Medium (15-30), High (30-40); hours of homework per week were converted similarly. Using the transformed data, we computed prediction rules for overall mental health and for the five components of mental health using the PRISM algorithm. Then, as before, we separately computed rules for each program of study (across all years) and each academic year (across all programs).

Having computed the above regression and classification models, we examined the regression coefficients and classification rules to understand which attributes have the greatest effect on the mental health of students.

Finally, we conducted exit interviews with a sample of the respondents to help explain the results of our analysis.

4. RESULTS

According to our regression and classification results, the strongest signals in the data were as follows:

- In terms of overall mental health, the year of study and number of hours of homework had the greatest effect.
- Second-year students had the highest overall scores and first-year students had the lowest scores. Fourth-year students also had relatively low scores.
- In terms of academic programs, Electrical Engineering students had lower mental health scores due to the competitive nature of the program, while Systems Design students had higher scores due to strong classmate relationships and a flexible curriculum.
- Women in all Engineering programs have lower overall mental health (especially in Mechanical Engineering), but higher Emotional Flexibility.

- Self-Actualization was negatively affected by a high number of hours of class per week, but positively affected by a high number of homework hours per week.
- Being in a relationship reduces the Balance and Emotional Flexibility mental health components.

Table 2 summarizes the Root Mean Squared Error (RMSE) of the regression models and the prediction accuracy (the number of correctly classified instances divided by the total number of instances) of the PRISM classification rules for each mental health component. We used 10-fold cross-validation in each case. It does not appear that any one component is significantly easier or harder to predict using the features collected in our survey.

Table 2: RMSE and prediction accuracy for each mental health component

Component	RMSE	Prediction Accuracy
Ability to Enjoy life	1.29	80%
Resilience	1.32	84%
Balance	1.31	80%
Self-Actualization	1.53	83%
Flexibility	1.12	75%

In the remainder of this section, we discuss the above findings in more detail, paying particular attention to the impact of the academic program, year of study and gender on the overall mental health and its five components.

4.1 Impact of Year of Study

For each academic program within the Faculty of Engineering, we computed a regression model that predicts the overall mental health score based on four indicator variables corresponding to the four academic years. Table 3 shows the regression coefficients. Based on their magnitudes and signs, it appears that being in first year or in fourth year has a negative effect on the overall mental health score, with first year having the most negative impact. Exit interviews were conducted to determine the lead indicators of these results.

Table 3: Regression coefficients for predicting overall mental health for each academic program based on academic year

	First Year	Second Year	Third Year	Fourth Year
Electrical	-1.1848	0.9269	-0.0288	-0.1853
Environmental	-1.2735	0.922	-0.0748	-0.1589
Mechanical	-1.2312	0.9174	-0.0021	-0.1374
Civil	-1.235	0.9194	-0.0175	-0.1042
Systems Design	-1.159	0.8555	0.0714	-0.2464
Computer	-1.2241	0.9146	0.0085	-0.152
Geological	-1.1696	0.9022	-0.0725	-0.0732
Chemical	-1.2026	0.915	-0.0248	-0.13

Based on the exit interviews, first-year students found it difficult to be separated from family and friends from home. They also found that moving to a new city required lifestyle and workload adjustments. Furthermore, first-year Engineering students have a high classroom workload (35 to 40 hours of class per week), with many courses requiring weekly homework assignments, which adds to the high workload.

Second-year students were more relaxed due to the decreased focus on weekly evaluations. In particular, second-year students reported the lowest number of hours of homework. These students also found that the reduced focus on the fear of failure was a large stress reducer. As they had been successful passing first year, their confidence levels were increased.

However, by the time students reach the last year of study, many students find themselves overwhelmed with complex and lengthy course projects. Student debts are typically increasing and this compounds their worries. Late in the third year and early in the final year, some students feel that they are ready to finish their undergraduate career.

4.2 Impact of Academic Program

Both regression and classification results revealed that students from Systems Design Engineering tend to have higher scores. For example, the following rule was discovered by PRISM:

> If Systems Design Engineering = Yes and Year = Third then Overall Mental Health = Very High

Exit interviews suggested that Systems Design Engineering students expressed a high cohesion among classmates across all years. From the first day, these students are encouraged to make strong relationships. Students in this program found that their ability to be hired in any industry for internships proves their program to be fulfilling and accomplishing in real-world applications. Finally, Systems Design students have the freedom to focus on areas of interest through a more lenient course elective program.

On the other hand, being in Electrical Engineering was negatively correlated with mental health according to our regression and classification results. As per the exit interviews, the ultra-competitive nature of this program and the large number of compulsory courses (leaving little time for elective courses) appeared to be the main culprits.

4.3 Impact of Gender

In four of the five categories (Ability to Enjoy Life, Resilience, Balance, and Self-Actualization), being female negatively impacted the regression model for mental health across all programs and years. Being female was a positive factor only for Emotional Flexibility. Running PRISM to determine the rules associated with the mental health of women in engineering provided some specific insight into the Mechanical Engineering program as well as females in engineering as a whole:

> If Mechanical Engineering = Yes, and Gender = Female then Overall Mental Health = Low
>
> If Gender = Female then Flexibility = Very High

To gather some insight into these findings, exit interviews were conducted with female students. Being the minority in most classes, especially in Mechanical Engineering where the gender bias is severe, makes most women feel like they need to prove themselves to their professors, their male counterparts and in male-dominated industries on work terms. Most of the professors are male and the delivery of material is tailored towards men (humour during class is often directed towards male students). Lastly, there is a lack of female companionship and there are high levels of competitiveness within the women in classes, which contributes to feelings of isolation and loneliness.

4.4 Factors Affecting the Five Components of Mental Health

Overall, Engineering students rated their ability to enjoy life more highly than other mental health aspects. According to our exit interviews, students recognized that “some things can’t be changed”, and when asked why, the responses ranged from technical to organizational and even philosophical barriers.

According to the linear regression results, being in the Electrical Engineering program had negative coefficients for every mental health category except Self-Actualization. Also, for students in Environmental Engineering, Self-Actualization was lower in the first and second year and higher in later years.

The number of hours of class per week had a negative effect on Self-Actualization across all programs, and a negative effect on the Ability to Enjoy Life for most programs. However, the number of hours of class per week had a positive coefficient for the other three categories (Emotional Flexibility, Balance and Resilience). On the other hand, the number of hours of *homework* had a positive effect on Self-Actualization, but a negative effect on every other category. Time in the classroom is mandated and there is no emphasis on discovery, but the time spent applying this knowledge on homework is self-motivated, which could aid in realizing one’s full potential.

According to our regression results across all programs, the number of hours spent on extracurricular activities had a very small negative effect on the Flexibility category of mental health, and a small positive effect on all other categories.

We found that being in a committed relationship had a negative effect on Balance and Emotional Flexibility, most likely due to the difficulty of managing a demanding academic program and personal relationships. In particular, we obtained the following rule:

If In a Committed Relationship of More than 6 Months = Yes and Year = First and Gender = Male then Balance = Low

Finally, according to linear regression results, having gone through a recent breakup yields a negative effect on the Ability to Enjoy Life and Emotional Flexibility.

5. CONCLUSIONS, RECOMMENDATIONS AND FUTURE WORK

In this paper, we presented a case study of how data mining may be used to understand factors affecting the mental health of students. We applied linear regression and classification algorithms to mental health surveys completed by Engineering undergraduate students, which revealed interesting relationships between various aspects of mental health and the academic program, year of study, gender, workload and relationship status.

The results of this study suggest a number of recommendations to help improve the mental health of Engineering undergraduate students. Given that the number of hours of homework was an important factor, it may be beneficial to offer first-year students additional time-management training. Furthermore, more support should be provided to female Engineering students, e.g., counseling services or forums to invite women to talk about their experiences.

Our analysis of the impact of the year of study on mental health (recall Table 2) suggests that first-year Engineering students are under pressure to succeed and more support should be provided to them. Furthermore, it appears that fourth-year students experience the pressure of multiple course projects. One solution may be to spread out project-heavy courses through the second and third year rather than leaving all of them till the last year.

Another interesting finding was that students in a highly competitive and challenging program (Electrical Engineering) tend to have low overall mental health scores but high Self-Actualization scores, whereas System Design Engineering students have high overall scores, which are possibly due to the flexible curriculum. Further research should be done on balancing the number of required and elective courses while maintaining a challenging and practical curriculum.

In this paper, we focused mainly on academic factors that may affect mental health, such as year of study, program and workload. An interesting direction for future work is how to reliably collect and analyze (in a manner that ensures data privacy) data describing other factors that may affect the mental health of students, such as instructor-class relationships, grades, and satisfaction with work placements in co-operative education.

6. REFERENCES

[1] Diederich, J., Al-Ajmi, A., Yellowlees, P., (2007). *Ex-ray: Data Mining and Mental Health*, Applied Soft Computing, 7(3):923-928.

[2] Hall M., Frank E., Holmes G., Pfahringer B., Reutemann P., Witten I. H., (2009). *The WEKA Data Mining Software: An Update*, SIGKDD Explorations, 11(1):10-18.

[3] Li, H., Li, W., Liu, Q., Zhao, A., Prevatt, F., Yang, J., (2008). *Variables Predicting the Mental Health Status of Chinese College Students*, Asian Journal of Psychiatry, 1(2):37-41.

[4] Meaning of Mental Health, Canadian Mental Health Association (2013), retrived from http://www.cmha.ca/mental_health/meaning-of-mental-health/#.UHr5GVH08YQ.

[5] National Union of Students Scottland, (2010). *Silently Stressed: A Survey into Student Mental Wellbeing*, retrieved from http://www.nus.org.uk/PageFiles/12238/THINK-POS-REPORT-Final.pdf.

[6] Paglia-Boak, A., Adlaf, E.M., Hamilton H.A., Beitchman, J.H., Wolfe, D., Mann R.E., (2012). *The Mental Health and Well-Being of Ontario Students, 1991-2011: Detailed OSDUHS Findings* (CAMH Research Document Series No. 34). Toronto, ON: Centre for Addiction and Mental Health.

[7] Soet, J., Sevig, T., (2006). *Mental Health Issues Facing a Diverse Sample of College Students:Results from the College Student Mental Health Survey*, Journal of Student Affairs Research and Practice, 43(3):786-807.

[8] Trockel, M.T., Barnes, M.D., Egget, D.L., (2000). *Health-Related Variables and Academic Performance Among First-Year College Students: Implications for Sleep and Other Behaviors,* The Journal of American College Health, 49(3):125-131

[9] Zacaj, A., (2010). *Gender Differences in Engineering Education*, M.A.Sc. Thesis, University of Waterlo

Hints: You Can't Have Just One

Ilya M. Goldin Kenneth R. Koedinger Vincent Aleven
Human-Computer Interaction Institute
Carnegie Mellon University
goldin@cmu.edu koedinger@cmu.edu aleven@cs.cmu.edu

ABSTRACT

A student using an interactive learning environment (ILE) may take multiple attempts to solve a problem step, at times using hints. But how effective are hints? Because data mining occasionally finds implausible negative effects of hints, a method is needed to remove selection effects related to hint use.

We distinguish multiple attempts in which a student repeatedly seeks hints from multiple attempts to answer the problem. Exploratory analysis of log data from a tutoring system shows that making a hint request rather on the first attempt on a problem step correlates with hint requests on subsequent attempts, and proficiency on a first attempt correlates with proficiency on subsequent attempts. Based on this, we devise a multinomial logistic regression that distinguishes hint-request tendency from proficiency. We find that seeking just one hint is associated with repeated hint-seeking, but when students do make attempts to solve a problem after viewing a hint, they succeed about half of the time. Thus, the model removes seemingly negative "effects" of hints. We also find that individual differences among students are more prominent in hint-seeking tendency than in proficiency with hints. We conclude with some ideas to improve our model.

Keywords

Effect of help on performance, individual differences, learning skills, multilevel Bayesian models, Item Response Theory

1. INTRODUCTION

Work on help-seeking in Interactive Learning Environments (ILEs) shows that effects of help are not always straightforward. For instance, different types of hints may differ in effectiveness, and students may differ in proficiency with hints [5, 6]. Further, students may have a variety of help-seeking behaviors, such as help-avoidance (a failure to seek help when the student would likely benefit from it), and help-abuse (seeking help when the student can likely answer the problem). [1] Occasionally, use of help may be linked with *negative* effects [2, 4, 5], but the negative estimate is unsatisfactory. It is doubtful that hints *cause* incorrect performance: although a hint may at times confuse a student and thus contribute to an error, it does not reduce student knowledge. More plausibly, a hint request evidences that the student has not understand the material. In a sense, negative estimates of hint effects imply that the statistical method behind these estimates is a poor representation of human performance (or learning). A better model would reflect a positive or neutral hint effect.

We consider whether the negative hint effects estimated in prior work are due to student tendency to request multiple hints without intervening attempts that could solve the problem. For example, one perspective is that attempt outcomes are effectively binary indicators of skill mastery, either successful (if correct) or not (if incorrect or a hint request). An incorrect attempt suggests that skill mastery is somehow deficient (although it may also be a slip), and a hint request suggests that the student does not know enough to answer the problem. Nonetheless, students may request hints to learn. Because hint requests and incorrect attempts can differ, we may need to distinguish tendency to answer correctly or incorrectly (proficiency) from tendency to request hints.

While we can only hypothesize about the myriad reasons that students may have to avoid or overuse hints, something we can quantify is a tendency to request a hint rather than answer incorrectly, i.e., when a hint may help [8]. This Tendency to Ask for Help-Not Risking an Incorrect (TAH-NRI) may differ across categories of attempts; for instance, on average, students may like to try to solve a problem a second time rather than to use hints. Further, students may differ in their tendency to request hints.

Counts of student actions may underestimate TAH-NRI, e.g., if the student answers incorrectly because a hint was unavailable, and overestimate it, e.g., if the student only seeks the bottom-out hint and must skip other hints to get to the last hint in a sequence. Proficiency may be viewed as the correctness rate when the student actually gives an answer. An operational definition of that is declining to request a hint, but students decline for a variety of reasons, including when they actually desire help. For instance, a student may be aware of own lack of prerequisite knowledge, yet may have poor experience with hints. In this light, distinguishing TAH-NRI from proficiency is a crude but potentially useful representation of metacognition.

We examine these two notions empirically. First, we explore frequencies and correlations of student proficiency and TAH-NRI on different types of problem-solving attempts. Second, this exploratory analysis informs a statistical model of proficiency and TAH-NRI. The model improves on exploratory estimates of proficiency and TAH-NRI by taking other predictors into account.

2. EXPLORATORY DATA ANALYSIS

We perform exploratory and model-based (Sec 0) analyses on a dataset of 51 9th grade students using the Geometry Cognitive Tutor. The students worked through 170 geometry problems, consisting of 1666 problem steps (about twice a week for five weeks). Each student only saw a subset of the 170 problems. In the Geometry Cognitive Tutor, a student may make multiple attempts to complete a problem step. Completing a step requires a correct response. On each attempt, a student may supply a correct answer, an incorrect answer, or may ask for a hint. We omit second hint displays; a hint's effectiveness in a specific problem step for a specific student is only evaluated once.

In our analysis, we first consider that how students behave on a first attempt on a problem step yields contextual information for understanding subsequent attempts. This leads us to explore the relation of first-attempt hint-request behavior to behavior on subsequent attempts. Second, we ask whether there are individual differences among students in hint requests and proficiency that may characterize the behavior of a student across time.

We group hint messages that differ in terms of surface features, i.e., in terms of the names and measures of the angles in a geometry problem. [6] We manually categorize each group of hint messages as feature-pointing, principle-stating or providing the

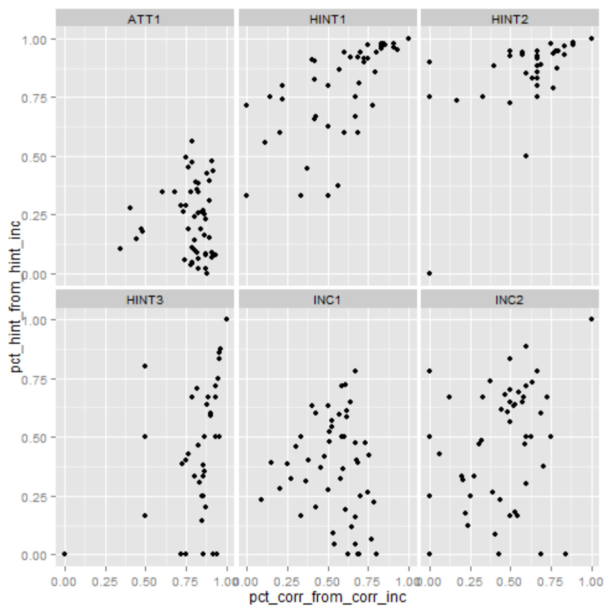


Figure 1: Average student proficiency vs. average TAH-NRI on each attempt type (in percentage-based measures). From top left: first attempts, attempts after feature-pointing hints, after principle-stating hints, after bottom-out hints, after the first incorrect outcome, after the second incorrect outcome.

bottom-out hint. Feature-pointing hints make salient important problem features, e.g., by pointing out that two particular angles are vertical angles. Principle-stating hints give a domain-specific principle that is necessary to solve the problem, e.g., that vertical angles are equal in measure. Bottom-out hints show how to find the answer to the problem, such as by summing known quantities.

Table 1: Rates of hint request and incorrect outcomes

First-Attempt Rate			
Hint Requests		Incorrects	
5%		16%	
Second-Attempt Rate		**Second-Attempt Rate**	
Hint Requests	Incorrects	Hint Requests	Incorrects
70%	5%	22%	35%

How does first-attempt hint behavior relate to behavior on a subsequent attempt? On a first attempt to solve a problem, students request hints only 5% of the time (Table 1), and enter incorrect responses more often (16%). On a second attempt, the conditional probability of a hint request given that the first attempt was also a hint request increases from 5% to 70%. In other words, students are unlikely to request a hint in the first place, but there is an extremely high rate of second hint requests after a first hint, 70% of all second attempts and 93% [70/(70+5)] of the non-correct second attempts after a hint. Statistical models of help-seeking behavior should take first-attempt behavior into account.

Second, we consider individual differences. In exploratory analysis, we define proficiency as the percent correct out of all attempts where a student p actually tries to solve the problem, i.e., attempts may be correct or incorrect, but not hint requests: $\frac{Corr_p}{Corr_p+Inc_p}$. Similarly, we define a student's TAH-NRI as the percent of hint requests out of all attempts when the student likely does not know enough to solve a problem, i.e., attempts that end in a hint request or incorrect, not in a correct outcome: $\frac{Hint_p}{Hint_p+Inc_p}$.

Figure 1 presents a distribution of proficiencies (X axis) and hint-request tendencies (Y axis) on each attempt type. Individual differences in proficiency and in TAH-NRI characterize a student across opportunities on each attempt type. Proficiency on attempts after hints is moderately or strongly related to TAH-NRI on those types of attempts. Proficiency on attempts after a first incorrect is weakly or even negatively related to TAH-NRI, but proficiency after a second incorrect is moderately related to TAH-NRI.

A student who is proficient on one attempt type should also be proficient on others. We find (Table 2) that proficiency on first attempts is moderately ($r = 0.39$) related to proficiency on attempts after a feature-pointing hint and after a second incorrect outcome ($r = 0.49$), and strongly related to proficiency on attempts after the first incorrect ($r = 0.74$). Nonetheless, proficiency on first attempts is not related to proficiency after other hints, nor to TAH-NRI. In general, we should take first-attempt proficiency into account when predicting performance on other attempts, but not necessarily when predicting hint requests.

Table 2: Correlations of first-attempt proficiency with proficiency and TAH-NRI on other types of attempts

Attempt Type	Corr. vs. Proficiency on Other Attempts	Corr. vs. TAH-NRI
First attempts	1.00	0.01
After FP Hint	0.39	0.12
After PS Hint	0.18	-0.08
After BOH	0.20	0.47
After 1st Incorrect	0.74	0.14
After 2nd Incorrect	0.46	0.30

In sum, models should account for student differences, and student effects are different when predicting a hint request rather than a correct outcome.

3. MODELING

We present a baseline-category multinomial logistic regression to predict which outcome is most likely on an attempt (*INCORRECT*, *HINT-REQUEST*, *CORRECT*). *INCORRECT* is the baseline outcome against which other outcomes are compared. With K outcomes, there are $K-1$ comparisons: comparison $k = 1$ of *INCORRECT* vs. *HINT-REQUEST* yields parameters related to TAH-NRI; comparison $k = 2$ of *INCORRECT* vs. *CORRECT* yields parameters related to proficiency.

As a basis for this multinomial model, we take the ProfHelp-ID logistic regression. [5] ProfHelp-ID classifies *CORRECT* versus other outcomes, combining *INCORRECT* and *HINT_REQUEST* because both indicate that the student lacks the knowledge to answer correctly. ProfHelp-ID predicts whether an attempt will have a *CORRECT* outcome based on general student proficiency, individual differences in proficiency with different attempt types, knowledge component easiness, and a history of prior practice with the knowledge component. Compared to a logistic regression with M parameters, a baseline-category model estimates up to $(K-1)M$ parameters, i.e., twice the number in ProfHelp-ID.

$$logit(\Pr(Y_{kph} = k)) = \boldsymbol{Z_h \lambda_{kp}} + \sum_{j \in KC} (\beta_{kj} + \gamma_{kj} s_{pj} + \rho_{kj} f_{pj})$$

Equation 1: ProfHelp-Multinomial

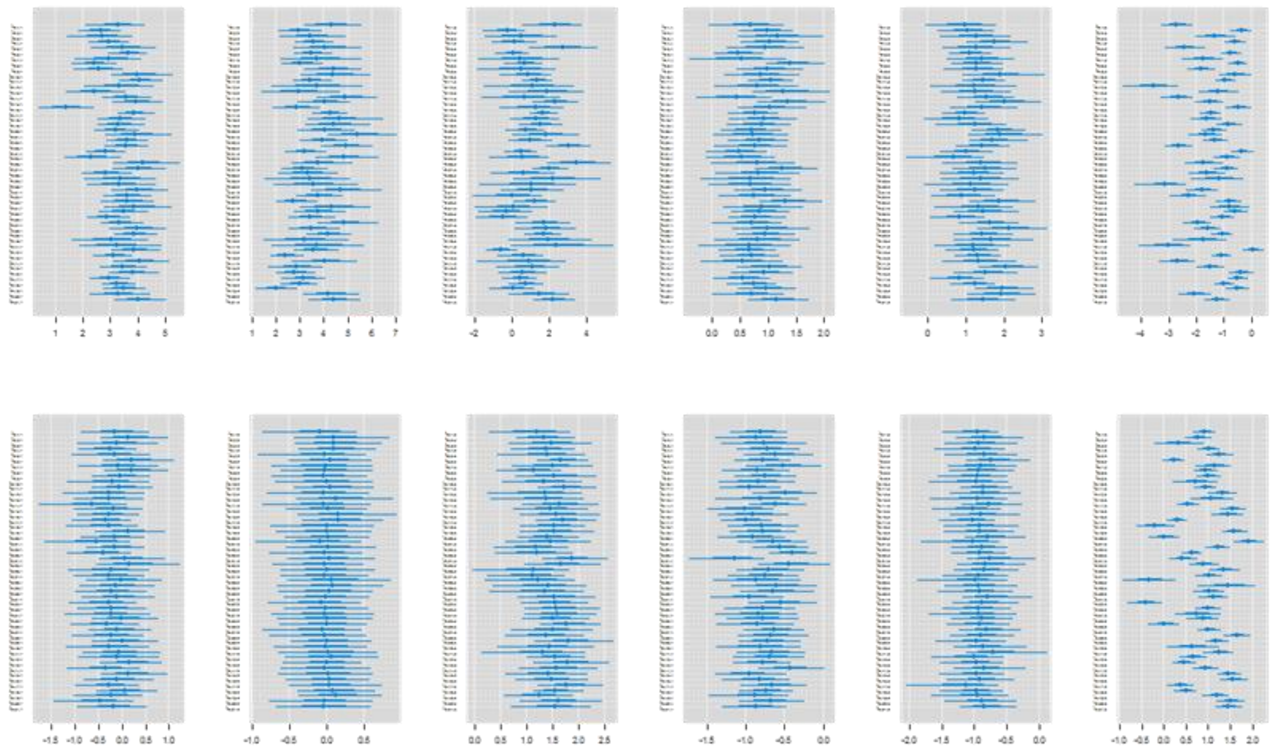


Figure 2: Logit estimates of λ_{kph}. TAH-NRI (top row) and proficiency (bottom). Left to right, the blocks are attempts after feature-pointing hints, principle-stating hints, bottom-out hints, one incorrect, two incorrects, and on first attempts. Each bar shows a 95% Credible Interval for λ_{kph} for one student, and the dot shows the median estimate. The X axes differ across blocks.

$\boldsymbol{\lambda}$ is a matrix, with one parameter for each pupil p, attempt type h and logit comparison k. The 6 possible attempt types are first attempts, attempts directly after each of three types of hints, and attempts directly after a first or second incorrect outcome on the given step. The subscript k implies that the parameter varies across the $K-1$ comparisons. Thus, for pupil p and attempt type h, λ_{1ph} represents TAH-NRI, and λ_{2ph} represents the proficiency, e.g., $\lambda_{1,2,3}$ is TAH-NRI of student 2 on attempts directly following level-2 (principle-stating) hints. As the exploratory data analysis suggests, first-attempt performance may relate to both TAH-NRI and proficiency. Accordingly, Z is a fixed $H \times H$ matrix such that the vector product of $\boldsymbol{Z_h}$ and $\boldsymbol{\lambda_{kp}}$ makes the estimate over first-attempts λ_{kp1} a reference level for estimates for the same k and p on other attempt types (i.e., where $h \neq 1$).

Pupil parameters are partially pooled, $\boldsymbol{\lambda_{kp}} \sim MVN_h(\boldsymbol{\Lambda_k}, \boldsymbol{\Sigma})$. Thus, each student's vector $\boldsymbol{\lambda_{kp}}$ is based on the averages $\boldsymbol{\Lambda_k}$ across all students, and each student's contribution to $\boldsymbol{\Lambda_k}$ is weighted by the number of observations for the student. The hyperprior for $\boldsymbol{\Lambda_h}$ is $N_h(0,1000)$, i.e., uninformative. We estimate the variance of the per-pupil parameters (diagonal of matrix $\boldsymbol{\Sigma}$), and impose a structure of zero covariance (off-diagonal cells in $\boldsymbol{\Sigma}$).

Other parameters pertain to knowledge component j. By analogy with per-pupil TAH-NRI and proficiency, β_{1j} represents the attractiveness to hints of KC j, and β_{2j} represents the KC's easiness. The slopes γ and ρ are the effects of student p's prior first-attempt successes s_{pj} and failures f_{pj} on the increased likelihood of a hint (for $k = 1$) and correct response ($k = 2$).

In sum, the model estimates the main effects and individual differences in proficiency on each attempt type, and the main effects and individual differences in TAH-NRI on each attempt type. Unlike the percentages used in exploratory analysis, these estimates account for other predictors, e.g., prior practice, and the model fitting yields credible intervals (CI) about these parameters.

4. RESULTS AND DISCUSSION

The model-fitting indicates substantial individual differences in first-attempt TAH-NRI (Figure 2, top right) and in first-attempt proficiency (bottom right). Estimates of first-attempt TAH-NRI are negative for almost all pupils, with posterior 95% CI for $\Lambda_{1,6}$ ranging -1.71 to -1.14, implying that first attempt hint-requests were unlikely. First-attempt proficiencies are about 1.0, i.e., students in this dataset tend to answer correctly on a first attempt.

Taking first-attempt TAH-NRI for each student as a reference level, attempts after hints are very positively associated with hint-requests rather than incorrects: feature-pointing and principle-stating hints (top left two blocks) are strongly positive for all students, and even bottom-out hints (top row, third block) trend positive for many students.

With first-attempt proficiency as a reference level, proficiencies on attempts after feature-pointing and principle-stating hints (bottom left two blocks) tend to neutral, with (-0.40, 0.08) and (-0.27, 0.25) posterior 95% CIs for $\Lambda_{2,1}$ and $\Lambda_{2,2}$, respectively. In other words, by decoupling incorrect outcomes from hint requests, a multinomial model removed the strongly negative "effect" of these hints estimated by a binary model. [5] Moreover, individual differences in proficiency with hints [5] may be due to differences in TAH-NRI (top left three plots) rather than differences in proficiency (bottom left three plots). It is plausible that correct responses are more likely after bottom-out hints than other hints, since bottom-out hints often reveal the correct response.

TAH-NRI is unlike other forms of help-seeking, including gaming-the-system behavior that involves "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" [3]. Neutral-to-positive proficiency with hints implies that students try to learn from hints, suggesting that TAH-NRI signifies learning, not gaming. Further, harmful gaming behavior was observed in only 8% to 27% of students; we find positive TAH-NRI for almost all students.

TAH-NRI is positive directly after a first incorrect and second incorrect outcomes, with (0.65, 1.05) and (1.15, 1.66) posterior 95% CIs for $\Lambda_{1,4}$ and $\Lambda_{1,5}$, respectively. By contrast, proficiency after a first incorrect or a second incorrect is negative, with 95% CI for $\Lambda_{2,4}$ =(-0.89, -0.64) and for $\Lambda_{2,5}$ = (-1.08, -0.71).

In sum, once we adjust for first-attempt tendency to request hints, we find that students use the hint system extensively, although there are ample individual differences. Students request hints more often after a previous hint request than after an incorrect.

By contrast, once we adjust for first-attempt proficiency, students differ less in proficiency than in TAH-NRI. The "main effect" of hints is neutral (feature-pointing and principle-stating hints) or positive (bottom-out hints). We caution that a better evaluation of these hint types would consider a variety of hint sequences.

4.1 Model Adequacy

As often happens in classification, the model is biased (Table 3) in favor of predicting the majority class (*CORRECT*).

Table 3: Confusion matrix

	Predicted *INCORRECT*	Predicted *HINT_ REQUEST*	Predicted *CORRECT*
INCORRECT	607	951	3123
HINT_ REQUEST	1017	2078	1195
CORRECT	235	953	15778

Attempts after a first or a second incorrect are more likely to be misclassified than indicated by their prevalence in the full dataset (Table 4). This is puzzling. At the heart of ProfHelp-Multinomial is the PFA model of first-attempt performance [7]. Performance after one incorrect or two incorrect outcomes is highly correlated with first-attempt performance (Table 1), so PFA should be reasonably accurate for these attempts, and the λ_{kph} parameters in ProfHelp-Multinomial should further improve accuracy.

For future work, first, in combination with prior findings on help-avoidance and help-abuse, our results imply that statistical models not only take into account local transitions from attempt to attempt, but longer sequences of attempts. This is consistent with the help-seeking model of Aleven et al [1]. Bridging data-mining and theoretical approaches will lead to a model that more accurately reflects student help use.

Second, the fact that students often make multiple attempts to solve a problem-step without hints coupled with ProfHelp-Multinomial's poor predictive accuracy of performance on attempts after incorrects calls for data mining and other research to understand same-step performance after incorrect outcomes.

Table 4: Prevalence of observations *vs.* prediction errors

Attempt Type	Percent of Dataset	Percent of Prediction Errors
First attempts	67	50
After FP Hint	7	10
After PS Hint	4	5
After BOH	6	5
After 1st Incorrect	11	20
After 2nd Incorrect	4	9

5. CONCLUSIONS

This work advances the study of same-step help use in ILE. Students have a tendency to request multiple hints in a row rather than risk an error. Our analysis improves on prior analyses of TAH-NRI [8] in that we find that TAH-NRI may differ based on the type of attempt, and it persists after accounting for proficiency, for a knowledge component's attractiveness to hints, and for prior practice. TAH-NRI is distinct from other help-abuse behavior, e.g., from gaming-the-system. Further, there are persistent individual differences among students in TAH-NRI.

Formalizing TAH-NRI in the ProfHelp-Multinomial model alleviates the selection bias that caused another model to estimate that hints had negative effects. The ProfHelp-Multinomial results suggest that students make a strategic decision to get help and stick with it across attempts, i.e., the decision to try to solve vs. to request a hint is not an independent decision at each attempt.

The improved understanding of help-seeking developed here is a step towards developing effective and efficient ILE, including systems that adapt to individual differences among students.

6. REFERENCES

[1] Aleven, 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.

[2] Aleven, V. and Koedinger, K.R. 2001. Investigations into help seeking and learning with a cognitive tutor. *Papers of the AIED-2001 Workshop on help provision and help seeking in interactive learning environments* (2001), 47–58.

[3] Baker, R.S.J. d. et al. 2008. Developing a generalizable detector of when students game the system. *User Modeling and User-Adapted Interaction*. 18, 3 (Jan. 2008), 287–314.

[4] Beck, J.E. et al. 2008. Does Help Help? Introducing the Bayesian Evaluation and Assessment Methodology. *Intelligent Tutoring Systems*. B.P. Woolf et al., eds. Springer. 383–394.

[5] Goldin, I.M. et al. 2012. Learner Differences in Hint Processing. *Proceedings of 5th International Conference on Educational Data Mining* (Chania, Greece, 2012), 73–80.

[6] Goldin, I.M. and Carlson, R. 2013. Learner Differences and Hint Content. *Proceedings of 16th International Conference on Artificial Intelligence in Education* (Memphis, TN, 2013).

[7] Pavlik Jr, P. et al. 2009. Performance Factors Analysis - A New Alternative to Knowledge Tracing. *Proceedings of 14th International Conference on Artificial Intelligence in Education* (Brighton, England, 2009), 531–538.

[8] Wood, H. and Wood, D. 1999. Help seeking, learning and contingent tutoring. *Computers and Education*. 33, 2 (1999), 153–170.

What and When do Students Learn? Fully Data-Driven Joint Estimation of Cognitive and Student Models

José P. González-Brenes[†] and Jack Mostow[*]
Project LISTEN
www.projectlisten.com
[†]Language Technologies Institute, [*]Robotics Institute
Carnegie Mellon University, Pittsburgh, PA
{joseg, mostow}@cmu.edu

ABSTRACT

We present the Topical Hidden Markov Model method, which infers jointly a cognitive and student model from longitudinal observations of student performance. Its cognitive diagnostic component specifies which items use which skills. Its knowledge tracing component specifies how to infer students' knowledge of these skills from their observed performance. Unlike prior work, it uses no expert engineered domain knowledge — yet predicts future student performance in an algebra tutor as accurately as a published expert model.

Keywords

knowledge component discovery, student modeling, cognitive diagnostic model, knowledge tracing

1. INTRODUCTION

Assessing students' skills from their performance requires a *cognitive diagnostic model* specifying which observed items require which skills (sometimes called knowledge components), and a *student model* that infers how well students know each skill, based on their performance on items requiring that skill. For example, a cognitive diagnostic model for a reading tutor that listens to children read aloud might model the graphophonemic patterns in a word as distinct skills. Cognitive diagnostic models are typically engineered by human domain experts at considerable expense. Methods to infer them automatically from student performance data have been restricted to static instruments such as exams or homework assignments administered only once or twice. However, intelligent tutorial decisions require a student model that traces changes in student skills dynamically over time. This paper presents and evaluates the novel data-driven Topical HMM method to discover a cognitive diagnostic model and a student model simultaneously.

.

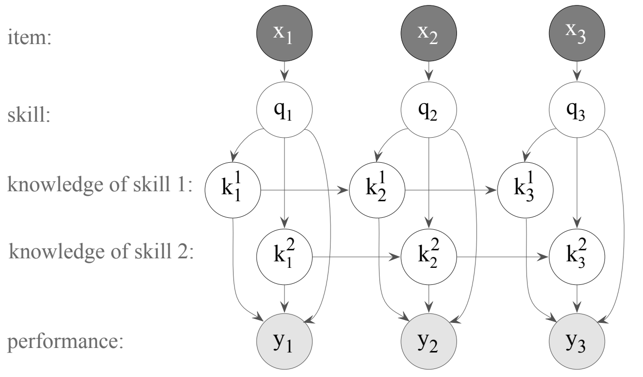


Figure 1: Unrolled example of Topical HMM with two skills ($\mathbf{S} = 2$), for a single user ($\mathbf{U} = 1$) with three time steps ($\mathbf{T_1} = 3$). Student indices, parameters (Q, K, D) and priors (α, τ, ω) are omitted for clarity. Dark gray variables are observable during both training and testing. Light gray variables are visible only during training. White variables are never observed (latent).

2. TOPICAL HIDDEN MARKOV MODEL

Topical HMM treats the skills required by a sequence of observed items as latent topics. We use a mixed membership model to represent the latent skill(s) required by an item. That is, we represent the item as requiring a single skill whose identity is uncertain but has a specified probability distribution, which we interpret as specifying the relative weight of each skill for the item. Figure 1 unrolls this graphical model for two skills. The absence of connections between knowledge nodes for different skills assumes no transfer between skills, i.e., the student's knowledge of a skill can change only when the student encounters an item that requires the skill. This assumption makes possible an efficient Gibbs sampler not described here.

Algorithm 1 specifies Topical HMM's generative story. It has hyper-parameters, variables, parameters, and priors.

Topical HMM's hyper-parameters are given or tuned:

- **S** is the number of skills in the model.
- **U** is the number of users (students).
- $\mathbf{T_u}$ is the number of time steps student u practiced.
- **M** is the number of items. For example, in the case of a reading tutor, **M** may represent the vocabulary size.

In a tutor that creates items dynamically, **M** is the number of templates from which items are generated.

- **L** is the number of levels of knowledge of a skill, typically 2 (knowing it or not). To distinguish novice, medium, and expert proficiency, we would use $\mathbf{L} = 3$.

Topical HMM's variables correspond to nodes in Figure 1:

- $x_{u,t}$ is the item the student u encountered at time t.
- $q_{u,t}$ is a latent random variable specifying the skill(s) required for item $x_{u,t}$.
- $k^s_{u,t}$ is a variable that takes values from $1 \ldots L$ to represent the level of knowledge of skill s. There is a Markovian dependency across time steps: if skill s is known at time $t-1$, it is likely to still be known at time t.
- $y_{u,t}$ represents student performance as a binary variable (correct or not), observed only during training.

Topical HMM's parameters specify the distributions of these variables. Since we take a fully Bayesian approach, we model parameters as random variables:

- $Q^{x_{u,t}}$ is the cognitive diagnostic model. It represents the skill(s) required for item $x_{u,t}$ as a multinomial $Q^{x_{u,t}}$ to model soft membership. For example, $Q^{x_{u,t}} = [0.75, 0.25, 0, 0]$ means that item $x_{u,t}$ depends mostly on skill 1, less on skill 2, and not at all on skills 3 or 4. Unlike prior work where the mapping of items to skills must be given, Topical HMM allows Q to be hidden, i.e. discovered entirely from data.
- $K^{s,l}$ is a multinomial that specifies the transition probabilities from knowledge state l of skill s to other knowledge states.
- $D^{s,l}$ is a binomial that specifies the emission (output) probability of a correct answer given the student's proficiency level l on the required skill s.

Topical HMM uses Dirichlet priors α, τ, ω for its parameters.

3. EVALUATION

We use data collected by the Bridge to Algebra Cognitive Tutor® [8] from 123 students, each of whom encountered an average of 340.7 items (minimum 48, maximum 562, median 341), for a total of 41,911. The data is unbalanced: over 80% of the items were correct.

We randomly partition the data into three sets with non-overlapping students – a training set with 97 students, and development and test sets with 13 students each. We use the development set to tune hyper-parameters and select the number of skills to model the data. We use the training set exclusively for learning the parameters of the model, and we only report results on the development or test set. To avoid tuning on test data, we used the test set only once, just before writing this paper.

The data set contains data from 893 different problems. Each problem consists of a sequence of one or more steps, and it is at this level that we do our analysis. We consider the different steps to be the items the student encounters. Students did not follow the curriculum in the same order; the tutor decided which problems to assign in what order, and the students chose the order to do the steps in each problem. To name items consistently across students, we named each

Algorithm 1 Generative story of Topical HMM

Require: A sequence of item identifiers $x_{1 \ldots T_u}$ for **U** users, number of skills **S**, number of student states **L**, number of items **M**

```
 1: function Topical HMM($x_1 \ldots x_t, \mathbf{S}, \mathbf{U}, \mathbf{L}, \mathbf{M}$)
 2:     ▷ Draw parameters from priors:
 3:     for each skill $s \leftarrow 1$ to $\mathbf{S}$ do
 4:         for each knowledge state $l \leftarrow 1$ to $\mathbf{L}$ do
 5:             Draw parameter $K^{s,l} \sim$ Dirichlet($\tau^{s,l}$)
 6:             Draw parameter $D^{s,l} \sim$ Dirichlet($\omega^{s,l}$)
 7:     for each item $m \leftarrow 1$ to $\mathbf{M}$ do
 8:         Draw $Q^m \sim$ Dirichlet($\alpha$)
 9:     ▷ Draw variables from parameters:
10:     for each student $u \leftarrow 1$ to $\mathbf{U}$ do
11:         for each timestep $t \leftarrow 1$ to $\mathbf{T}_u$ do
12:             Draw skill $q_{u,t} \sim$ Multinomial($Q^{x_{u,t}}$)
13:             for $s \leftarrow 0$ to $\mathbf{S}$ do
14:                 if $s = q_{u,t}$ then
15:                     ▷ knowledge state could change:
16:                     $k'' \leftarrow k^s_{u,t-1}$          ▷ previous time step
17:                     Draw $k^s_{u,t} \sim$ Multinomial($K^{s,k''}$)
18:                 else
19:                     ▷ knowledge state can't change:
20:                     $k^s_{u,t} \leftarrow k^s_{u,t-1}$
21:             $q' \leftarrow q_{u,t}$                         ▷ current skill
22:             $k' \leftarrow k^{q_{u,t}}_{u,t}$               ▷ current knowledge state
23:             Draw performance $y_{u,t} \sim$ Multinomial($D^{q',k'}$)
```

item by concatenating the tutor-logged problem name and step name, yielding 5,233 distinct items.

We evaluate cognitive diagnostic model by how accurately they predict future student performance. We operationalize predicting future student performance as the classification task of predicting whether students correctly solved the items on a held-out set. This paper focuses on predicting performance on unseen students. To make predictions on the development and test set, we use the history preceding the time step we want to predict. To speed up computations, we predict up to the up to the 200th time step in the test set. Since we run evaluations multiple times in the development set, we only predict up to the 150th time step. Therefore, our development and test sets have 1950 and 2600 observations respectively.

We evaluate the classifiers' predictions using a popular data mining metric, the Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve. The ROC evaluates a classifier's performance across the entire range of class distribution and error costs. An AUC of 1 represents a perfect classifier; an AUC of 0.5 represents a useless classifier, regardless of class imbalance. AUC estimates can be interpreted as the probability that the classifier will assign a higher score to a randomly chosen positive example than to a randomly chosen negative example.

The manual expert cognitive diagnostic model was developed and refined by two cognitive scientists and a teacher over four years. They first identified 76 different categories of items, and then determined that students would need fifty different skills to answer them. The manual model includes some items that use multiple skills.

When we use Topical HMM with a manually designed model, we initialize the parameter Q of Topical HMM with the expert model and do not update its values. In the case that the expert decided that an item uses multiple skills, we assign uniform weight to each skill even though the experts assumed a conjunctive model. Topical HMM cannot handle a conjunctive cognitive diagnostic model.

We now describe the values we use for the priors' hyper-parameters α, τ, and ω.

- **Sparse cognitive model**. We encourage sparsity on the cognitive diagnostic model parameter (Q), motivated by the assumption that items use only a few skills. We set $\alpha = 0.1$, because when the value of the hyper-parameter of a Dirichlet prior is below one, the samples are sparse multinomials. For example, $Q_i = [1, 0, 0, 0]$ is a sparse multinomial, that represents that item i depends on skill 1, but not on skill 2, 3 or 4.
- **Practice helps learning, and there is no forgetting**. Manipulating the magnitude of the hyperparameters τ and ω allows us to select the strength of the prior belief that students transition to a higher level of knowledge, and that they do not go back to the previous level. We use cross validation to select the magnitude of these hyperparameters with values 10 or 100.

For our experiments, we initialize the model randomly and then collect 2,000 samples from a Gibbs Sampling Algorithm. We discard the first 500 samples as a burn-in period. To infer future student performance, we save the last 1,500 samples, averaging over the samples and calculating the Maximum A Posteriori (MAP) estimate.

We compare the performance of these methods:

- **HMM**. Can we find evidence of multiple skills? Topical HMM should perform better than a cognitive model that assigns all of the items to a single skill. We run Knowledge Tracing [4] with a cognitive diagnostic model that has only one skill in total. This approach is equivalent to a single HMM.
- **Student Performance**. What is the effect of students' individual abilities? We predict that the likelihood of answering item at time t correctly is the percentage of items answered correctly up to time $t - 1$. Intuitively, this is the student's "batting average".
- **Random cognitive diagnostic model**. Does the cognitive diagnostic model matter? We create a random cognitive diagnostic model with five skills and assign items randomly to one of five categories. We then train Topical HMM to learn the student model (transition and emission probabilities), without updating the cognitive diagnostic model.
- **Item difficulty**. What is the classification accuracy of a simple classifier? We use a classifier that predicts the likelihood of answering item x as the mean performance of students in the training data on item x. Note that this classifier does not create a cognitive diagnostic model.
- **Manual cognitive diagnostic model**. How accurate are experts at creating a cognitive diagnostic models? We use Topical HMM with the 50-skill cognitive diagnostic model designed by an expert.
- **Data-driven cognitive diagnostic model**. We initialize Topical HMM with the best model discovered using the development set (with 5 skills).

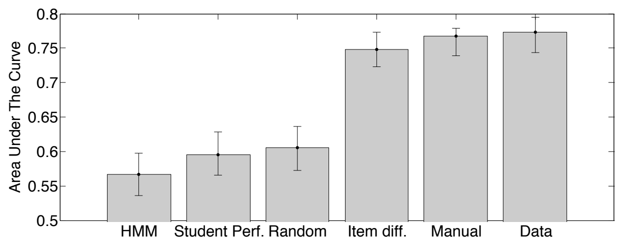


Figure 2: Test set AUC performance of different models

Figure 2 shows the AUC of the different methods applied to the test set, with 95% confidence intervals calculated with an implementation of the Logit method [1]. Our data-driven model with five skills outperforms all of the other models, with an AUC of 77.28. Because the confidence intervals do not overlap, we can conclude with 95% confidence that our data-driven model is significantly better than assuming a cognitive diagnostic model with a single skill (HMM), using the student's "batting average" (Student Perf.), or assigning items to skills randomly (Random). The confidence intervals for the data-driven cognitive diagnostic model, the manually engineered cognitive diagnostic model, and the item difficulty approach overlap, with AUC scores of 77.28, 76.71 and 74.76 respectively.

4. RELATION TO PRIOR WORK

This section relates Topical HMM to prior work in automatic discovery of cognitive diagnostic models and student models. In psychometrics, the branch of psychology and education concerning educational statistics, matrix factorization methods have been applied to discover a cognitive diagnostic model from static assessment instruments such as a single exam, or a homework assignment. A survey of previous approaches to automatic discovery of cognitive diagnostic models can be found elsewhere [13]; popular approaches include Item Response Theory [10], and matrix factorization techniques such as Principal Component Analysis, Non-Negative Matrix Factorization [5, 13], and the Q-Matrix Method [1]. These methods can help explain what skills students have mastered, but they ignore the temporal dimension of data. Unlike Topical HMM, these approaches do not discover a clustering of items to skills per se: performance is based on continuous latent traits. More specifically, matrix factorization techniques predict student performance as a combination of latent user traits, and latent item difficulty traits (skills) that may be multidimensional. Moreover, matrix factorization techniques cannot be applied to the problem of predicting performance of unseen students, because they require the latent user trait matrix. This problem also carries over for higher dimension factorization techniques, such as tensor factorization [12].

Learning Factors Analysis [3] uses temporal data, but requires initial knowledge to improve upon. Dynamic Cognitive Tracing [7] proposed a fully automatic method, but did not scale due to memory use exponential in the number of items and runtime exponential in the number of skills. Moreover, Dynamic Cognitive Tracing was only tested on synthetic data.

[1] http://www.subcortex.net/research/code/area_under_roc_curve

To our knowledge, we are the first ones to take time into consideration to estimate a cognitive diagnostic model from data of real students interacting with a tutor.

Knowledge Tracing [4] is a popular method to model students' changing knowledge during skill acquisition. It requires (a) a cognitive diagnostic model that maps each item to the skill(s) required, and (b) logs of students' correct and incorrect answers as evidence of their knowledge of particular skills. Knowledge Tracing can be formulated as a graphical model: items that belong to the same skill are grouped into a single sequence, and an HMM is trained for each sequence. The observable variable is the performance of the student solving the item, and the hidden state is a binary latent variable that represents whether the student knows the skill. Topical HMM generalizes Knowledge Tracing, which assumes the cognitive diagnostic model is known and each item uses exactly one skill. Topical HMM discovers the cognitive diagnostic model automatically and is more flexible since it allows more than one skill per item.

Attempts to use tensor factorization – matrices with more than two dimensions – to model student learning have been limited [12] as they require all students and items to be seen during training, which is often not feasible.

Other approaches to student modeling also exist. Performance Factors Analysis [6] predicts student performance based on item difficulty and student performance. Learning Decomposition [2] uses non-linear regression to determine how to weight the impact of different types of practice opportunities relative to each other. Parameter Driven Process for Change [11] is able to use different student modeling techniques, such as Knowledge Tracing or NIDA [9], to group students with similar response or skill patterns over time.

5. CONCLUSIONS AND FUTURE WORK

Our main contribution is a novel method, Topical HMM, which discovers cognitive and student models automatically. A difficulty of modeling real student data is sparsely observed students, items and skills. Unlike some prior methods, Topical HMM discovers cognitive diagnostic models that generalize to unseen students. Our work is also the first automatic approach to discover a cognitive diagnostic model from real student data collected over time.

Previous work on automatic discovery of cognitive diagnostic models from static data was successful in distinguishing between broad areas (i.e., French and Math), but not finer distinctions within an area [13, 5]. Given that we were able to discover different skills within an algebra tutor data set we are optimistic about this line of research. In future work we are interested in assessing the interpretability of the cognitive diagnostic models discovered by Topical HMM. A limitation of this study is that we evaluated our approach on only one dataset. Future work may test Topical HMM on more data sets from real students.

Acknowledgements

This work was supported in part by the Pittsburgh Science of Learning Center, the Costa Rican Ministry of Science and Technology (MICIT), and National Science Foundation Grant IIS1124240 to Carnegie Mellon University. The opinions expressed are those of the authors and do not necessarily represent the views of PSLC, MICIT or the National Science Foundation. We thank the educators, students, and LISTENers who helped in this study and the reviewers for their helpful comments.

6. REFERENCES

[1] T. Barnes, D. Bitzer, and M. Vouk. Experimental Analysis of the Q-Matrix Method in Knowledge Discovery. In M.-S. Hacid, N. Murray, Z. Ras, and S. Tsumoto, editors, *Foundations of Intelligent Systems*, volume 3488 of *Lecture Notes in Computer Science*, pages 11–41. Springer Berlin / Heidelberg, 2005.

[2] J. Beck and J. Mostow. How who should practice: Using learning decomposition to evaluate the efficacy of different types of practice for different types of students. In B. Woolf, E. Ameur, R. Nkambou, and S. Lajoie, editors, *Intelligent Tutoring Systems*, volume 5091 of *Lecture Notes in Computer Science*, pages 353–362. Springer Berlin / Heidelberg, 2008.

[3] H. Cen, K. Koedinger, and B. Junker. Learning factors analysis: A general method for cognitive model evaluation and improvement. In M. Ikeda, K. Ashley, and T.-W. Chan, editors, *Intelligent Tutoring Systems*, volume 4053 of *Lecture Notes in Computer Science*, pages 164–175. Springer Berlin / Heidelberg, 2006.

[4] A. Corbett and J. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.

[5] M. Desmarais. Conditions for Effectively Deriving a Q-Matrix from Data with Non-negative Matrix Factorization. In M. Pechenizkiy and T. Calders and C. Conati and S. Ventura and C. Romero and J. Stamper, editor, *Proceedings of the 4th International Conference on Educational Data Mining*, pages 169–178, 2011.

[6] Y. Gong, J. Beck, and N. Heffernan. Comparing knowledge tracing and performance factor analysis by using multiple model fitting procedures. In V. Aleven, J. Kay, and J. Mostow, editors, *Intelligent Tutoring Systems*, volume 6094 of *Lecture Notes in Computer Science*, pages 35–44. Springer Berlin / Heidelberg, 2010.

[7] J. P. González-Brenes and J. Mostow. Dynamic Cognitive Tracing: Towards Unified Discovery of Student and Cognitive Models. In Yacef, K., Zaïane, O., Hershkovitz, H., Yudelson, M., and Stamper, J. , editor, *Proceedings of the 5th International Conference on Educational Data Mining*, pages 49–56, 2012.

[8] K. Koedinger, R. Baker, K. Cunningham, A. Skogsholm, B. Leber, and J. Stamper. *A data repository for the community: The PSLC DataShop*. CRC Press, Boca Raton, FL, 2010.

[9] E. Maris. Estimating multiple classification latent class models. *Psychometrika*, 64(2):187–212, 1999.

[10] G. Rasch. On general laws and the meaning of measurement in psychology. In *Proceedings of the Fourth Berkeley symposium on mathematical statistics and probability*, volume 4, pages 321–333. University of California Press Berkeley, CA, 1961.

[11] C. Studer, B. Junker, and H. Chan. Incorporating learning into the cognitive assessment framework. In *Presentation at Annual Meeting of the Society for Research on Educational Effectiveness*, Washington, D.C., March 2012.

[12] N. Thai-Nghe, L. Drumond, A. Krohn-Grimberghe, and L. Schmidt-Thieme. Recommender system for predicting student performance. *Procedia Computer Science*, 1(2):2811 – 2819, 2010.

[13] T. Winters, C. Shelton, T. Payne, and G. Mei. Topic extraction from item-level grades. In J. Beck, editor, *American Association for Artificial Intelligence 2005 Workshop on Educational Datamining*, Pittsburgh, PA, 2005.

An Investigation of Psychometric Measures for Modelling Academic Performance in Tertiary Education

Geraldine Gray, Colm McGuinness, Philip Owende
Institute of Technology Blanchardstown
Blanchardstown Road North
Dublin 15, Ireland
geraldine.gray@itb.ie

ABSTRACT

Increasing college participation rates, and a more diverse student population, is posing a challenge for colleges in facilitating all learners achieve their potential. This paper reports on a study to investigate the usefulness of data mining techniques in the analysis of factors deemed to be significant to academic performance in first year of college. Measures used include data typically available to colleges at the start of first year such as age, gender and prior academic performance. The study also explores the usefulness of additional psychometric measures that can be assessed early in semester one, specifically, measures of personality, motivation and learning strategies. A variety of data mining models are compared to assess the relative accuracy of each.

Keywords

Educational data mining, academic performance, ability, personality, motivation, learning style, self-regulated learning.

1. INTRODUCTION

Factors impacting on academic performance have been the focus of research for many years and still remain an active research topic [5], indicating the inherent difficulty in defining robust deterministic models to predict academic performance, particularly in tertiary education [14]. Non progression of students from first year to second year of study continues to be a problem across most academic disciplines. This is particularly true of the Institute of Technology[1] sector in Ireland, where students have a weaker academic history than university students, and there is increasing numbers of non-standard students in the classroom [12]. This increase in participation poses a challenge to colleges to respond in a pro-active way to enable all leaners achieve their potential.

[1]The Institute of Technology sector is a major provider of third and fourth level education in Ireland, focusing on the skill needs of the community they serve (www.ioti.ie).

Educational Data Mining (EDM) is emerging as an evolving and growing research discipline in recent years, covering the application of data mining techniques to the analysis of data in educational settings [4, 17]. EDM has given much attention to date to datasets generated from students' behaviour on Virtual Learning Environments (VLE) and Intelligent Tutoring Systems (ITS), many of which come from school education [1]. Less focus has been given to college education, and in particular, to modelling datasets from outside virtual or online learning environments. This paper reports on the preliminary results of a study to analyse the significance of a range of measures in building deterministic models of student performance in college. The dataset includes data systematically gathered by colleges for student registration. The usefulness of additional psychometric measures gathered early in semester one is also assessed.

2. STUDY CRITERIA

In deciding on measures to include in the study, four key areas were reviewed: aptitude, personality, motivation and learning strategies. These were chosen firstly because research highlights these factors as being directly or indirectly related to academic performance [18], and secondly because these factors can be measured early in semester one. The following sections will report on correlations between individual factors and academic achievement, and also look at regression models of combinations of measures. All studies cited below are based on college education.

2.1 Influence of learner ability

There is broad agreement that ability is correlated to academic performance, although opinions differ on the range of sub factors that constitute ability [8]. For example, some studies have used specific cognitive ability tests to measure ability, for which there is extensive validity evidence. However such tests have been criticised with regarding to the objects of measurment. For example Sternberg 1999 [19] asserts that high correlation between cognitive intelligence scores and academic performance is because they measure the same skill set rather than it being a causal relationship. Therefore many studies use data already available to colleges to measure ability, i.e. grades from 2nd level education or SAT/ACT (Scholastic Aptitude Test / American College Testing) scores [18]. In a meta analysis of 109 studies by Robbins et al 2004 [16] prior academic achievement based on high school GPA or grades was found to have moderate correlation with academic performance (90% CI: 0.448 $\pm$ 0.4,). Average correlation between SAT scores and academic performance was lower (90% CI: 0.368± 0.035).

2.2 Influence of personality

Factor analysis by a number of researchers, working independently and using different approaches, has resulted in broad agreement of five main personality dimensions, namely openness, agreeableness, extraversion, conscientiousness and neuroticism, commonly referred to as the Big Five [9]. Of the five dimensions, conscientiousness is the best predictor of academic performance [20]. For example, Chamorro et al 2008 [6] reported a correlation of 0.37 with academic performance ($p<0.01$, $n=158$). Openness is the second most significant personality factor, but results are not as consistent. Chamorro et al 2008 [6] reported a correlation of 0.21 ($p<0.01$, $n=158$) between openness and academic performance. However the strength of the correlation is influenced by assessment type, with open personalities doing better where the assessment method is not restricted by rules and deadlines [10]. Studies on the predictive validity of other measures of personality are inconclusive [20].

2.3 Motivational factors

Motivation is explained by a range of complementary theories, which in turn encompass a number of factors, some of which have been shown to be relevant, directly or indirectly, to academic performance [16]. Factors relevant to academic performance in college include achievement motivation (drive to achieve goals), self-efficacy and self-determined motivation (intrinsic and extrinsic motivation). In Robbins et al 2004 meta analysis of 109 studies, self-efficacy and achievement motivation were found to be the best predictors of academic performance [16]. Correlations with self-efficacy averaged at 0.49 ±0.05 (CI: 90%), correlations with achievement motivation averaged at 0.303 ± 0.04 (CI: 90%). Self-determined motivation is not as strong a predictor of academic performance [11].

2.4 Influence of learning strategies

The relationship between academic performance and personality or motivation is mediated by a students approach to the learning task. Such learning strategies include both learning style (such as deep, strategic or shallow learning approach) [6] and learning effort or self-regulation [13].

Analysing the influence of learning style directly on academic performance, some studies show higher correlations with a deep learning approach [6], while others cite marginally higher correlations with a strategic learning approach [5]. The difference in these results can be explained, in part, by the type of knowledge being tested for in the assessment itself [21]. Many studies argue that there is a negative correlation between a shallow learning approach and academic performance [5].

Self-regulated learning is recognised as a complex concept to define, as it overlaps with a number of other concepts including personality, learning style and motivation, particularly self-efficacy and goal setting [2]. For example, while many students may set goals, being able to self-regulate learning can be the difference between achieving, or not achieving, goals set. Violet 1996 [21] argued that self-regulated learning is more significant in tertiary level than earlier levels of education because of the shift from a teacher-controlled environment to one where a student is expected to manage their own study. A longitudinal tertiary level study by Ning and Dowling 2010 [13] investigating the interrelationships between motivation and self-regulation found while both had a significant influence on academic performance, motivation was a stronger predictor of academic performance.

2.5 Combining psychometric measures

Individually the factors discussed above are correlated with academic performance. Also of relevance is how much of the variance in academic performance they account for. Cassidy 2011 [5] accounted for 53% of the variance in a regression model including prior academic performance, self-efficacy and age ($n=97$, mean age=23.5). Chamorro-Premuzic et al 2008 [6] accounted for 40% of the variance in a regression model including ability, personality factors and a deep learning strategy ($n=158$, mean age=19.2). A similar variance was reported by Dollinger et al 2008 [7] (44%) in a regression model including prior academic ability, personality factors, academic goals and study time ($n=338$, mean age=21.9). However not all studies concur with these results. In a study of non-traditional students, Kaufmann et al 2008 [11] accounted for 14% of the variance in a model with prior academic performance, personality factors and self-determined motivation ($n=315$, mean age=25.9). This suggests that models based on standard students may not be applicable to a more diverse student population.

There is clearly overlap, either correlation or causal, between the factors discussed above. A dataset that includes these measures will have a complex pattern of interdependencies. This raises questions on how best to model this type of data, what dimensions are useful to include, and how consistent are results across student groups.

3. THE STUDY DATASET

This study was based on first year students at an Institute of Technology in Ireland. 30% of first year students took part ($n=713$) based on their participation in online profiling during induction. Data was gathered over two years, September 2010 and September 2011. Students were from a variety of academic disciplines including computing, engineering, humanities, business and horticulture. The age range was [18,60] with an average age of 23.75. Average CAO Points[2] was 257.9 ± 75. 59% of the students were male.

The data was compiled from three sources:

1. Prior knowledge of the student: Table 1 lists attributes used from data available following student registration. This includes age, gender and ability measures based on prior academic performance. Access to full time college courses in Ireland is based on academic achievement in a set of state exams at the end of secondary school. Students have a grade for Maths, English, a foreign language, and four additional subjects chosen by the student. These subjects were categorised as science, humanities and creative / practical.

[2]CAO Points are a measure of prior academic performance in Ireland, range [0,600].

2. Psychometric data: Table 2 lists the additional measures used which were assessed using an online questionnaire developed for the study (www.howilearn.ie). The questionnaire was completed during first year induction. It covered measures of personality, motivation, learning style and self-regulated learning. Questions are taken from openly available, validated instruments.

3. Academic performance in year 1: A binary class label was used based on end of year GPA score, range [0-4]. GPA is an aggregated score based on results from between 10 and 12 modules delivered in first year. The two classes were poor academic achievers who failed overall (GPA<2, n=296), and strong academic achievers who achieved honours overall (GPA≥2.5, n=340). To focus on patterns that distinguish poor academic achievers from strong academic achievers, students with a GPA of between 2.0 and 2.5 were excluded, giving a final dataset of (n=636).

4. RESULTS

To date, six algorithms have been used to train models on the dataset, Support Vector Machine(SVM), Neural Network(NN), k-Nearest Neighbour (k-NN), Naïve Bayes, Decision tree and Logistic Regression, using RapidMiner V5.2 (rapid-i.com). Models were run on the full dataset, and also on two subgroups of the dataset split by age. An age boundary of 21 was chosen because the majority of under 21s had a poor academic performance (61%), whereas the majority of over 21s had a strong academic performance (70%). All datasets were balanced by over sampling the minority class, the attributes were scaled to the range [0,1], and model accuracy was assessed using cross validation (k=10). Model accuracy for data available at registration only (prior) was compared to model accuracy when psychometric data was included (all). Results and model parameters are detailed in Table 3.

When modelling all students using prior attributes only, pairwise comparison of the mean accuracies using least significant difference ($d_{lsd} = 5.07, p = 0.05$) indicates model performance was comparable across learners, with the only significant pairwise difference being between Naïve Bayes (75.74%) and SVM (69.41%). Model accuracies changed marginally when psychometric variables were included. The most notable change was the SVM model, where model accuracy increased from 69.41% to 75% ($t(18) = 2.05$, $p = 0.055$). The accuracy of most models improved when trained on younger students only. Again the most notable change was the SVM model. It increased from 69.41% to 82.62%, which was significant ($t(18) = 3.53, p = 0.0024$). Model accuracies changed marginally when psychometric variables were included for this group.

Models trained on older students were the least accurate with the exception of an SVM model using all attributes, which achieved the highest accuracy of all models at 93.45%. Accuracies for Decision Tree and Logistic Regression were particularly poor, with models performing no better than random guessing. In this student group, prior academic performance was not available for 38% of the students (n=108), explaining the improvement in accuracies when psychometric attributes were included in the models. When students with missing data were removed from this group, including psychometric measures did not improve model accuracies significantly for any of the models. SVM again had the highest accuracy at 91%, while Neural Networks achieved similar accuracies to the under 21 student group.

Table 1: Data available from the college

Learner Ability measures, mean and standard deviation:	
Aggregate Mark (CAO points) (258±76)	
English Result (45.5±18)	Maths Result (23.8±14)
Highest Result (64.7±14)	Humanities Average(39.7±13)
Science Average (31.8±16)	Creative Average (47.9±19)
Other factors:	
Age (23.75±7.6)	Gender (m=440, f=304)

Note: Range for age is [18,59], valid range for CAO points is [0,600], valid range for other values is [0,100]

Table 2: Additional measures*

Personality, Goldbergs IPIP scales (http://ipip.ori.org):	
Conscientousness (5.9±1.5)	Openness (6.35±1.3)
Motivation, MSLQ [15]:	
Intrinsic Goal Orientation(7.1±1.4)	Self Efficacy (6.8±1.5)
Extrinsic Goal Orientation (7.8±1.4)	
Learning style, based on R-SPQ-2F [3]:	
Shallow Learner (1.4±2)	Deep Learner (5.2±2.9)
Strategic Learner (3.5±2.5)	
Self-regulated Learning, MSLQ [15]:	
Self Regulation (5.8±1.3)	Study Effort (5.9±1.7)
Study Time (6.2±2.3)	

*The mean, standard deviation, and instrument from which questions were sourced are given for each measure. Valid range for all measures is [0,10]

5. CONCLUSION

Results from this study show that models of academic performance in tertiary education can achieve good predictive accuracy, particularly if younger students and mature students are modelled separately. This suggests that patterns are different for standard versus non-standard students. The preliminary analysis has demonstrated that good accuracy can be achieved based on data already available to colleges. Including additional psychometric measures improves predictive accuracy for mature students, but the evidence so far suggests this is due to missing data regarding prior academic performance rather than the additional added value of the psychometric measures themselves.

The most accurate models were SVMs trained on under 21s and over 21s separately. In general, models that can learn more complex patterns, and handle high dimensionality, are getting higher accuracies for both student groups. The difference in accuracy across models is most pronounced for mature students, suggesting the patterns in that subgroup are more complex. When training a single model for all students, including students for whom prior academic results are not available (i.e. the quality of the dataset is reduced), models give comparable accuracy (73.82%±1.8).

The results published here represent early results from the study. Further analysis of the psychometric measures is required to determine their predictive value, and also their usefulness in understanding how the profile of students who fail

Table 3: **Model accuracies (% mean and st. dev)**

Dataset	Attri-butes	SVM*	NN**	k-NN (k=16)	Naïve Bayes	Decision Tree	Log Reg
Full	prior	69.41	74.32	73.97	75.74	72.94	72.01
(n=636)		±7.11	±4.72	±4.46	±5	±5.78	±6.41
	all	75	75.33	74.85	72.35	74.56	70.84
		±4.88	±7.99	±3.63	±6	±5.54	±3.60
Under21	prior	82.62	78.1	76.9	77.14	70	74.94
(n=350)		±9.47	±8.7	±4.77	±5.55	±4.42	±6.22
	all	80.71	78.1	79.05	79.29	69.76	76.45
		±10.4	±5.3	±5.41	±7.76	±4.89	±6.47
Over 21	prior	77.03	72.29	64.99	57.16	50.62	48.96
(n=286)		±9.56	±11.33	±7.33	±16.79	±1.41	±22.12
	all	93.45	77.31	71.7	57.16	50.62	52.47
		±4.41	±9.3	±4.86	±16.79	±1.41	±18.08
Over 21	prior	91.03	78.3	71.54	62	51.5	64.31
no miss-		±6.46	±11	±11.47	±11	±1.36	±10.71
ing data	all	91.03	79.6	70.69	69.26	51.5	66.26
(n=178)		±6.46	±12.12	±6.1	±5.67	±1.36	±10.07

prior: attributes available from the college, all: all attributes
*Anova kernel, epsilon=0.7 and C=1.
*Learning rate=0.7, and momentum=0.4, 2 hidden layers (10,5).

differs from those who do well. To date, two subgroups have been examined, split by age. Other subgroups will be reviewed, including an analysis of differences across academic disciplines, and an analysis of students with GPA between 2.0 and 2.5 (not included in this study). Finally, model accuracies will be verified by testing the models against a third year of data, students registered in September 2012.

6. ACKNOWLEDGEMENTS

The authors would like to thank the Institute of Technology Blanchardstown (www.itb.ie) for their support in facilitating this research, and the staff at the National Learning Network Assessment Services (www.nln.ie) for their assistance in administering questionnaires during student induction.

7. REFERENCES

[1] R. S. J. D. Baker and K. Yacef. The state of educational data mining in 2009" a review and future visions. *Journal of Educational Data Mining*, pages 3–17, 2010.

[2] T. Bidjerano and D. Y. Dai. The relationship between the big-five model of personality and self-regulated learning strategies. *Learning and Individual Differences*, 17:69 – 81, 2007.

[3] J. Biggs, D. Kember, and D. Leung. The revised two-factor study process questionnaire: R-spq-2f. *British Journal of Education Psyhcology*, 71:133–149, 2001.

[4] T. Calders and M. Pechenizkiy. Introduction to the special section on educational data mining. *SIGKDD*, 13(2):3–3, 2011.

[5] S. Cassidy. Exploring individual differences as determining factors in student academic achievement in higher education. *Studies in Higher Education*, pages 1–18, 2011.

[6] T. Chamorro-Premuzic and A. Furnham. Personality, intelligence and approaches to learning as predictors of academic performance. *Personality and Individual Differences*, 44:1596–1603, 2008.

[7] S. J. Dollinger, A. M. Matyja, and J. L. Huber. Which factors best account for academic success: Those which college students can control or those they cannot? *Journal of Research in Personality*, 42:872–885, 2008.

[8] D. P. Flanagan and K. S. McGrew. Interpreting intelligence tests from contemporary gf-gc theory: Joint confirmatory factor analysis of the wj - r and kait in a non - white sample. *Journal of School Psychology*, Vol. 36, No. 2:151 – 182, 1998.

[9] L. R. Goldberg. The development of markers for the big-five factor structure. *Psychological Assessment*, 4 (1):26–42, 1992.

[10] R. Kappe and H. van der Flier. Using multiple and specific criteria to assess the predictive validity of the big five personality factors on academic performance. *Journal of Research in Personality*, 44:142–145, 2010.

[11] J. C. Kaufman, M. D. Agars, and M. C. Lopez-Wagner. The role of personality and motivation in predicting early college academic success in non-traditional students at a hispanic-serving institution. *Learning and Individual Differences*, 18:492 – 496, 2008.

[12] O. Mooney, V. Patterson, M. O'Connor, and A. Chantler. A study of progression in higher education: A report by the higher education authority. Technical report, Higher Education Authority, Ireland, October 2010.

[13] H. K. Ning and K. Downing. The reciprocal relationship between motivation and self-regulation: A longitudinal study on academic performance. *Learning and Individual Differences*, 20:682–686, 2010.

[14] Z. A. Pardos, R. S. J. D. Baker, S. M. Gowda, and N. T. Heffernan. The sum is greater than the parts: Ensembling models of student knowledge in educational software. *SIGKDD Explorations*, 13(2), 2011.

[15] P. Pintrich, D. Smith, T. Garcia, and W. McKeachie. A manual for the use of the motivated strategies for learning questionnaire. Technical Report 91-B-004, The Regents of the University of Michigan, 1991.

[16] S. B. Robbins, K. Lauver, H. Le, D. Davis, and R. Langley. Do psychosocial and study skill factors predict college outcomes? a meta analysis. *Psychological Bulletin*, 130 (2):261–288, 2004.

[17] C. Romero and S. Ventura. Educational data mining: A survey from 1995 to 2005. *Expert Systems with Applications*, 33:135–146, 2007.

[18] N. Schmitt, F. L. Oswald, T. Pleskac, R. Sinha, and M. Zorzie. Prediction of four-year college student performance using cognitive and noncognitive predictors and the impact on demographic status of admitted students. *Journal of Applied Psychology*, 2009.

[19] R. Sternberg. Intelligence as developing expertise. *Contemporary Educational Psychology*, 24:359 – 375, 1999.

[20] A. B. Swanberg and Ø. L. Martinsen. Personality, approaches to learning and achievement. *Educational Psychology*, 30(1):75–88, 2010.

[21] S. E. Volet. Cognitive and affective variables in academic learning: the significance or direction and effort in students' goals. *Learning and Instruction*, 7(3):235–254, 1996.

Modeling Affect in Student-driven Learning Scenarios

Paul Salvador Inventado[*] , Roberto Legaspi, Rafael Cabredo[*], and Masayuki Numao
The Institute of Scientific and Industrial Research, Osaka University
8-1 Mihogaoka, Ibaraki, Osaka, Japan, 567-0047
{inventado,cabredo,roberto}@ai.sanken.osaka-u.ac.jp, numao@sanken.osaka-u.ac.jp

ABSTRACT

Much research has been done on affect detection in learning environments because it has been reported to provide better interventions to support student learning. However, students' actions inside these environments are limited by the system's interface and the domain it was designed for. In this research, we investigated a learning environment wherein students had full control over their activities and they had to manage their own goals, tasks and affective states. We identified features that would describe students' learning behavior in this kind of environment and used them for building affect models. Our results showed that although a general affect model with acceptable performance could be created, user-specific affect models seemed to perform better.

Keywords

affect modeling, educational data mining, student-driven learning

1. INTRODUCTION

The current society and workplace is dynamic and requires people to continuously learn new skills and adapt to what is needed. In order to prepare students for this kind of environment, they need to learn how to manage their learning goals, their time, their motivation and their affective states in environments wherein they receive little or no support and they have complete control over their learning.

Self-regulated learners are likely to be capable of adapting to such environments because they can effectively manage the different aspects of a learning scenario. One of the most important yet difficult skills to learn in self-regulation is monitoring one's cognitive and affective states. Knowledge of one's thoughts and affective states helps students evaluate the current situation and identify if it is better to continue with the current activity or change it. When students learn on their own, it is likely that they will spend time doing non-learning related activities. These do not always serve as distractions because they have also been shown to help regulate emotions [12]. Self-monitoring becomes essential in this case because students need to identify when and how much time spent in non-learning related activities is acceptable so that they can still achieve their learning goals. Self-monitoring is not easy because it is a complex meta-cognitive activity requiring much attention and sophisticated reasoning [13]. Learning in complex domains increases cognitive load and makes self-monitoring even more difficult.

In this research we are moving towards the creation of systems that can help students self-monitor by automatically detecting their affective states. It will be helpful for students to be informed about their affective states when they experience high cognitive load so that they can change their behavior accordingly. Such systems can also suggest activities to help them learn better (e.g., refer to notes, seek help) when certain affective states are detected. Another goal of the research is to use a data collection methodology that does not disrupt students' usual learning behavior. The succeeding sections discuss our methodology, the data we used, our affect model creation process and our results.

[*]also affiliated with: Center for Empathic Human-Computer Interactions, College of Computer Studies, De La Salle University, Manila, Philippines

2. RELATED WORK

Many researchers have tried improving existing learning systems by incorporating affect detection for better feedback. D'Mello et al. [7] for example developed affect models using data from students' interactions with a conversational agent in the domain of computer literacy. The features they used for building these models were based on students' responses, the correctness of the students' answers, their progress and the type of feedback provided by the system. The model they built to distinguish each affective state from each other did not perform very well (i.e., Kappa = 0.163), however the models they built for distinguishing affective states from a neutral state performed better (i.e., Kappa = 0.207 - 0.390).

Baker et al. [1] also developed affect models for students using Cognitive Tutor Algebra. They used features that described students' actions, the correctness of their actions and their previous actions. They built affect models which distinguished one affective state from another (e.g., bored vs. not bored, frustrated vs. not frustrated) whose resulting Kappa values ranged from 0.230 to 0.400.

Our work differs from previous research because we built affect models using data from students who were not limited to learning in a certain domain, who controlled their own learning, who did not receive feedback and there was no information regarding their learning progress.

3. LEARNING BEHAVIOR DATA

In our previous work, we collected data from one male undergraduate student, one male master's student and two female doctoral students who engaged in research activities as part of their academic requirements [9]. The students were aged between 17 and 30 years old wherein three of them were taking Information Science while one doctoral student was taking Physics. During the data collection period, two of the students were writing conference papers and two made power point presentations about their research. Students had control over how they conducted their learning activities and did not receive any direct support from their supervisor. These conditions required all students to manage their own cognitive and affective states as they learned which satisfied our target learning scenario.

Data was collected in five separate two-hour learning episodes from each student over a span of one week. Students freely decided on the time, location and type of activities they did but were required to learn in front of a computer that recorded their learning behavior. All students used a computer in doing their research so the setup was naturalistic and they did not have to change the ways in which they usually learned.

Data about the students' learning behavior was collected by asking them to annotate their behavior after each learning episode using a behavior recording and annotation tool we developed called Sidekick Retrospect [9]. At the beginning of a learning episode, students inputted their learning goals. The system then began logging the applications they used, taking screenshots of their desktop and capturing image stills from their webcam with corresponding timestamps. After a learning episode, students were presented a timeline which showed the desktop screenshots and image stills depending on the position of their mouse on the timeline. This helped students recall what happened during the learning episode so they could annotate it.

Students made annotations by selecting a time range and inputting their *intentions*, *activities* and *affective state*. Intentions can either be goal related or non-goal related relative to the goals set at the beginning of the learning episode. Activities referred to any activity done while learning which could either be done on the computer (e.g., using a browser) or out of the computer (e.g., reading a book). Two sets of affect labels were used for annotating affective states wherein goal-related activities were annotated as delighted, engaged, confused, frustrated, bored, surprised or neutral and non-goal related activities were annotated as delighted, sad, angry, disgusted, surprised, afraid or neutral. Academic emotions [4] were used for annotating goal related intentions because they gave more contextual information about the learning activity. However, academic emotions might not have captured other emotions outside of the learning context so Ekman's basic emotions [8] were used to annotate non-goal related intentions.

Students would inherently recall what happened during a learning episode when they made annotations so it would be easier for them to identify the appropriate labels. Going through the entire learning episode sequentially would also help them annotate more accurately because they would see how and why their activities changed as well as its outcomes. It is possible that students might not annotate the data correctly for fear of judgment or getting lower scores. However, in our experiment we made it clear to the students that their learning behavior would not affect their grades in any way and assured them that these would not be shown or discussed with their supervisors.

The students' annotations were processed and cleaned so that contiguous annotations had a different intention, activity or affective state. Those that were exactly the same were merged. The resulting data consisted of 1,081 annotations from all students with an average of 54.05 annotations (N=20; σ=27.18) in each learning episode.

4. FEATURE ENGINEERING

The data consisted of only three features (i.e., timestamp, intention and activity) and the affective state label for creating affect models. Models built using these initial features performed poorly so new features had to be designed.

Although the students worked on different topics and used different applications, all of them processed and performed experiments on previously collected data, searched for related literature and created a report or document about it. Although students performed many different activities, analyzing the data showed that these activities can be categorized into six general types – information search (e.g., using a search engine), view information source (e.g., reading a book, viewing a website), write notes, seek help from peers (e.g., talking to a friend), knowledge application (e.g., paper writing, presentation creation, data processing) and off-task (e.g., playing a game). This was used as a new feature which we called task. We also added as features the duration of the task and its position in the learning episode. A task's position in the learning episode was expressed as a normalized time value ranging from zero to 100, wherein zero indicated the start of the episode, 50 indicated the middle of the episode and 100 indicated the end of the episode.

Previous research has shown that the occurrence and duration of previous cognitive or affective states influenced the student's current affective state [2, 9]. However, there is no study that describes how long their influences last. For this study, we only considered the effects of cognitive and affective states in the last five minutes which was based on the average duration of tasks in our data. Similarly, there was no study indicating how many elements in a sequence of previous tasks influenced the current task. However, the data showed that students performed only a maximum of five tasks within a five minute interval (i.e., when students quickly shifted from one task to another). So, we considered the past five tasks relative to the current task as features.

To express the relationship between the previous and current tasks, we used *task frequency* (i.e., the number of times a certain type of task was performed in the last five minutes), *task duration* (i.e., the number of seconds each type of task

Table 1: Affect model performance

Classifier	Kappa	F-measure	Accuracy
Naïve Bayes	**0.345**	**0.349**	**63.10%**
J48 (C4.5)	0.333	0.290	62.57%
JRip	0.326	0.331	61.22%
SVM	0.286	0.351	59.91%
Rep-tree	0.284	0.362	59.06%
Bayesian Network	0.181	0.317	39.43%

was performed in the last five minutes), *most frequent task* (i.e., the most frequent type of task in the last five minutes) and *dominant task* (i.e., the type of task that was performed for the longest time in the last five minutes).

The new set of 22 features was used with the affect label for affect modeling.

5. AFFECT MODELING

Rapidminer 5.3 [11] was used in running different machine learning algorithms to build the affect models. Batch cross-validation was used for evaluating the models such that all the data from one student was held out for testing each time. This was used to test if the resulting model would generalize over students.

RapidMiner's genetic algorithm feature selector was used to identify the most relevant features for the classification task. The fitness of each feature subset was calculated by running a given machine learning algorithm on that subset and then using the resulting model's batch cross-validated kappa value. Cohen's Kappa [3] was used because it considers misclassifications of multiple class labels which cannot be handled by other measures like accuracy. Kappa has also been used frequently in classifying educational data [1, 7].

Table 1 shows the results of the evaluation where the Naïve Bayes model gave the highest kappa value of 0.345 using the features selected by the feature selector. This indicates that the model can perform around 34% better than chance.

The feature selector used 12 out of the 22 features for building the affect models. Three of these features were related to the student's current state (i.e., position in the learning episode, duration and task). Five of the features were related to the past tasks employed by the student (i.e., task$_{n-1}$ ... task$_{n-5}$). Two of the features were related to the amount of time spent performing a previous task in the last five minutes (i.e., information search duration and write notes duration). Finally, two features were related to the frequency of performing tasks in the last five minutes (i.e., apply knowledge frequency and off-task frequency).

Just like other research, features related to previous actions were also found to correspond with the current affective state [2, 6, 9, 12] and is probably the reason why these were selected. The task feature was probably selected because some affective states occurred more frequently while performing a particular task. For example, confusion and engagement were commonly associated with knowledge application and viewing information sources most likely because these activities require utilizing current knowledge and understanding new information. However, boredom and neutral affective states were commonly experienced only when viewing information sources probably because unlike knowledge application, some information sources may not have been relevant to the student. Delight was usually experienced when students performed off-task activities probably because students engaged in activities they enjoyed during this period.

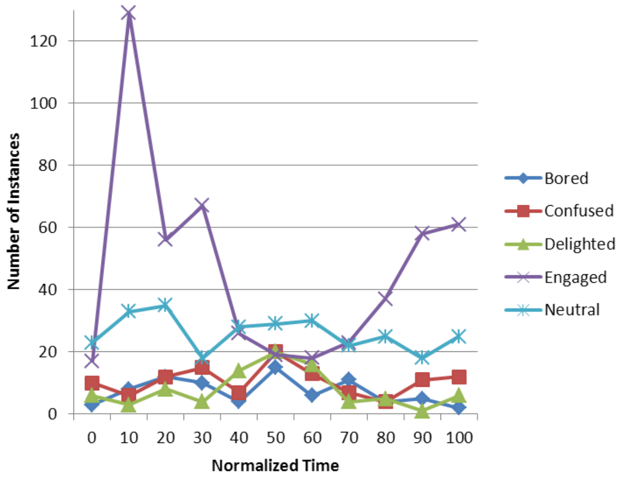


Figure 1: Occurrence of affective states over time

Table 2: Kappa values of user-specific affect models

	J48	JRip	Rep-Tree	SVM	BN	NB
1	**0.675**	0.665	0.628	0.625	0.604	0.310
2	**0.270**	0.147	0.166	0.164	0.229	0.206
3	**0.498**	0.490	0.400	0.425	0.414	0.375
4	**0.532**	0.445	0.484	0.529	0.300	0.256

Certain affective states were experienced more frequently at certain positions in the learning episode which might have caused it to be selected as a feature. Figure 1 shows the total number of times an affective state was experienced by students at certain points of the learning episode. Engagement was experienced more frequently at the start and at the end of the session while the occurrence of confusion and delight increased in the middle of the episode. This is indicative of the phases of flow wherein a learner starts in an equilibrium state of understanding, which is challenged by new knowledge usually exhibited by feelings of confusion and is later assimilated leading back to an equilibrium state [5, 10].

We also investigated the performance of user-specific affect models by using each student's data separately. The models were evaluated with batch cross-validation using the session number to see if it generalized over learning sessions.

Table 2 shows that the kappa values of the user-specific affect models were higher compared to the general affect model. Among all machine learning algorithms, J48 performed best with a Kappa value of 0.675. The feature selector selected features similar to those in the general affect model with subtle differences in the features related to the frequency and duration of previous tasks. For example, in one student's affect model, the frequency of information searches in the past five minutes was selected as a feature while the frequency of writing notes in the past five minutes was selected instead in another student's model. These are indicative of students' affective states being influenced differently by certain tasks. This also shows that individual differences play a part in the affective states experienced by a student making them behave differently in similar contexts.

The features selected in both the general and user-specific models described the frequency, duration and type of previous actions performed by the students as well as the students' current learning state. These are contextual information about the students' learning state which seems to be good predictors of students' affective states as shown by the performance of the resulting affect models.

6. CONCLUSION

In this paper, we have presented the development of affect models that are capable of identifying students' affective states. The features used described the context in which the student learned such as the previous and current tasks they performed and are currently doing. The novelty of our work is that the affect models we built could identify affective states in a learning environment wherein students were not bound by a particular domain or learning system and the students had complete control over their activities. Even though information regarding the students' progress was unavailable, the performance of the models was still acceptable. Our results could not directly be compared to previous work because the affective states predicted by our models were in the context of a particular task unlike the works of Baker et al. [1] and D'Mello et al. [7] that predicted affective states in particular time intervals. However, the approach seems promising because the performance of the model was almost as good as the results in these works.

We acknowledge that the general affect model was created using only a few participants. However, the important observation we got was that user-specific models had better results indicating the importance of individual differences in building affect models. Evaluating the performance of affect models using data from more students would help confirm such findings.

There is a need to find features that could increase the performance of these affect models and experiment on different thresholds (e.g., task frequencies and durations in either less than or more than five minutes prior to the current task)

Affect models built with our methodology can be used by other systems for monitoring affective states. Students can then be made aware of their affect through prompts so they can adapt their activities accordingly. These systems could also suggest changes to particular tasks when certain affect is detected. Enabling systems to help students self-monitor can help them self-regulate and thus learn better.

Acknowledgements

This work was supported in part by the Management Expenses Grants for National Universities Corporations from the Ministry of Education, Culture, Sports, Science and Technology of Japan (MEXT) and JSPS KAKENHI Grant Number 23300059. We would also like to thank all the students who participated in our data collection.

7. REFERENCES

[1] R. S. Baker, S. M. Gowda, M. Wixon, J. Kalka, A. Z. Wagner, A. Salvi, V. Aleven, G. W. Kusbit, J. Ocumpaugh, and L. Rossi. Towards Sensor-Free affect detection in cognitive tutor algebra. In *Proceedings of the International Conference on Educational Data Mining*, pages 126–133, 2012.

[2] R. S. Baker, M. M. Rodrigo, and U. E. Xolocotzin. The dynamics of affective transitions in simulation Problem-Solving environments. In *Proceedings of the 2nd international conference on Affective Computing and Intelligent Interaction*, ACII '07, pages 666–677, 2007.

[3] J. Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46, 1960.

[4] S. D. Craig, A. C. Graesser, J. Sullins, and B. Gholson. Affect and learning: An exploratory look into the role of affect in learning with AutoTutor. *Journal of Educational Media*, 29(3):241–250, 2004.

[5] M. Csikszentmihalyi. *Flow : the psychology of optimal experience.* Harper & Row, 1990.

[6] S. D'Mello and A. Graesser. Dynamics of affective states during complex learning. *Learning and Instruction*, 22(2):145–157, 2012.

[7] S. K. D'Mello, S. D. Craig, A. Witherspoon, B. Mcdaniel, and A. Graesser. Automatic detection of learner's affect from conversational cues. *User Modeling and User-Adapted Interaction*, 18(1):45–80, 2008.

[8] P. Ekman. Are there basic emotions? *Psychological Review*, 99(3):550–553, 1992.

[9] P. S. Inventado, R. Legaspi, R. Cabredo, and M. Numao. Student learning behavior in an unsupervised learning environment. In *Proceedings of the 20th International Conference on Computers in Education*, pages 730–737, 2012.

[10] B. Kort, R. Reilly, and R. W. Picard. External representation of learning process anddomain knowledge: affective state as a determinate of its structure and function. In *Artificial Intelligence in Education*, pages 64–69, 2001.

[11] I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, and T. Euler. YALE: Rapid prototyping for complex data mining tasks. In L. Ungar, M. Craven, D. Gunopulos, and T. Eliassi-Rad, editors, *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 935–940, 2006.

[12] J. Sabourin, J. Rowe, B. Mott, and J. Lester. When Off-Task is On-Task: The affective role of Off-Task behavior in Narrative-Centered learning environments. In G. Biswas, S. Bull, J. Kay, and A. Mitrovic, editors, *Artificial Intelligence in Education*, volume 6738 of *Lecture Notes in Computer Science*, pages 534–536. 2011.

[13] B. J. Zimmerman. Self-regulated learning and academic achievement: An overview. *Educational psychologist*, 25(1):3–17, 1990.

An Algorithm for Reducing the Complexity of Interaction Networks

Matthew W. Johnson
University of North Carolina at Charlotte
Charlotte, NC
mjokimoto@gmail.com

Michael Eagle
University of North Carolina at Charlotte
Charlotte, NC
maikuusa@gmail.com

John Stamper
Carneige Mellon University
Pittsburgh, PA
john@stamper.org

Tiffany Barnes
North Carolina State University
Raleigh, NC
tmbarnes@ncsu.edu

ABSTRACT

We present an algorithm for reducing the size and complexity of the Interaction Network, a data structure used for storing solution paths explored by students in open-ended multi-step problem solving environments. Our method reduces the number of edges and nodes of an Interaction Network by an average of 90% while still accounting for 40% of actions performed by students, and preserves the most frequent half of solution paths. We compare our method to two other approaches and demonstrate why it is more effective at reducing the size of large Interaction Networks.

1. INTRODUCTION

One major advantage of computer based tutoring systems, for multi-step problems, is the ability to log data showing 'how' students solved problems, and their mistakes, an aspect not often found in traditional paper based homework. However providing educators a method of understanding the data logged by these systems, efficiently so that it can be acted upon, remains challenging. One approach is to provide a visualization tool to display the solutions to educators, to facilitate gaining insights into how students solved their open-ended multi-step problems. We define open-ended problems, as problems with at least two differing solution paths requiring multiple steps to complete. These types of problems are often seen in Intelligent Tutoring Systems (ITS) and similar computer based instruction, like the Deep Thought Logic Tutor[3].

The interaction network is a network similar to a state-space for tutors, built from data-logs, which leverages student information. One important challenge facing the Interaction Network and InVis, the tool built for exploring those networks, is the size of the network, often resulting in thousands of nodes and edges for roughly a hundred students worth of data. These large networks make it difficult to retrieve a general overview and understanding of student solutions. We present a reduction algorithm that drastically reduces the number of nodes in the network, allowing users to focus on the common approaches used by students to solve problems.

We compare the reduction of nodes and edges between our reduced network and the original, as well as other metrics, like the percent coverage of student solutions. We also compare our approach to two alternative filtering processes to show the benefits of our method. We provide a set of domain experts with one of the problems from the Deep Thought logic tutor and ask they describe the different approaches students use to solve the problem, as well as the common mistakes. We compare the expert provided solutions to the reduced Interaction Network to confirm whether or not our method has appropriately captured the solution paths.

We show that our proposed reduction algorithm when applied to the Interaction Network successfully reduces the number of nodes of the Interaction Network by between 86 and 96 percent, while preserving the solution traces to an average of 52 percent of the goals and 40 percent of the student action frequencies. Furthermore, our method preserves all the solutions suggested by experts. These types of reduced networks could aid in providing a more efficient means of understanding student behaviors, mainly by limiting the network to the most important solutions. When combined with InVis, this could provide an efficient method of understanding how students solved problems in computer based systems, potentially providing a useful role for the educator in their course, by providing a better understanding of student solutions.

2. RELATED WORK

Others have looked at reducing the state space in learning environments for purposes of improving intelligent tutor efficiency and improving interpretation of the data for use by course developers and instructors [9]. Our work differs, as we focus on clustering different student solutions to complex problems in order to reduce the space of student strategies.

The source for our student data is the Deep Thought logic tutor. This tutor allows students to solve first-order prepositional logic problems[3]. Students are provided a set of premises and are challenged with deriving a conclusion. By applying a set of different logic axioms, students can either work from the premises towards the conclusion or alternatively, Deep Thought allows students to work backwards, from the conclusion towards the premises. The Interaction Network can be applied to an individual problem.

Sudol et al. describe a method of generating a similar state space that we use here, but for the domain of programming. In their work, they present the probability distance metric for states in programming problems for introductory students [10]. Menzel and Le have also focused on exploring the state-space of 'ill-defined' domains, using a constraint based system[7]. Mitrovic explores open-ended problems in their web-based SQL tutor which is a constraint based system, but their system has also incorporated a student model to aid users [8].

From our experience with earlier versions of the InVis tool, large networks made it difficult for educators and researchers to efficiently decipher the types of solutions students are using to solve problems from the Deep Thought logic tutor. In our previous work, users explored networks for nearly 20 students at a time. However, our goals for the InVis tool are to make understanding student solutions efficient, which can be achieved by viewing more students at a time. An advantage to looking at more students at once is, it can be easier for users to compare different solutions, as they will not have to maintain those different solutions in their working memory but can quickly make comparisons based on the visualization. Finally, the visualization research community provides us with the Visual Analytics mantra, which argues when there is too much data, visualizations should leverage the machine to analyze the data, identify and present the important features of the data, rather than providing an overview of all the data and relying on the user to filter[2].

An Interaction Network is a model of the state space which includes student information on edges and nodes. It is a connected, directed, labeled multi-graph with states as vertices, actions as directed-edges to connect the states. The Interaction Network stores the set of all students who visited any particular state-vertex or action-edge, allowing us to count frequencies and connect other information, like test scores or hint usage values, to the Interaction Network representation. A detailed description of the Interaction Network is provided in previous work[5].

3. REDUCTION ALGORITHM

Data sets containing many student solution attempts can create large state spaces. One of the goals of InVis is to provide an efficient understanding of common student behaviors. However, exploring networks with thousands of nodes can be slow, and is also subject to hardware limitations. In our experience, even professional software tools for viewing graphs start to slow down when the node counts exceed 1500, on typical PC hardware. We developed an algorithm for reducing the network by roughly 90%, while preserving important information.

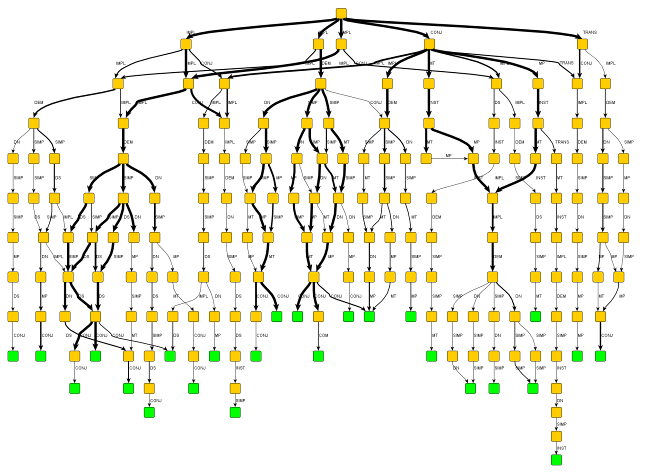


Figure 1: This is the reduced network for problem 3-5. This reduced network contains 186 nodes (85% reduction) and 219 edges (88% reduction).

The purpose of this algorithm is to maximize the amount of information we can gain from the data, while minimizing the number of nodes and edges, to make common approaches more clear. We also want to be as close as possible to a directed simple graph. A simple directed graph is defined as a graph containing no loops or parallel edges. Our assumption is that simple graphs are easier to read when following state-transitions, because they have no parallel edges. Next we want to preserve as many paths from the problem start to the goals as possible, to retain as many student solutions as we can. We would also like to provide continuity and solution variations. Continuity in this case, implies the reduced network maintain complete solution paths, so the graph is understandable, as opposed to a list of the most frequent nodes. By providing variations to similar solutions we should be able to provide better estimations to the numbers of students who performed a particular solution. Without the context of the progression of the states, users would be unable to understand how the problems were solved. We want to provide a means for understanding how many students solved the problem, not just which actions were most frequent.

We will use four metrics for measuring our success.

1. Vertex and Edge Reduction Rates
2. Number of Goals
3. Number of Interactions
4. Average Student Frequency per Edge

The vertex and edge counts will inform us how well we reduced the number of states and actions, we aim for reduction in magnitude. Goal counts will let us know how many of the solution paths we have maintained, from start to finish. For this metric, not only the count is important, but to maintain continuity all goals must have a path from the start of the problem to the respective goal state. The sum of edge frequencies will inform us of the total number of

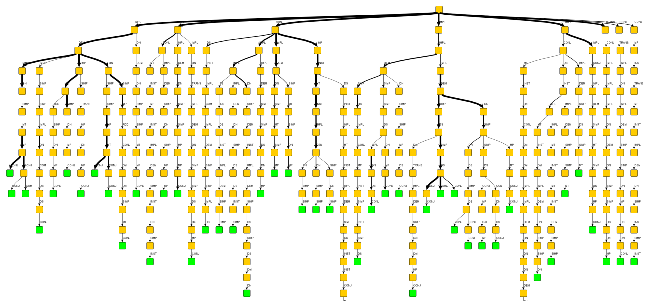


Figure 2: This is the problem 3-5 data set after the shortest paths reduction applied. This network contains 383 nodes(69% reduction), and 382 edges (79% reduction). Note, the two longest solution paths have been cropped in this image.

actions performed by all students, and in turn what percent of student actions are being preserved in our reduction. Arguably, edges with higher frequencies are more informative because more students performed that action, over an action performed by a fewer students. Lastly, the average student frequency per edge will give us an indicator of how important each edge is in the network.

We asked two professors with over a decade of experience teaching logic, and two graduate students who have either taught the course or performed a teaching assistant role to provide us with the set of solutions they expected to see from students. These four experts provided us with eight solutions total, four of which differed in direction or actions used to solve the problem. Problem 3-5 was chosen because it has one of the larger ranges of possible solutions in our problem set. We will use these provided solutions in comparison to the reduced network and compare how many of those solutions are preserved in the reduced network.

3.1 Algorithm

The idea for this algorithm is inspired by compression algorithms. We want to identify the edges with the highest frequencies and preserve them, then find goal states which are close to those paths. The Interaction Network for the problem 3-5 data set has 1252 nodes and 1835 edges. The proposed algorithm works by focusing on high frequency edges, of which there are few, and filtering out the low, and often frequency one edges, for which there are hundreds. This algorithm works by accepting three parameters, the Interaction Network on which to act upon, the percent of desired reduction, and a growth parameter. Prior to reduction, we first calculate a set of values in a pre-reduction step. In tutors which do not contain 'undo' actions, this step will not be necessary. To adjust for the behavior of moving forward, followed by an undo, we calculate a table of negative weights. For each state, an incoming action followed by an 'undo', will increment a negative weight counter for the incoming action. This will be used to devalue the frequency of these actions. Next we remove the 'undo' edges from the network, this reduces the number of cycles and parallel edges presumably making the flow of state-transitions in the Interaction Network easier to follow.

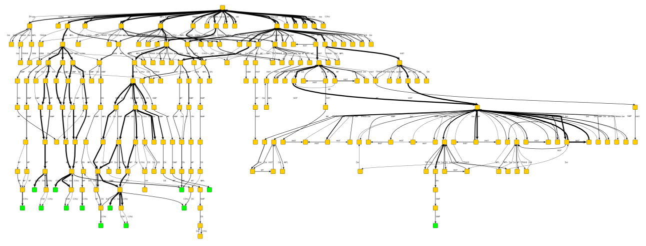


Figure 3: This is the problem 3-5 data set after the frequency one filter reduction is applied. This filtered network contains 235 nodes(81% reduction), and 400 edges(78% reduction).

Next we calculate the adjusted edge frequencies, which are equal to an edge's original frequency minus the weight calculated in the previous step. Now, the network is reduced using the percent reduction parameter. We aimed for an order of magnitude reduction and so this parameter was set to 10%. For the reduction step, we generate a new network using the edges with the top 10% of student frequencies, and their source and target nodes. Depending on the network a set of disjoint graphs will be created, we find the roots of each disjoint graph, which are the nodes with zero in-degree. We then calculate the shortest paths from the problem start to each disjoint-root, and inject the necessary edges and nodes to reconstruct a connected graph. Following this step we check the list of all goal nodes and attempt to connect any node in the reduced graph to any of the goal nodes, again using the shortest path in the original network. We use the growth parameter to limit the distance of the shortest path, for this work we used a value of ten. That is, if a goal node can be reached within ten edges, the path is added, otherwise it is ignored. As a final step, we attempt to connect all the nodes within the reduced graph to any other node in the reduced graph, again using shortest path. The reduced network for problem 3-5 is provided in figure 1.

4. RESULTS

For each problem we generate the Interaction Network, the reduced Interaction Network using our algorithm described here, as well as two other reductions described below. Next we average the values across all 11 problems and compare. We discuss the results presented in table 1 and the comparisons with the other methods of reduction. Experts provided eight total solutions independently, four of which differed in either the actions or direction in which the problem was solved. Our reduced graph contained three out of the four solutions provided by experts. The fourth solution, a forward disjunctive syllogism and Modus Tollens approach was not present in our reduced network or the original full-data network. A working backward version of this solution is in the reduced graph which was solved by a single student.

4.1 Comparison

We compare against two alternative methods of filtering or reduction to help confirm the quality of our chosen approach. Those methods were a shortest path approach and a frequency 1 removal approach, which are somewhat naive but provide for good comparison. The shortest path method of reduction, takes in the start state of the problem and a

Table 1: The average metric scores across 11 problems and 2239 problem sessions. Original refers to the full network values. Each column is a method and its score, with percentile comparison to the Original network in parenthesis. For vertices and edges the percent is the amount of reduction, for goals and interactions it is the amount of coverage or inclusion.

	Original	Reduced	Shortest Paths	Greater than Frequency One
Vertices	1172	**114 (90.26%)**	238 (77.85%)	203 (81.73%)
Edges	1690	**132 (92.23%)**	237 (84.75%)	348 (78.33%)
Goals	38	20 (52.54%)	**38 (100.00%)**	12 (33.04%)
Interactions	3332	1283 (39.90 %)	1162 (36.89)	**1990 (60.77%)**
Avg. Edge Freq.	2.10	**12.34**	6.60	6.03

set of goal nodes for the problem. Next, Dijkstra's shortest path algorithm[4] is run and the result is the union of the shortest paths to each goal. The Frequency one filter approach, simply removes all edges from the network with student frequency one.

Referring to table 1 we can see some advantages and disadvantages of each approach. First, as expected, the shortest path approach naturally has 100% goal coverage, that is we can see *a* path to every goal from the original Interaction Network. The disadvantages of this approach is that the paths chosen do not optimize the frequencies of edges, because the shortest path can contain many frequency one edges. Next the overall reduction rates are half as effective as our method, leaving on average twice as many nodes and edges. This method preserves fewer actions performed by students while having lower rates of reduction. The resulting shortest paths network for problem 3-5 is shown in figure 2. Note, if the growth parameter is set to infinity, the path to all goals will be preserved - same as the shortest path method, though naturally reduction rates will be affected. Thus, our method can facilitate 100% goal coverage.

Alternatively, the frequency one filter, maintains a higher rate of interactions, as we would expect since fewer edges are removed. However, frequency one filtering suffers from low rates of reduction, having double the number of nodes and triple the number of edges on average, while also having lower rates of goal coverage, 33% compared to our method which achieved 52%. This method has lower reduction rates and lower goal coverage. Figure 3 shows the resulting network for problem 3-5 using the frequency one filtering process. By comparing the average edge frequencies in table 1 we can see our method has double the value than either of the other two approaches. This score is meaningful because it is the average number of actions performed by students, per edge within the network.

5. CONCLUSIONS

We provide an algorithm for reducing the complete Interaction Network to a summary of the most common problem solving approaches used by students. We showed that this algorithm was capable of reducing the number of vertices and edges of the Interaction Network by an average of around 90%, while still depicting more than half the of the solution paths and accounting for 40% of interactions performed by students.

6. REFERENCES

[1] G. Cobo, D. García-Solórzano, J. A. Morán, E. Santamaría, C. Monzo, and J. Melenchón. Using agglomerative hierarchical clustering to model learner participation profiles in online discussion forums. In *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*, LAK '12, pages 248–251, New York, NY, USA, 2012. ACM.

[2] B. Craft and P. Cairns. Beyond guidelines: What can we learn from the visual information seeking mantra? In *Proceedings of the Ninth International Conference on Information Visualisation*, pages 110–118, Washington, DC, USA, 2005. IEEE Computer Society.

[3] M. J. Croy. Graphic interface design and deductive proof construction. *J. Comput. Math. Sci. Teach.*, 18:371–385, December 1999.

[4] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[5] M. Eagle, M. Johnson, and T. Barnes. Interaction networks: Generating high level hints based on network community clusterings. In *EDM*, pages 164–167, 2012.

[6] D. Feng, J. Kim, E. Shaw, and E. Hovy. Towards modeling threaded discussions using induced ontology knowledge. *In Proceedings of National Conference on Artificial Intelligence (AAAI-2006)*, 2006.

[7] N.-T. Le and W. Menzel. Using constraint-based modelling to describe the solution space of ill-defined problems in logic programming. In *Proceedings of the 6th international conference on Advances in web based learning*, pages 367–379, Berlin, Heidelberg, 2008.

[8] A. Mitrovic. An intelligent sql tutor on the web. *Int. J. Artif. Intell. Ed.*, 13(2-4):173–197, Apr. 2003.

[9] S. Ritter, T. K. Harris, T. Nixon, D. Dickison, R. C. Murray, and B. Towle. Reducing the knowledge tracing space. In *EDM*, pages 151–160, 2009.

[10] L. A. Sudol, K. Rivers, and T. K. Harris. Calculating probabilistic distance to solution in a complex problem solving domain. In K. Yacef, O. R. ZaÃ̃ŕane, A. Hershkovitz, M. Yudelson, and J. C. Stamper, editors, *EDM*, pages 144–147. www.educationaldatamining.org, 2012.

[11] D. D. Suthers, H. U. Hoppe, M. de Laat, and S. B. Shum. Connecting levels and methods of analysis in networked learning communities. In *Proceedings of the 2nd International Conference on Learning Analytics and Knowledge*, LAK '12, pages 11–13, New York, NY, USA, 2012. ACM.

Mining Temporally-Interesting Learning Behavior Patterns

John S. Kinnebrew
Department of EECS and ISIS
Vanderbilt University
1025 16th Ave S, Ste 102
Nashville, TN 37212
john.s.kinnebrew@vanderbilt.edu

Daniel L.C. Mack
Department of EECS and ISIS
Vanderbilt University
1025 16th Ave S, Ste 102
Nashville, TN 37212
dmack@isis.vanderbilt.edu

Gautam Biswas
Department of EECS and ISIS
Vanderbilt University
1025 16th Ave S, Ste 102
Nashville, TN 37212
gautam.biswas@vanderbilt.edu

ABSTRACT
Identifying sequential patterns in learning activity data can be useful for discovering, understanding, and ultimately scaffolding student learning behaviors in computer-based learning environments. Algorithms for mining sequential patterns generally associate some measure of pattern frequency in the data with the relative importance or ranking of the pattern. However, another important aspect of these patterns is the evolution of their usage over the course of a student's learning or problem-solving activities. In order to identify and analyze learning behavior patterns of more interest in terms of both their overall frequency and their evolution over time, we present a data mining technique that combines sequence mining with a novel information-theoretic, temporal-interestingness measure and a corresponding heat map visualization. We demonstrate the utility of this technique through application to student activity data from a recent experiment with the Betty's Brain learning environment and a comparison of our algorithm's pattern rankings with those of an expert. The results support the effectiveness of our approach and suggest further refinements for identification of important behavior patterns in sequential learning activity data.

Keywords
sequence mining, interestingness measure, information gain, learning behaviors

1. INTRODUCTION
Identifying sequential patterns in learning activity data can be useful for discovering, understanding, and ultimately scaffolding student learning behaviors. The primary sequential pattern mining task, as applied in a variety of domains including education, is to discover sequential patterns of items that are found in many of the sequences in a dataset [1]. Some researchers have employed sequential pattern mining to inform student models for customizing learning to individual students (e.g., [2]). Other researchers have employed sequential pattern mining to better understand learning behavior in particular conditions or groups (e.g., [6, 8]).

However, once these behavior patterns are mined, researchers must interpret and analyze the often large set of patterns to identify a relevant subset of important patterns to investigate or utilize further. Ideally, these patterns provide a basis for generating models and actionable insights about how students learn, solve problems, and interact with the environment. Algorithms for mining sequential patterns generally associate some measure of pattern frequency to rank identified patterns. However, researchers have developed a variety of other measures to utilize properties beyond pattern frequency in ranking mined patterns [4]. These measures are often referred to as "interestingness measures" and have been applied to results from a variety of data mining techniques. To better analyze student learning and behavior, interestingness measures have been used tasks like ranking mined association rules (e.g., [7]).

Investigation of the frequency with which a pattern occurs over time can reveal additional information for pattern interpretation. Further these changes in pattern occurrence may help identify more important patterns, which occur only at certain times or become more/less frequent, rather than patterns with frequent, but uniform, occurrence over time. Qualitatively, we would like to identify patterns that are not rare overall and have significant variations in their frequency over time. In this paper, we present a novel approach, combining sequence mining and an information-theoretic measure for ranking behavior patterns that combines temporal variation in occurrence and overall frequency to provide more effective identification of temporally-interesting patterns. To effectively analyze these patterns and quickly identify trends in the evolution of pattern usage, we employ a related visualization in the form of heat maps. We demonstrate the utility of this technique through application to student activity data from a recent experiment with the Betty's Brain learning environment and a comparison of our algorithm's pattern rankings with those of an expert. The results support the effectiveness of our approach and suggest further refinements for identification of important behavior patterns in sequential learning activity data.

2. IDENTIFYING TEMPORALLY-INTERESTING PATTERNS
With long sequences of temporal data, such as student learning activities in a computer-based learning environment, re-

searchers and analysts are not only interested in discovering frequent sequential patterns, but, in many cases, also need to analyze their occurrence over time. In this paper, our focus is on studying how students' learning behaviors and strategies are employed with differing frequency over the course of learning or problem-solving activities (e.g., as the result of scaffolds and feedback provided by the learning environment or changing demands of the task over the course of learning). In this section, we present the Temporal Interestingness of Patterns in Sequences (TIPS) technique, and corresponding interestingness measure, for identifying and visualizing the most *temporally-interesting* patterns of student behavior.

The first step in analyzing learning activity sequences is to define and extract the actions that make up those sequences from interaction traces logged by the environment. The definition of actions in these sequences for Betty's Brain data is discussed further in Section 3. Given a set of sequences corresponding to the series of actions performed by each student, the TIPS technique consists of four primary steps:

1) Generate candidate patterns that are common to the majority of students by applying sequential pattern mining to students' learning activity sequences (with a frequency threshold of 50%).

2) Calculate a temporal footprint for each candidate pattern by mapping it back to locations where it occurs in the activity sequences. Specifically, each sequence is divided into n consecutive slices, such that each contains $\frac{100}{n}$% of the student's actions in the full sequence, where n is the chosen number of bins defining the temporal granularity of the comparisons. Corresponding slices (e.g., the first slice from each sequence, the second slice from each, and so on) are then grouped into bins and each action in the slices is marked to indicate whether or not it is the beginning of a pattern match in its original sequence. This set of binned and marked actions defines the temporal footprint of the pattern.

3) Provide a ranking of the candidate patterns using an information-theoretic interestingness measure (described in more detail below) applied to the temporal footprint of each pattern.

4) For the highly-ranked patterns, visualize their temporal footprints using heat maps to more easily assess usage trends and spikes. Specifically, we employ a single-dimensional heat map where each temporal bin's value is its percentage of the total pattern occurrence. The heat map is generated by assigning a color to each bin, which is determined by where its value falls between the highest and lowest value in the heat map.

In order to identify the more temporally-interesting patterns, the TIPS interestingness measure (in step 3) applies information gain with respect to pattern occurrence across the n bins of the temporal footprint. Information gain (IG) is defined as the difference in expected information entropy between one state and another state where some additional information is known (e.g., a set of data points considered as a homogeneous group versus one split into multiple groups based on the value of some other feature or attribute). IG is leveraged in classifiers to determine which features are most discriminatory because they provide the least amount of uncertainty among classes in the data. TIPS applies information gain to determine which patterns are the best descriptors of the data because knowledge of their occurrence provides the least amount of uncertainty about the temporal location of actions in the sequences. In TIPS, IG is applied to the temporal footprint of a pattern by using the n bins defined in the first step as the classes for the data points, where a data point is a single action in one of the students' sequences. The feature, in this case, corresponds to whether the action is the start of an occurrence of a given pattern.

For example, when analyzing students' activity sequences to extract their learning behavior patterns, we may divide up their activity sequence into 5 bins, with each bin containing 20% of the student's actions in the learning environment. The TIPS measure is then applied for a pattern by determining the information gain between the baseline where approximately the same number of actions are found in any particular bin and the case where we know whether each action corresponded to the occurrence of a pattern, which may happen more or less often in different bins. This IG measure for a pattern defines its temporal-interestingness in TIPS and is used to rank all candidate patterns in descending order, so the pattern that has the highest information gain will be ranked first.

The application of information gain to define the TIPS measure provides two important properties: 1) given two patterns with the same total occurrence, the pattern with the *greater temporal specificity* (i.e., the one that more uniquely distinguishes actions among the periods of time defined by the bins) will have the higher rank, and 2) given two patterns with the same proportions of their total occurrence in corresponding temporal bins, the pattern with the *greater total frequency* will have the higher rank. In this manner, the TIPS measure allows a trade-off between pattern frequency and temporal specificity. Therefore, TIPS tends to emphasize patterns with interesting temporal evolution (e.g., spikes of usage during specific time periods, as well as strongly increasing, decreasing, or peaking trends) even when they are not especially frequent, while also emphasizing particularly frequent patterns with more moderate changes in occurrence over time. Conversely, TIPS tends to deemphasize patterns that are homogeneous over the length of the sequence or that occur rarely.

3. BETTY'S BRAIN DATA

The data employed for the analysis in Section 4 consists of student interaction traces from the Betty's Brain [3]. learning environment. In Betty's Brain, students read about a science process and teach a virtual agent about it by building a causal map. They are supported in this process by a mentor agent, who provides feedback and support for their learning activities. The data analyzed here was obtained in a recent study with 68 7^{th}-grade students taught by the same teacher in a middle Tennessee school. At the beginning of the study, students were introduced to the science topic (global climate change) during regular classroom instruction, provided an overview of causal relations/maps, and given hands-on training with the system. For the next four 60-minute class periods, students taught their agent about climate change and received feedback on both domain con-

Table 1: Selected Patterns with TIPS and Occurrence Rankings

Pattern	TIPS Rank	Occurrence Rank	Avg Occurrence
[Read] → [Add link$^+$]	5	12	7.4
[Remove link$^-$] → [Quiz]	22	38	3.8
[Quiz] → [3+ Explanations]	23	78	2.0

tent and learning strategies from the mentor agent.

In Betty's Brain, the students' learning and teaching tasks are organized around seven activities: (1) reading resource pages to gain information, (2) adding or removing causal links in the map to organize and teach causal information to Betty, (3) querying Betty to determine her understanding of the domain based on the causal map, (4) having Betty take quizzes that are generated and graded by the mentor to assess her current understanding and the correctness of links in the map, (5) asking Betty for explanations of which links she used to answer questions on the quiz or in queries, (6) taking notes for later reference, and (7) annotating links to keep track of their correctness determined by quizzes and reading. Actions were further distinguished by context details, which for this analysis were the correctness of a link being edited and whether an action involved the same subtopic of the domain as at least one of the previous two actions. The definition of actions in Betty's Brain learning activity sequences are discussed further in [5].

4. RESULTS

To illustrate and characterize the performance of the TIPS technique, we present selected results of its application to learning activity sequences from the Betty's Brain classroom study described in Section 3. From the 68 students' activity sequences, sequential pattern mining identified 215 activity patterns that occurred in at least half of the students. For a broad, initial analysis of their usage evolution over time, we chose to bin pattern occurrence values into fifths of the activity sequences.

Table 1 presents 3 of the top 30 ranked patterns identified by TIPS with their average occurrences per student and a comparison of their rank between TIPS and the baseline ranking by frequency of occurrence. Overall, nearly half of the analyzed TIPS patterns (13 of the top 30) had a rank past 50th by occurrence, with most of those (9 of the top 30) ranking beyond 100th. Such low-ranking (by occurrence) patterns would be unlikely to have been noticed without the TIPS analysis.

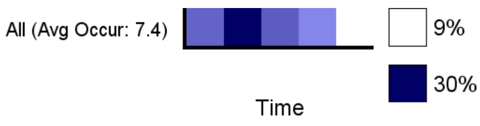


Figure 1: [Read] → [Add link$^+$]

Figure 1 illustrates the frequency over time for the pattern of reading followed by adding a correct link. This pattern was highly ranked both by occurrence (because it had a high average occurrence) and by TIPS (because it also had strong temporal variation). Students tended to perform this pattern earlier in their learning activities with a peak between 20% and 40% of their complete sequence of activity. An initial estimation of students' behavior by researchers assumed that students read and added correct links most in the first fifth of their activities with a decreasing trend as the remaining causal relationships were those that were harder to identify. Rather, the identified usage pattern suggests that students require most of an hour working with the system and reading before reaching peak efficiency in determining correct causal links from the resources.

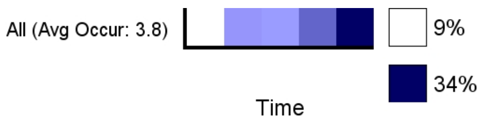


Figure 2: [Remove link$^-$] → [Quiz]

Another interesting pattern identified by TIPS, which students tended to perform late rather than early, was the removal of an incorrect link followed by taking a quiz. This pattern, illustrated in Figure 2, suggests a monitoring activity in which students employ the quiz to check whether the link was incorrect and should be removed. Although a generally increasing trend was expected since students add more incorrect links over time, the pattern's occurrence was more heavily weighted toward the end than expected. Over a third of the occurrences were in the last fifth of student activities, suggesting that most students either did not feel the need to monitor their evolving causal map until surprisingly late or took longer than expected to effectively identify potentially incorrect links and/or understand how to use the quiz to verify their removal.

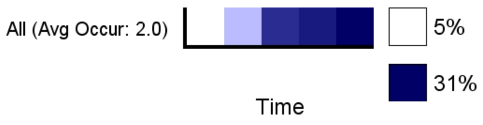


Figure 3: [Quiz] → [3+ Explanations]

Figure 3 illustrates another monitoring pattern in which students take a quiz and then ask for explanations of multiple (three or more) quiz questions in a row. This pattern also tends to occur later in the students' work on the system, but without as clear a peak at the end compared to the previous monitoring pattern. Further, this monitoring pattern had an occurrence rank of 78, making it especially unlikely to have been investigated without TIPS. Although the pattern only occurs twice per student on average, it does suggest that many students attempted to understand and analyze their quiz results in depth as part of their monitoring during the latter half of their work on the system.

To further assess the effectiveness of the TIPS analysis, we performed a preliminary comparison of the TIPS rankings

Table 2: Expert Interest Ranking

Measure	High	Medium	Low
TIPS	10	6	4
Occurrence	2	8	10

to those made by an expert - another researcher who coordinated the Betty's Brain study analyzed here, and who has analyzed student activity data but had no knowledge of the TIPS approach. For this comparison, we used the top 20 patterns from each of the TIPS and occurrence rankings, which resulted in 31 total patterns (9 patterns were in top 20 of both rankings). We presented the expert with only the total occurrence information and the occurrence over time (split into fifths corresponding to how the data was analyzed with TIPS). The order of the patterns was randomized and values for both the overall occurrence and the occurrence over time were represented on separate color-coded scales (between low and high values across all included patterns) to provide some visualization for comparison among the patterns. The expert was asked to group the patterns into three relative categories based on the provided information: high interest (10 patterns), medium interest (10 patterns), and low interest (11 patterns).

Table 2 presents the number of patterns identified by the TIPS and occurrence rankings that the expert grouped into each level of interest. All 10 of the expert's high interest results were in the top 20 identified by TIPS, with only 2 of them also in the top 20 ranked by occurrence. These results suggest that the TIPS ranking is closer to the expert's own interest ranking, given the total occurrence and temporal evolution information about each. Next, we presented the expert with the same information but also included the specific activity pattern for each result. When asked to rank the patterns again with this additional information, the results were more equally balanced between the TIPS and occurrence rankings, with six of each in the high interest category, and TIPS having two more than the occurrence ranking in the medium interest category. Overall, these preliminary experiments illustrate the expected point that the activity pattern itself is a major factor in its overall interestingness, but its occurrence and temporal evolution are both important factors. Further, it suggests that rather than relying on only one interestingness measure for identifying potentially important activity patterns, consideration of the top patterns identified by each of multiple measures, including both occurrence and TIPS, may be the most effective way to analyze mined patterns from learning activity sequences.

5. CONCLUSION

While identification of common and high-occurrence patterns is undoubtedly useful, finding patterns that have interesting evolution of usage over time is also important for researchers and experts in education, as well as other domains. In this paper, we presented the TIPS technique and interestingness measure for identifying temporally-interesting behavior patterns in learning activity sequences. TIPS is designed to identify patterns with interesting temporal behavior (e.g., spikes of usage during specific time periods or strongly increasing, decreasing, and peaking trends) even when they are not especially frequent, as well as particularly frequent patterns that have at least some clear changes in occurrence over time.

Results from the use of this technique to mine Betty's Brain data illustrated the potential benefits of identifying behaviors with an interesting evolution over time and helped characterize differences between TIPS and a baseline occurrence ranking. Although general trends in occurrence may be expected for some patterns through consideration of the constraints imposed by the system and the learning activities, TIPS concretely identifies the patterns with strong temporal evolution, confirming some expectations but also identifying patterns with temporal trends that differ from expectations or that would not even have been considered without the TIPS analysis. Further, results from an expert ranking of patterns provided preliminary evidence that patterns identified by TIPS are of particular interest. Overall, the results illustrated the utility of the TIPS technique and suggested that combining the top patterns identified by TIPS and occurrence ranking may be the most useful approach for initial analysis and identification of important learning behavior patterns. Future work will include automatic identification of an effective number of bins for splitting a given set of activity sequences in TIPS and application of identified patterns to improve dynamic scaffolding of learning.

6. REFERENCES

[1] R. Agrawal and R. Srikant. Mining sequential patterns. In *Proceedings of the Eleventh IEEE International Conference on Data Engineering (ICDE)*, pages 3–14, 1995.

[2] S. Amershi and C. Conati. Combining unsupervised and supervised classification to build user models for exploratory learning environments. *Journal of Educational Data Mining*, 1(1):18–71, 2009.

[3] G. Biswas, K. Leelawong, D. Schwartz, N. Vye, and T. Vanderbilt. Learning by teaching: A new agent paradigm for educational software. *Applied Artificial Intelligence*, 19(3):363–392, 2005.

[4] L. Geng and H. J. Hamilton. Interestingness measures for data mining: A survey. *ACM Computing Surveys (CSUR)*, 38(3):9, 2006.

[5] 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*, In Press, 2013.

[6] R. Martinez, K. Yacef, J. Kay, A. Al-Qaraghuli, and A. Kharrufa. Analysing frequent sequential patterns of collaborative learning activity around an interactive tabletop. In *Proceedings of the Fourth International Conference on Educational Data Mining*, Eindhoven, Netherlands, 2011.

[7] A. Merceron and K. Yacef. Interestingness measures for association rules in educational data. *Educational Data Mining 2008*, page 57, 2008.

[8] J. Nesbit, M. Zhou, Y. Xu, and P. Winne. Advancing log analysis of student interactions with cognitive tools. *12th Biennial Conference of the European Association for Research on Learning and Insruction (EARLI)*, 2007.

Modeling Students' Learning and Variability of Performance in Problem Solving

Petr Jarušek
Masaryk University Brno
xjarusek@fi.muni.cz

Matěj Klusáček
Masaryk University Brno
matej.klusacek@mail.muni.cz

Radek Pelánek
Masaryk University Brno
pelanek@fi.muni.cz

ABSTRACT
Given data about problem solving times, how much can we automatically learn about students' and problems' characteristics? To address this question we extend a previously proposed model of problem solving times to include variability of students' performance and students' learning during sequence of problem solving tasks. We evaluate proposed models over simulated data and data from a "Problem Solving Tutor". The results show that although the models do not lead to substantially improved predictions, the learnt parameter values are meaningful and capture useful information about students and problems.

1. INTRODUCTION
In intelligent tutoring systems [1, 15] student models typically focus on correctness of student answers [6], and correspondingly the problems in tutoring systems are designed mainly with the focus on correctness. This focus is partly due to historical and technical reasons – the easiest way to collect and evaluate student responses are multiple choice questions. Thanks to the advances in technology, however, it is now relatively easy to create rich interactive problem solving activities. In such environments it is useful to analyze not only correctness of students answers, but also timing information about a solution process.

To attain a clear focus, here we consider only the information about problem solving times, i.e., we model students performance in exercises where the only performance criterium is time to solve a problem. Examples of such exercises are logic puzzles (like the well-known Sudoku puzzle) or suitably formulated programming and mathematics problems [9].

In previous work [8] we described a model which assumes a linear relationship between a problem solving skill and a logarithm of time to solve a problem, i.e., exponential relation between skill and time. In this work we present extensions of the model in two directions. The first extension models variability of performance of individual students. The second extension models students' learning (improvement of problem solving skill) during a sequence of tasks.

The feasibility of detecting students' learning and variability of performance from problem solving data depends on many factors: how much data are available, what is the noise in the problem solving times, what is the magnitude of learning effects, whether students solve problems in the same order. We use simulated data to identify conditions under which detection of students' parameters may be possible.

We also evaluate our models on real data from the Problem Solving Tutor [7, 9]. With the available data set the extended models do not bring substantial improvement of predictions of future problem solving times. Nevertheless, the experiments show that the fitted parameter values are reasonably robust and bring useful information about students' learning and variability of their performance.

2. MODELING PROBLEM SOLVING TIMES
We recapitulate the model of problem solving times which was introduced in [8] and then describe two extensions of the model and parameter estimation for these models.

2.1 Related Work
The presented models extend our previous work on modeling problem solving times [8]. The modeling approach is related to several areas. It is an example of a learner modeling in intelligent tutoring systems (see [6] for an overview) and the model is closely related to models used in item response theory [2]. Both of these areas focus mainly on modeling correctness of answers (probability that a student will answer a test item correctly), we focus on modeling timing (probability distribution of time to solve a problem). Another related area are recommender systems [10], which are used mainly in e-commerce to recommend users products that may be interesting for them. Particularly relevant technique is collaborative filtering [11], which takes matrix of ratings of products by users, and predicts future ratings. Our model is analogical to collaborative filtering when we reinterpret users as students, products as problems, and ratings as performance (for other similar work see [3, 14]). With respect to modeling learning, there is an extensive research on learning curves (e.g., [13, 12]). In the context of intelligent tutoring systems the most often used approach is Bayesian knowledge tracing [5], which models probability that a student has learned a particular skill during a sequence of attempts.

2.2 The Basic Model

We assume that we have a set of students S, a set of problems P, and data about problem solving times: t_{sp} is a logarithm of time it took student $s \in S$ to solve a problem $p \in P$. In the following we always work with a logarithm of time and use subscript s to index student parameters and subscript p to index problem parameters.

We assume that a problem solving performance depends on one latent problem solving skill θ_s and two main problem parameters: a basic difficulty of the problem b_p and a discrimination factor a_p. The basic structure of the model is simple – a linear model with Gaussian noise: $t_{sp} = b_p + a_p\theta_s + \epsilon$. Basic difficulty b describes expected solving time of a student with average skill. Discrimination factor a describes the slope of the function, i.e., it specifies how the problem distinguishes between students with different skills. Finally, ϵ is a random noise given by a normal distribution with zero mean and a constant variance (note that the model in [8] assumes problem dependent variance). The presented model is not yet identified as it suffers from the "indeterminacy of the scale" (analogically to many IRT models). This is solved by normalization – we require that the mean of all θ_s is 0 and the mean of all a_p is -1.

2.3 Modeling Variability of Students' Performance

The basic model outlined above assumes constant variance of the noise. But some students are more consistent in their performance than others and also problem characteristics influence the variance of the noise. To incorporate these factors we propose to model the variance as $c_p^2 + \sigma_s^2 a_p^2$ – a weighted sum of a problem variance c_p^2 and a student variance σ_s^2, where student's contribution to the variance depends on the discrimination of the problem. Intuitively, student's characteristics matter particularly for more discriminating problems.

Thus now we have three problem parameters a, b, c and two student parameters θ, σ. The model is the same as before, only the noise is modeled in more detail:

$$t_{sp} = b_p + a_p\theta_s + \mathcal{N}(0, c_p^2 + a_p^2\sigma_s^2)$$

The model with variance $c_p^2 + a_p^2\sigma_s^2$ is equivalent to the following approach: "at first determine a student's local skill for the attempt (based on his variance σ^2) and then determine the problem solving time with respect to this local skill":

$$p(\theta'|s) = \mathcal{N}(\theta'|\theta_s, \sigma_s^2) \qquad p(t|\theta', p) = \mathcal{N}(t|b_p + a_p\theta', c_p^2)$$

The equivalence of these two definitions is a special case of a general result for Gaussian distributions (see e.g., [4]).

2.4 Modeling Learning

It is sensible to incorporate learning into the model. The basic model assumes a fixed problem solving skill, but students problem solving skill should improve as they solve more problems – that is, after all, aim of tutoring systems. The model extension is inspired by the extensive research on learning curves (e.g., [13, 12]). A learning curve is a graph which plots the performance on a task (usually time or error rate) versus the number of trials. The shape of the learning curve is in the majority of human activities driven by power law: $T = BN^{-\alpha}$ (where T is the predicted time, N the number of trials, α the learning rate and B the performance at the first trial).

If we take the logarithm of the above mentioned form of the power law, it can be naturally combined with our basic model of problem solving times:

$$t_{sp} = b_p + a_p(\theta_s + \delta_s \cdot \log(k_{sp})) + \epsilon$$

where δ_s is a student's learning rate and k_{sp} is the order of the problem p in problem solving sequence of a student s. In the current analysis of this model we assume constant variance. Nevertheless, the model can be easily combined with the more detailed model of the noise presented above.

2.5 Parameter Estimation

We need to estimate model parameters from given data. To do so we use maximum likelihood estimation and stochastic gradient descent. As this is rather standard approach (see e.g., [4]) we focus in the following description only on the derivation of the error function and the gradient.

We derive the maximum likelihood estimation for the model with detailed noise (including student and problem variance). The likelihood of observed times t_{sp} for this model is:

$$L = \prod_{s,p} \mathcal{N}(t_{sp}|b_p + a_p\theta_s, c_p^2 + a_p^2\sigma_s^2)$$

To make the derivation more readable, we introduce the following notation: $e_{sp} = t_{sp} - (b_p + a_p\theta_s)$ (prediction error for a student s and a problem p), $v_{sp} = c_p^2 + a_p^2\sigma_s^2$ (variance for a student s and a problem p). Thus we can write the log-likelihood as:

$$\ln L = \sum_{s,p} -\frac{e_{sp}^2}{2v_{sp}} - \frac{1}{2}\ln(v_{sp}) - \frac{1}{2}\ln(2\pi)$$

Maximizing the log-likelihood is equivalent to minimizing the following error function:

$$E = \textstyle\sum_{s,p} E_{sp} \quad \text{where } E_{sp} = \frac{1}{2}(\frac{e_{sp}^2}{v_{sp}} + \ln(v_{sp}))$$

It is intractable to find the minimum analytically, but we can minimize the function using stochastic gradient descent. To do so we need to compute a gradient of E_{sp}:

$$\begin{aligned}
\frac{\partial E_{sp}}{\partial a_p} &= \frac{-e_{sp}\theta_s v_{sp} - a_p\sigma_s^2 e_{sp}^2}{v_{sp}^2} + \frac{a_p\sigma_s^2}{v_{sp}} \\
&= -\frac{e_{sp}}{v_{sp}}(\theta_s + a_p\sigma_s^2\frac{e_{sp}}{v_{sp}}) + \frac{a_p\sigma_s^2}{v_{sp}} \\
\frac{\partial E_{sp}}{\partial b_p} &= -\frac{e_{sp}}{v_{sp}} \\
\frac{\partial E_{sp}}{\partial \theta_s} &= -a_p\frac{e_{sp}}{v_{sp}} \\
\frac{\partial E_{sp}}{\partial c_p^2} &= \frac{1}{2}(-\frac{e_{sp}^2}{v_{sp}^2} + \frac{1}{v_{sp}}) = -\frac{1}{2v_{sp}^2}(e_{sp}^2 - v_{sp}) \\
\frac{\partial E_{sp}}{\partial \sigma_s^2} &= \frac{1}{2}(-a^2\frac{e_{sp}^2}{v_{sp}^2} + a^2\frac{1}{v_{sp}}) = -\frac{a^2}{2v_{sp}^2}(e_{sp}^2 - v_{sp})
\end{aligned}$$

Note that the obtained expressions have in most cases straightforward intuitive interpretation. For example the gradient with respect to θ_s is $-a_p\frac{e_{sp}}{v_{sp}}$, which means that the estimation procedure gives more weight to attempts over problems which are more discriminating and have smaller variance.

Stochastic gradient descent can find only local minima. However, by good initialization we can improve the chance of

finding a global optimum. In our case there is a straightforward way to get a good initial estimate of parameters:
b_p = mean of t_{sp} (for the given p);
a_p = -1;
θ_s = mean of $b_p - t_{sp}$ (for the given s);
$c_p = \frac{1}{2}$ of variance of $b_p - t_{sp}$ (for the given p);
$\sigma_s = \frac{1}{2}$ of variance of $b_p - t_{sp}$ (for the given s).

If we return to the original simplifying assumption and assume that the variance is constant (independent of a particular problem and student), then the error function is the basic sum-of-squares error function and the computation of gradient simplifies to $\frac{\partial E_{sp}}{\partial a_p} = -\theta_s e_{sp}$, $\frac{\partial E_{sp}}{\partial b_p} = -e_{sp}$, $\frac{\partial E_{sp}}{\partial \theta_s} = -a_p e_{sp}$. Parameter estimation for the model with learning is analogical.

3. EVALUATION

Now we report on evaluation of the models over simulated data and real data from the Problem Solving Tutor.

3.1 Simulated Data

The models described in Section 2 can be easily used to generate simulated data. Even though we have large scale data about real students, the simulated data are still useful, because for these data we know "correct answers" and thus we can thoroughly evaluate the parameter estimation procedure. The results show that the basic difficulty of problems b and students' skill θ can be estimated easily even from relatively few data. Estimating problem discrimination a is more difficult – to get a good estimate we need data about at least 30 solvers and even with more data the further improvement of the estimates is slow. As could be expected, it is most difficult to get a reasonable estimate of student and problem variance. To do so we need data about at least 50 problems and 150 students.

For the model with learning, the possibility of detection of learning depends on the situation. If the differences in learning rates are high and noise is low, then it is easy to detect the learning in the data. If the students' learning rates are very similar and noise in data is high, it is impossible to detect the learning. The feasibility of detection of learning also depends on the order in which students solve problems. In many practical cases the ordering in which students solve problems is very similar. Often students proceed from simpler problems to more difficult ones (this is certainly true for our data which are used below). If the ordering of problems for individual students is highly correlated, there is no way to distinguish between the absolute values of student learning and intrinsic difficulty of problems. On the other hand, the ordering of problems does not impact the estimation of relative learning rates (i.e., comparing students' learning rates).

3.2 Predictions

Next we report on the evaluation of predictions of problem solving times for data on real students using a Problem Solving Tutor [7, 9] – a free web-based tutoring system for practicing problem solving (available at `tutor.fi.muni.cz`). The system has more than 10 000 registered students (mainly university and high school students), who have spent more then 13 000 hours solving more than 400 000 problems. The system contains 30 types of problems, particularly computer science problems (e.g., binary numbers, robot programming, turtle graphics, introductory C and Python programming, finite automata), math problems (e.g., functions and graphs, matching expressions), and logic puzzles (e.g., Sokoban, Nurikabe, Slitherlink). For the experiment we used 8 most solved problem types from the Problem Solving Tutor, for each problem we consider only students who solved at least 15 instances of this problem.

We compare model predictions with two simpler ways to predict problem solving times. At first, we consider the mean time as a predictor – the simplest reasonable way to predict solving times (note that, consistently with the rest of the work, we compute the mean over the logarithm of time and thus the influence of outliers is limited and the mean is nearly the same as the median).

At second, we consider a simple 'personalized' predictor $\hat{t}_{sp} = m_p - \delta_s$, where m_p is the mean time for a problem p and δ_s is a "mean performance of student s with respect to other solvers", i.e., $\delta_s = (\sum m_p - t_{sp})/n_s$, where n_s is the number of problems solved by the student. Note that this corresponds to the initialization of our basic model (Section 2.5); we call it a baseline predictor.

Evaluation of model predictions was done by repeated random subsample cross-validation, with 10 repetitions. The training and testing sets are constructed in the following way: we choose randomly 30% of students and put the data of the last 20% of their attempts to the testing set; the remaining data forms the training set. Table 1. compares the results using the root mean square error metric.

The results show that the model provides improvement over the use of a mean time as a predictor. Most of the improvement in prediction is captured by the baseline model; the basic model brings a slight but consistent improvement. This improvement is larger for educational problems (e.g., Binary numbers) than for logic puzzles (e.g., Tilt maze).

Different variants (basic model with constant variance, individual variance, learning) of the model lead to similar predictions and similar values of RMSE. The model with individual variance leads in same cases to improved RMSE, the model with learning leads to slightly worse results than the basic model. One possible reason can be the higher number of parameters which causes slight overfitting. The second reason may be difference in scale between the parameters – values of the learning coefficient are typically between 0 and 0.2 while other parameters have wider spread. Thus it should be possible to improve the results of gradient descent using different step sizes for each parameter, particularly smaller step size for the learning parameter δ. Experiments with the improved algorithm really show statistically significantly better in results in some cases (e.g., in the case of Slitherlink, which is a puzzle with many opportunities for improving performance).

3.3 Analysis of Parameter Values

Even through the more complex models do not lead to substantially improved predictions, they can still bear interesting information. Predictions are useful for guiding behaviour

Table 1: Quality of predictions for different models and problems measured by root mean square error metric.

	Tilt	Robot.	Binary	Region	Slith.	Sokoban	Rush.	Nurik.
Mean time predictor	1.045	1.376	1.259	1.37	1.195	1.246	1.077	1.143
Baseline predictor	0.925	1.324	1.174	1.28	0.976	1.037	0.995	1.026
Basic model	0.92	1.301	1.148	1.28	0.948	1.021	0.981	1.025
Model with variance	0.918	1.304	1.161	1.278	0.947	1.016	0.978	1.025
Model with learning	0.948	1.313	1.181	1.322	0.967	1.034	0.993	1.04

Table 2: Spearman's correlation coefficient for parameter values obtained from two independent halves of the data.

	Tilt	Robot.	Binary	Region	Slith.	Sokoban	Rush.	Nurik.
student skill θ	0.748	0.641	0.822	0.472	0.816	0.789	0.737	0.904
student learning rate δ	0.525	0.394	0.623	0.576	0.455	0.394	0.509	0.570
basic problem difficulty b	0.994	0.961	0.951	0.927	0.981	0.963	0.962	0.837
problem discrimination a	0.469	0.564	0.569	0.282	0.533	0.347	0.434	0.195

of the tutoring systems, but small improvement in prediction precision will not change the behaviour of the system in significant way. The important aim of the more complex models is to give us additional information about students and problems, e.g., the student's learning rate, which can be used for guiding the behaviour of tutoring system and for providing feedback to users.

Since the model with learning does not improve predictions, it may be, however, that the additional parameters overfit the data and thus do not contain any valuable information. To test this hypothesis we performed the following experiment: we split the data into two disjoint halves, we use each half to train one model, and then we compare the parameter values in these two independent models. Specifically, we measure the Spearman correlation coefficient for values of each parameter.

Table 2 shows results for the model with learning. The results show, that estimates of basic difficulty and basic skill correlate highly, the weakest correlation between the estimates from the two halves is for the discrimination parameter. For students' learning rate, the additional parameter of the extended model, we get the correlation coefficient between 0.5 and 0.7 – a significant correlation which signals, that the fitted parameters contain meaningful values.

We also analyzed correlations among different model parameters, e.g., between skill θ and learning rate δ. Generally there is only weak correlation between parameters, which shows that the new parameters bring additional information.

4. REFERENCES

[1] J. Anderson, C. Boyle, and B. Reiser. Intelligent tutoring systems. *Science*, 228(4698):456–462, 1985.

[2] F. Baker. *The basics of item response theory.* University of Wisconsin, 2001.

[3] Y. Bergner, S. Droschler, G. Kortemeyer, S. Rayyan, D. Seaton, and D. Pritchard. Model-based collaborative filtering analysis of student response data: Machine-learning item response theory. In *Educational Data Mining*, pages 95–102, 2012.

[4] C. Bishop. *Pattern recognition and machine learning.* Springer, 2006.

[5] A. Corbett and J. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.

[6] M. C. Desmarais and R. S. J. de Baker. A review of recent advances in learner and skill modeling in intelligent learning environments. *User Model. User-Adapt. Interact.*, 22(1-2):9–38, 2012.

[7] P. Jarušek and R. Pelánek. Problem response theory and its application for tutoring. In *Educational Data Mining*, pages 374–375, 2011.

[8] P. Jarušek and R. Pelánek. Analysis of a simple model of problem solving times. In *Proc. of Intelligent Tutoring Systems (ITS)*, volume 7315 of *LNCS*, pages 379–388. Springer, 2012.

[9] P. Jarušek and R. Pelánek. A web-based problem solving tool for introductory computer science. In *Proc. of Innovation and technology in computer science education*, pages 371–371. ACM, 2012.

[10] P. Kantor, F. Ricci, L. Rokach, and B. Shapira. *Recommender systems handbook.* Springer, 2010.

[11] Y. Koren and R. Bell. Advances in collaborative filtering. *Recommender Systems Handbook*, pages 145–186, 2011.

[12] B. Martin, A. Mitrovic, K. R. Koedinger, and S. Mathan. Evaluating and improving adaptive educational systems with learning curves. *User Modeling and User-Adapted Interaction*, 21(3):249–283, 2011.

[13] A. Newell and P. Rosenbloom. Mechanisms of skill acquisition and the law of practice. *Cognitive skills and their acquisition*, pages 1–55, 1981.

[14] N. Thai-Nghe, T. Horváth, and L. Schmidt-Thieme. Factorization models for forecasting student performance. In *Proc. of Educational Data Mining*, pages 11–20, 2011.

[15] K. Vanlehn. The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education*, 16(3):227–265, 2006.

Estimating Student Knowledge from Paired Interaction Data

Anna N. Rafferty
Computer Science Division
University of California
Berkeley, CA 94720
rafferty@cs.berkeley.edu

Jodi Davenport
WestEd
Oakland, CA 94612
jdavenp@wested.org

Emma Brunskill
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
ebrun@cs.cmu.edu

ABSTRACT

Estimating students' knowledge based on their interactions with computer-based tutors has the potential to improve learning by decreasing time taking assessments and facilitating personalized interventions. Although there exist good student models for relatively structured topics and tutors, less progress has been made with more open-ended activities. Further, students often complete activities in pairs rather than individually, with no coding to indicate who performed each action. We investigate whether pair interactions with an open-ended chemistry tutor can be used to predict individual student post test performance. Using L_1-regularized regression, we show that student interactions with the tutor are predictive both of the average post-test score for the pair and of individual scores. Towards better understanding pair dynamics in this setting, we also find that for pairs composed of students with similar pre-test scores, we can predict the difference in students' post-test scores.

Keywords

Collaboration, embedded assessment, supervised learning

1. INTRODUCTION

Computer-based educational activities have many advantages over traditional tests as a means to assess student knowledge. The function of testing is to provide information about student proficiency. If an analysis of how a student completes an activity can provide similar information, time-intensive post-tests can be eliminated, and students can have access to the immediate feedback known to support learning. Projects such as ASSISTments [13] and stealth assessment [15] have demonstrated the potential for this approach.

Both interactive activities specifically designed for assessment and traditional intelligent tutoring systems provide valuable information about student knowledge. Simulation-based activities that are designed to be assessments have proven effective for measuring science inquiry and reasoning skills (e.g., [5, 12]). Many tutoring systems use student modeling to estimate proficiency as a student works through problems in the tutor. However, estimating students' knowledge based on their work in games and more open-ended environments introduces new challenges [3]. These environments are less structured, lack explicit tags about which tasks correspond to which skills, and may offer few opportunities to practice the same skill repeatedly in a similar context. Despite these challenges, games or more open-ended environments are important as they enable different forms of learning and can be used where formal testing is impractical.

A further challenge in estimating student knowledge from computer-based activities is that in classroom environments, students often work with computers in groups. Though collaboration can improve students' learning from computer-based science activities, automatically logged data rarely captures explicit collaboration, such as which student provided any given input, or what conversations occurred in conjunction with the activity.

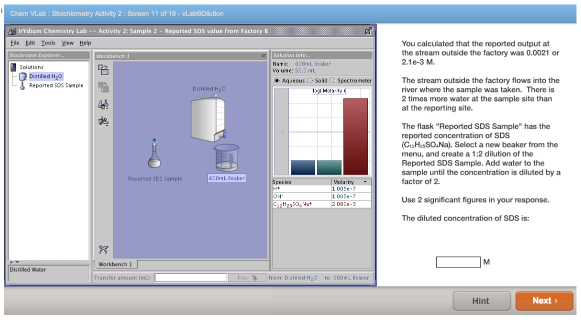
In this paper, we explore whether machine learning based approaches can predict student knowledge based on interactions with an open-ended chemistry tutor, ChemVLab+. Due to limitations in the number of available computers in many classrooms, students generally use ChemVLab+ in pairs, and we analyze only data from paired interactions. We investigate what predictions we can make about individual student knowledge, corroborated by a separate post-test, based on the students' interactions with ChemVLab+.

2. BACKGROUND

We briefly review the literature on open-ended environments and collaboration in computer-based educational activities.

2.1 Open-ended tutoring environments

Many computer-based educational environments have open-ended components in which students explore topics using free-form actions. One approach to understanding student learning is to identify behaviors that are correlated with high or low learning gains. For instance, the WISE platform has identified patterns of inquiry behavior that are common in more successful students [10]. Kinnebrew, Loretz, and Biswas [8] identified patterns of student actions associated with periods of productivity and analyzed which patterns were correlated with high learning gains. In contrast


Chem VLab : Stoichiometry Activity 2 : Screen 11 of 19 - vLabBDilution
IrYdium Chemistry Lab -- Activity 2: Sample 2 - Reported SDS value from Factory B
File Edit Tools View Help
Stockroom Explorer...
Solutions
Distilled H_2O
Reported SDS Sample
Workbench 1
Distilled H_2O
Reported SDS Sample
600mL Beaker
Solution Info...
Aqueous Solid Spectrometer
log(Molarity)
-5
Distilled Water
Workbench 1
Transfer amount (mL):
Pour
from Distilled H_2O to 600mL Beaker
M
Hint
Next

Name: 600mL Beaker
Volume: 50.0 mL

Species	Molarity
H^+	1.005e-7
OH^-	1.005e-7
$C_{12}H_{25}SO_4Na^+$	2.080e-3

You calculated that the reported output at the stream outside the factory was 0.0021 or 2.1e-3 M.

The stream outside the factory flows into the river where the sample was taken. There is 2 times more water at the sample site than at the reporting site.

The flask "Reported SDS Sample" has the reported concentration of SDS ($C_{12}H_{25}SO_4Na$). Select a new beaker from the menu, and create a 1:2 dilution of the Reported SDS Sample. Add water to the sample until the concentration is diluted by a factor of 2.

Use 2 significant figures in your response.

The diluted concentration of SDS is:

Figure 1: A screenshot of a virtual lab activity.

to identifying correlations between strategies and post-test performance, we focus on predicting student post-test scores given tutor interaction features.

Another area of work in open-ended learning environments focuses on recognizing strategies and goals. For example, Ha et al. [6] used Markov logic networks to identify student goals based on game behaviors. In the chemistry domain, progress has also been made on identifying semantic actions, such as a titration, based on individual behaviors in a virtual lab, such as pouring a small amount from one beaker to another [1, 4]. Strategy recognition provides descriptive information about student behaviors, but doesn't relate strategies to learning gains. Incorporate rich strategy features to predicting student understanding and post-test performance is an interesting future direction.

2.2 Data mining of student collaborations

Collaborative work is common in educational activities, and research provides evidence that collaborative work improves student learning [14, 21, 22]. Much work on computer-supported collaboration focuses on modeling the collaborative process and on how collaborative activities are related to learning and knowledge [18, 9]. Within this work, there is often explicit record of collaborative activities in the form of audio recordings, observation logs, or community interactions such as comments on a discussion forum [7, 16, 17]. Explicit records enable a system to learn classifiers to characterize individual student interactions and identify the roles of individual students in a collaboration [11, 19]. In contrast, our data lacks such explicit records of collaboration, a typical situation in the classroom due to practical difficulties and teacher preferences. Thus, we believe that exploring data from paired interactions in ChemVLab+ may be relevant to other educational tools as well.

3. CHEMVLAB+

ChemVlab+ is a collection of online activities that allow students to apply their chemistry knowledge in authentic, real-world contexts [2]. Each activity involves a separate problem, such as whether factories are reporting accurate pollution levels, and consists of a series of pages. ChemVLab+ activities include both freeform actions, such as virtual labs (see Figure 1), and more constrained actions, such as multiple choice questions.[1] The virtual labs enable similar actions as in a real chemistry lab: students manipulate beakers and use chemical instruments. These virtual labs are a key part of ChemVLab+ as they allow students to plan and execute experiments to investigate the real-world problem.

[1]Activities can be found at `http://www.chemvlab.org`.

The data we analyze were collected from three schools during the 2011-2012 school year. Students first completed a paper and pencil pre-test. They then completed four ChemVLab+ stoichiometry activities using computers in the classroom; activities were completed in the same order by all students and over at least four class periods. Shortly after the final use of the ChemVLab+ activities, the students completed a post-test, which was identical to the pre-test. The test contained multiple choice and numerical free response items; the topics covered were similar to those in ChemVLab+, and there were a total of 30 points on the test. In the data, all 266 students completed the activities in assigned groups of two students, but completed pre- and post-test individually.

4. PREDICTING POST-TEST SCORES

We now explore what we can learn about students' knowledge based on their interactions with ChemVLab+, beginning with prediction of post-test scores. Paired performance data cannot necessarily tell us about individuals: intraclass correlation shows that for our data, the two post-test scores in the pair are not significantly correlated ($r = 0.12$, $p = .08$, *n.s.*). However, we will shortly see we still can predict some interesting aspects of individual and pair performance.

4.1 Methods

For all analyses, we use methods based on lasso (L_1-norm) regularization [20]. Lasso regularization is a popular machine learning method that adds a penalty term $\lambda||\beta||_1$ to the objective in traditional linear regression approaches, where β is the vector of predictor coefficients and λ is a scaling factor. This term favors solutions where many features have weight zero, even if this results in some increase in error; larger values of λ favor using fewer predictors. Thus, feature selection is performed as part of the regression algorithm.

We computed features for each pair of students based on their behavior in the four ChemVLab+ activities. Twelve features were used for each activity, including four features based on help seeking and submission behavior on each page, four features for activity in the virtual labs, and four features capturing holistic behavior in the activity (e.g., total time on task). In some cases, a pair did not complete any pages in an activity, generally due to being absent from school. In these cases, we set the value of the feature for number of pages completed in the activity to zero. For all other features in that activity, we use feature imputation and set their values to the average value of that feature for other students.[2]

As students use ChemVLab+ in pairs but take the post-test separately, we predict three possible quantities using the interaction data: the higher of the two post-test scores, the lower of the scores, and the average of the scores. Note that if we use only pair interaction data, we can predict the higher of the two scores, but not which student will get which score. For each analysis, we want to maximize the proportion of the data that can be used for training while

[2]We also tried imputation using data only from pairs who were similar to the current pair on the other activities; this did not significantly affect predictive performance.

Prediction task	MAD by features for regression ChemVLab+	 Pretest
Avg. post-test score	2.6	2.5
Higher post-test score	3.0	2.9
Lower post-test score	3.2	3.3

Table 1: Regression error for post-test predictions.

minimizing overfitting. Given the relatively small dataset of 133 pairs of students, we use linear regression, which does not include interactions among features. By excluding interactions, we limit some of the risk of overfitting due to chance relationships among features. We fit the regression using 10-fold cross validation and limit the maximum number of features that can have non-zero coefficients to 20.

Accuracy is measured as the mean absolute deviation (MAD) for predictions for all pairs: MAD $= \frac{1}{n}\sum_{i=1}^{n}|\hat{Y} - Y|$, where $\hat{Y}$ is the predicted post-test score, Y is the true post-test score, and $n = 133$ is the number of pairs. Lower MAD values indicate more accurate performance. We compare the performance of regression using the features based on tutor-student interactions versus using only pre-test features. The pre-test is highly correlated with the post-test score ($r(265) = 0.67$, $p < .001$), so we would expect pre-test scores to be relatively accurate predictors of post-test scores. To predict the average post-test score using pre-test features, we have a feature for the higher pre-test score in the pair and the lower score. For predicting individual post-test scores using pre-test features, we use the student's pre-test score.

4.2 Results

As shown in Table 1, the tutor-student interaction features achieve comparable predictive performance to using the pre-test features to predict student performance, and both provide quite accurate estimates. This suggests that even without prior information about the students, interaction data alone can provide useful indicators of student knowledge, despite the additional challenge that all interaction data comes from paired performance. Both sets of features are slightly better at predicting the average post-test score for the pair, which has somewhat less variance, and both are slightly worse at predicting the lower score. The decrease in accuracy for the pre-test features on the latter target is likely because the lower score has a smaller correlation with the pre-test score than the higher score ($r(132) = 0.48$ versus $r(132) = 0.71$; for both, $p < .001$). For all analyses, we also examined using both interaction and pretest features, but this did not significantly improve performance, suggesting that the two types of features capture similar information.

Lasso regression favors sparse solutions: the regression models used between 9 and 14 features, with the model for predicting the lower post-test score having the fewest features and the model for predicting the higher score having the most. All models included features from each activity as well as virtual lab features. Overall, these results demonstrate that the interaction data are relatively accurate predictors of post-test scores, despite the variety of tasks and the lack of a model of learning in the tutor. We also explored predicting learning gains based on the interaction data, but had less success, probably due to the choice of features. Our features captured behavior averaged across pages, but did not take into account changes in behavior from page to page.

5. TOWARDS RECOGNIZING HOW PAIRINGS AFFECT LEARNING

The previous section demonstrates the potential for using the ChemVLab+ activities as embedded assessments. We now explore what we can learn about pairs as a unit by predicting the difference between the two post-test scores in the pair. When restricted to pairs with similar pretest scores, large differences in post-test scores may signal a lack of collaboration, which could be used to drive interventions. Predicting differences in post-test scores may also reveal interaction features related to collaboration.

5.1 Methods

Lasso regression is again used for prediction and feature selection, with the same 48 features as in the previous analysis. 10-fold cross validation is used to fit the model, and the regression is limited to 20 features with non-zero weights. In analyses with pretest features, these features are the highest pretest in the pair, the lowest pretest in the pair, and the difference between the two pre-test scores.

5.2 Results

We first predicted differences in post-test scores for all pairs. The average difference in post-test scores was 6.0 points, with a standard deviation of 4.8 points. As shown in Table 2, prediction is relatively poor, and including both tutor interaction features and pre-test features did not increase performance. Due to concerns about overfitting, we limited the regression to linear features, which means the weight of each tutor feature is the same regardless of pretest-score. However, we might expect that these weights should be dependent on the pre-test scores. For instance, in a pair with dissimilar pre-test scores, high rates of hint reading might be indicative of a lack of collaboration. In pairs with similar pre-test scores, rates of hint reading might be less predictive because both students are likely to benefit from the hints.

To address this issue, we restricted the regression to the 43 pairs who had pre-test scores that were within two points of one another. The average difference in post-test score for these pairs was 4.9 points (*SD*= 4.1), and only about one-third of the pairs have post-test scores that are within two points of one another. Regressing on pairs with similar pre-test scores results in substantially lower prediction error than when all pairs are included (Table 2). Prediction is much more accurate than the standard deviation, and the interaction features result in more accurate predictions than the pre-test scores. For the analysis using the interaction features, twelve of these features had non-zero coefficients, including six features based on behavior in the virtual lab.

The previous analysis showed that we can predict differences in post-test score for pairs with similar initial knowledge. However, it does not tell us how initial knowledge and collaboration interact. Just as features and weights for predicting differences in post-test scores may differ for pairs based on the similarity of their pre-test scores, the regression may differ for pairs with different levels of initial knowledge. To explore this issue, we performed two additional analyses: predicting post-test scores for only those pairs where both students had below-average pre-test scores (*low* pairs) and predicting post-test scores for only those pairs where both

	MAD by features for regression	
Pairs included	ChemVLab+	Pretest
All	3.9	3.7
Similar pre-test	2.0	3.4
High pre-test	3.0	3.2
Low pre-test	2.7	3.6

Table 2: Regression error for predicting differences between the post-test scores for students in the pair.

students had above-average pre-test scores (*high* pairs). The average pre-test score was 12.5 points out of 30.

The 35 *high* pairs had an average post-test score difference of 5.5 points (*SD*= 3.8). As shown in Table 2, this difference can be predicted relatively accurately. The most notable thing about this analysis, though, is that only two features are given non-zero weights. The small number of features suggests that when students have high initial knowledge, few features are indicative of the quality of collaboration.

In contrast, eight features have non-zero weight when predicting differences in post-test for the 43 *low* pairs. These pairs had an average post-test score difference 4.6 points (*SD*= 4.3), and the interaction features are more accurate predictors than the pretest features (Table 2). The features with non-zero weight included three lab features and at least one feature from each activity. One feature associated with smaller differences in post-test scores, due to having a relatively large negative weight, was the average number of submissions per page in Activity 2. This activity was difficult for students, and a lower number of submissions may have indicated that students were combining their knowledge, which is likely to result in more similar post-test scores.

6. CONCLUSIONS

Given differences in classroom implementations and the pedagogical benefits of more open-ended tutors, there are many advantages to predicting student performance based on real-world use of these systems. In this paper, we examined data from a series of chemistry activities that students completed in pairs, and found that pairs' interactions with the activities were predictive of individual post-test scores. Though we could make some predictions about differences in post-test scores for a pair, there is likely to be a limit on how well we can perform this task given the lack of data about individuals within the pair. We plan to explore how limited data about individual behavior, collected via classroom observation, can be used to create more accurate models of collaboration, and whether explicitly modeling control of the computer as a latent variable can improve performance. We would also like to explore a broader feature set, including features that capture changes in performance over time and more fine-grained virtual lab features (e.g., from pattern-mining [8]). We see this work as a first step in showing the potential of data mining techniques to transform collaborative educational activities into embedded assessments, even when activities are not designed for this purpose.

Acknowledgements. This research was supported an NDSEG Fellowship to ANR and by the Institute of Education Sciences, U.S. Department of Education, through Grant R305A100069 to WestEd. The opinions expressed are those of the authors and do not represent views of the Institute or the U.S. Department of Education.

7. REFERENCES

[1] O. Amir and Y. Gal. Plan recognition in virtual laboratories. In *IJCAI*, 2011.

[2] J. L. Davenport, A. N. Rafferty, M. J. Timms, D. Yaron, and M. Karabinos. ChemVLab+: Evaluating a virtual lab tutor for high school chemistry. In *ICLS*, 2012.

[3] S. De Freitas and M. Oliver. How can exploratory learning with games and simulations within the curriculum be most effectively evaluated? *Comput Educ*, 46(3):249–264, 2006.

[4] Y. Gal, E. Yamangil, S. M. Shieber, A. Rubin, and B. J. Grosz. Towards collaborative intelligent tutors: Automated recognition of users' strategies. In *ITS*, 2008.

[5] J. Gobert, M. Sao Pedro, R. S. Baker, E. Toto, and O. Montalvo. Leveraging educational data mining for real time performance assessment of scientific inquiry skills within microworlds. *J Educ Data Mining*, 4:153–185, 2012.

[6] E. Y. Ha, J. P. Rowe, B. W. Mott, and J. C. Lester. Goal recognition with Markov logic networks for player-adaptive games. In *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 2011.

[7] M. Kapur. Temporality matters: Advancing a method for analyzing problem-solving processes in a computer-supported collaborative environment. *IJCSCL*, 6:39–56, 2011.

[8] J. Kinnebrew, K. Loretz, and G. Biswas. A contextualized, differential sequence mining method to derive students' learning behavior patterns. *J Educ Data Mining*, in press.

[9] Y. Li, J. Wang, J. Liao, D. Zhao, and R. Huang. Assessing collaborative process in CSCL with an intelligent content analysis toolkit. In *ICALT*, 2007.

[10] K. W. McElhaney and M. C. Linn. Impacts of students' experimentation using a dynamic visualization on their understanding of motion. In *ICLS*, 2008.

[11] 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, 2009.

[12] E. S. Quellmalz, J. L. Davenport, M. J. Timms, G. DeBoer, K. Jordan, K. Huang, and B. Buckley. Next-generation environments for assessing and promoting complex science learning. *J Educ Psychol*, in press.

[13] L. Razzaq, M. Feng, G. Nuzzo-Jones, N. T. Heffernan, K. R. Koedinger, B. Junker, et al. The Assistment project: Blending assessment and assisting. In *ITS*, 2005.

[14] B. B. Schwarz, Y. Neuman, and S. Biezuner. Two wrongs may make a right... if they argue together! *Cognition Instruct*, 18(4):461–494, 2000.

[15] V. Shute. Stealth assessment in computer-based games to support learning. *Computer Games and Instruction*, 2011.

[16] A. Soller, J. Wiebe, and A. Lesgold. A machine learning approach to assessing knowledge sharing during collaborative learning activities. In *CSCL*, 2002.

[17] G. Stahl. Meaning and interpretation in collaboration. In *CSCL*, 2003.

[18] G. Stahl, T. Koschmann, and D. Suthers. Computer supported collaborative learning: An historical perspective. In R. K. Sawyer, editor, *Cambridge Handbook of the Learning Sciences*. Cambridge University Press, 2006.

[19] L. Talavera and E. Gaudioso. Mining student data to characterize similar behavior groups in unstructured collaboration spaces. In *AI in CSCL at ECAI*, 2004.

[20] R. Tibshirani. Regression shrinkage and selection via the lasso. *J Roy Stat Soc B Met*, pages 267–288, 1996.

[21] F. I. Winters and P. A. Alexander. Peer collaboration: The relation of regulatory behaviors to learning with hypermedia. *Instr Sci*, 30(4):407–427, 2011.

[22] F. I. Winters and R. Azevedo. High-school students' regulation of learning during computer-based science inquiry. *J Educ Comput Res*, 33(2):189–217, 2005.

Using a Lexical Analysis of Students' Self-Explanation to Predict Course Performance

Nicholas M. Rhodes
Dept. of Computer Science
University of California Riverside
Riverside, CA 92521, USA
nrhod001@ucr.edu

Matthew A. Ung
Dept. of Mechanical Engineering
University of California Riverside
Riverside, CA 92521, USA
mung001@ucr.edu

Alexander E. Zundel
Dept. of Mechanical Engineering
University of California Riverside
Riverside, CA 92521, USA
azund001@ucr.edu

James Herold
Dept. of Computer Science
jhero001@ucr.edu

Thomas F. Stahovich
Dept. of Mechanical Engineering
stahov@engr.ucr.edu

ABSTRACT

Numerous studies have shown that self-explanation can lead to improved learning outcomes. Here we examine how the words which students use in their self-explanations correlate with their performance in the course as well as with the effort they expend on their homework assignments. We compute two types of numerical features to characterize students' work: vocabulary-based features and effort-based features. The vocabulary-based features capture the frequency with which individual words and n-grams appear within students' self-explanation. The effort-based features estimate the effort expended on each assignment as the amount of time spent writing a homework solution or self-explanation response.

We use the most predictive vocabulary-based and effort-based features to train a linear regression model to predict students' overall course grade. This model explains up to 19.4% of the variance in students' performance. Furthermore, the underlying parameters of this model provide valuable insights into the ways students explain their own work, and the cognitive processes students employ when asked to self-explain. Additionally, we use the vocabulary-based features to train linear regression models to predict each of the effort-based features. In doing so we demonstrate that the vocabulary employed by a student to self-explain his or her solution to an assignment correlates with the amount of effort that student expends on that particular assignment. Both of these findings serve as a basis for a novel automated assessment technique for evaluating student performance.

1. INTRODUCTION

Self-explanation is the process by which a student explains his or her solution process, summarizing his or her understanding. Prior work has demonstrated that self-explanation can improve a student's metacognitive skills, leading to improved learning gains. These studies have typically focused on summative assessments of students' learning, demonstrating, for example, that students who were asked to provide self-explanation of their homework solutions performed better on exams than students who did not provide self-explanation. In this paper, we present a novel technique which provides a formative analysis of self-explanation, identifying behaviors which correlate with good performance. In particular we employ machine learning techniques to identify successful patterns latent in students' self-explanations.

This analysis is enabled by our unique dataset of students' handwritten coursework. We conducted a study in which students in an undergraduate Mechanical Engineering Statics course generated handwritten self-explanations of the major steps they followed when solving each of their homework problems. The students completed the homework and self-explanations using Livescribe^{TM}Smartpens. These devices produce a digital record of students' handwritten work in the form of time-stamped pen strokes, enabling us to see not only the final ink on the page, but also the order in which it was written.

We compute numerical features from this digital record which characterize the vocabulary used and the effort (time) expended, both in solving problems and writing self-explanation. Using these features we have computed a statistical model which predicts students' grades on various homework assignments. This model accounts for up to 19.4% of the variance in the students' performance. Furthermore, the underlying parameters of this model provide valuable insights into the ways students explain their own work, and the cognitive processes students employ when asked to self-explain.

Additionally, we use the vocabulary-based features to train linear regression models to predict each of the effort-based features. In doing so we demonstrate that the vocabulary employed by a student to self-explain his or her solution correlates with the amount of effort that student expends on that particular assignment.

2. RELATION TO PRIOR WORK

Chi et al. [4] have argued that "the metacognitive component of training is important in that it allows students to understand and take control of their learning process." Metacognition is the awareness of one's own learning process, and it serves as a major foundation for research performed on self-explanation. We use self explanation as a tool to improve students' metacognition.

Numerous studies have demonstrated the positive impact self-explanation has on student performance. Bielaczyc et al. [2] studied the impact of different self-explanation strategies on a student's ability to learn LISP programming. The experiment revealed a significant difference between the learning gains from the pre- to posttest performance of students who did and did not generate self-explanation. In this study students self-explained after viewing study materials but before solving problems. This differs from our study in which students generate self-explanation throughout their solution process.

Chi et al. [4] made comparisons between two groups of students: "poor" and "good" performing students. These students were asked to generate self-explanation after studying worked-out example problems. The results of this study demonstrated that students who perform poorly are typically unable to generate sufficient self-explanation of the worked-out example problems. This study indicates that a correlation may exist between the quality of students' self-explanation and their performance.

Hall and Vance [8] investigated the impact of self-explanation on student performance as well as self-efficacy in a Statistics course. This study showed that students who generated collaborative self-explanation performed significantly better at solving problems than students who did not self-explain. What these studies have in common is their use of summative performance assessments to show the positive impact of self-explanation on learning gains. To our knowledge, little prior work has focused on formative assessments which identify behaviors in students' self-explanations.

Prevost et al. [11] examined typed self explanations from an online system. Prevost et al. compared multiple choice responses versus constructed (free form) responses and found that constructed responses provided better insight into student thinking than multiple choice responses. Although mentioned, the authors did not examine the sequencing between individual words. Our paper focuses on analyzing sequences of words to predict student performance without manually scoring student self-explanations. While past research has typically examined data extracted from close-structured responses(e.g., multiple choice or check boxes), our paper examines free-form, handwritten responses in order to predict course performance. Our analysis is similar to that of Forbes-Riley et al. [5] in which the authors modeled students' spoken interactions with a tutoring system.

3. EXPERIMENTAL DESIGN

In the winter quarter of 2012, we conducted a study in which students enrolled in an undergraduate Mechanical Engineering Statics course were given Livescribe$^{\text{TM}}$Smartpens. These devices serve the same purpose as traditional pens, allowing students to handwrite their homework on paper. Additionally, these devices record a digital copy of the handwritten work as time-stamped pen strokes.

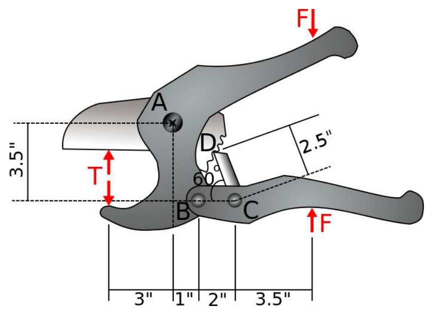


Figure 1: "The device shown is used for cutting PVC pipe. If a force, F = 15 lb, is applied to each handle as shown, determine the cutting force T. Also, determine the magnitude and the direction of the force that the pivot at A applies to the blade."

Thirty of the students in the course were asked to provide self-explanation on five of the homework assignments. A typical homework problem is shown in Figure 1. These problems required students to solve for unknown forces that result when external forces are applied to a system in equilibrium. Students were provided with three to five prompts eliciting explanations for each of their major solution steps. An example of a typical self-explanation prompt is, "Why did you select the system that you used for your free-body diagram?" Students handwrote their responses to these questions and submitted them along with their solutions.

4. DATA PROCESSING

We manually transcribed each handwritten self-explanation, producing 111 text documents. Each document contains all self-explanation written by a single student for a single homework assignment. During this manual transcription we made slight modifications to the students' explanations to make them suitable for later processing. First, we corrected any spelling mistakes, but did not correct grammatical errors. Second, we replaced each verb with its unconjugated form. For example, we replaced "pushed" (past tense) with "push" (infinitive). Our later analysis counts the number of occurrences of words based on exact spelling. These changes ensure that spelling variations do not prevent words from being correctly identified.

We also developed a thesaurus to replace synonymous words with a single, canonical word. Students use a variety of words to refer to a given concept or object. For example, when students described a free-body diagram, they often used the terms "system" and "body" interchangeably. To ensure that semantically identical words were identified as such, we manually developed a thesaurus that maps a canonical concept to each of the words that may be used to express that concept. For example, we created a "free-body diagram element" concept category that comprises every word that students used to refer to any component (body) in a free-body diagram, such as "jaw" or "handle". In this example, whenever the word "jaw" was found in a transcript, it was replaced with the token "FBD-Element". We developed a total of ten conceptual categories with the help of a Statics domain expert. There were approximately 1640 unique words used by students across all documents before correcting spelling or verb tense. After applying our thesaurus-based replacement, there were 750 unique words.

5. VOCABULARY, EFFORT, AND PERFORMANCE FEATURES

In this section we describe the numerical features which characterize: the vocabulary employed by a student in his or her self-explanation; the effort expended by a student on his or her solution and self-explanation; and each student's performance in the course.

We use the "term frequency - inverse document frequency" score (TF-IDF)[9] to characterize the importance of each word in a transcribed self-explanation document. The TF-IDF scores of all words encountered in all documents are used as features.

To characterize the sequence of words in students' self-explanation, we analyze the frequencies of the bigrams and trigrams which appear in each self-explanation document. In our analysis, we split word choices on periods and thus consider bigrams and trigrams within sentence boundaries. We use the N-Gram Statistics Package (NSP) [3, 1] to both identify and calculate the frequency of n-grams present in each document.

NSP provides a number of different methods for measuring the frequency of a given n-gram. We used the total mutual information (TMI) to score each gram. TMI scores an n-gram by computing the ratio of the log of the joint probability of all words in that n-gram over the marginal probability distributions of each word in an n-gram.

Additionally, we compute two features that characterize the effort spent on an assignment and the corresponding self-explanation. We used the average time spent drawing free-body diagrams, writing equations, and answering self explanation questions. This produced three separate effort based features.

6. FEATURE SUBSET SELECTION

Given that there are $\sim$ 750 unique words across all explanations, there would be over 19,000 TF-IDF, bigram, and trigram features computed for each of the 111 documents. This is too large a feature set and would lead to an over-fit model with inflated accuracy. To address this issue we use two feature subset selection algorithms to reduce the size of our feature set.

First, we apply the computationally inexpensive RELIEF [10] algorithm to prune our feature set to the top 500 features. The RELIEF algorithm scores each feature by its similarity to the nearest instance of the same class and to the nearest instance of each other class. Next, we apply the computationally expensive, but more rigorous Correlation Feature Selection (CFS) [7] algorithm to further reduce the feature subset. We use the RELIEF and CFS implementation available in the WEKA [6] machine learning software suite.

7. PREDICTING STUDENT PERFORMANCE AND EFFORT

We trained four separate linear regression models to predict students' course performance, equation effort, free-body diagram effort, and self-explanation effort respectively. Both of the aforementioned subset feature selection algorithms are executed separately for each model, resulting in four distinct feature subsets which range from 13 to 25 features. The effort-based features and vocabulary-based features are used as input to the feature subset selection for the performance-based model, and only the vocabulary-based features are used for the three effort-based models. Each of these four models was trained using the linear regression implementation available in WEKA [6].

Table 1 shows the coefficient weights for each of the features in the first regression model. The magnitude of each weight indicates the predictive power of that feature in determining either students' performance or effort. Similarly, the sign of the weight indicates whether or not that feature correlates positively or negatively with performance or effort. The performance, equation effort, free-body diagram, and self-explanation models are able to explain 19.4%, 17.8%, 20.0%, and 45.7% of the variance in their respective dataset. The final three models are computed, but not shown in this paper.

Table 1: Underlying parameters of the linear regression model used to predict students' overall course grade. Each row corresponds to a single feature which is the mutual information value of a single n-gram or the TF-IDF scores of single words. The attribute column presents the n-gram or word used to compute the feature and the weight column presents the weight of that feature in the linear regression model.

Weight	Attribute
-1174.8889	a<>force<>on
-395.8207	the<>twoforcemember
-358.6974	twoforcemember<>force
-300.7872	a<>force<>was
-292.9685	on<>an
-153.5296	action<>and<>i
-135.6537	the<>direction<>steeper
-135.6536	direction<>steeper<>angle
-135.6535	steeper
75.4336	solving<>solving<>of
87.9657	point<>i
92.4137	asked<>for
99.8558	interaction<>with
104.9789	knew
115.2228	body<>but
115.4557	interaction<>would<>be
125.317	and<>the<>boom
125.3176	we<>look<>at
125.3178	we<>are<>act
125.3181	act<>a<>force
125.3182	are<>act<>a
125.3186	act<>a
125.319	because<>we<>are
152.9983	to<>the
459.9756	body<>is<>a

8. DISCUSSION

The accuracy of our model for predicting student performance is encouraging. More interesting though, is the fact that the model and its parameters indicate the self-explanation behaviors that correlate with strong or weak performance. By manually investigating these behaviors, we are able to identify the metacognitive skills students demonstrate regarding their problem-solving processes.

Take, for example, the feature with greatest weight, the TMI score of the trigram "body<>is<>a". By manually inspecting the self-explanation responses that contain this trigram, we found that the trigram "body<>is<>a" is typically used by a student to identify a special type of body in a system. For example, one student's self-explanation response reads "The lever is a two-force system." In this example, the word "lever" belongs to the "body" thesaurus category. This provides strong evidence of a students' ability to both recognize and apply concepts learned in class to given homework problems. By identifying the "two-force system" the student is able to apply a particular technique from class which only applies to two-force systems.

Similarly, consider the difference between the trigram "a<>force<>was" and the bigram "point<>i". In examining the self-explanation responses, we found that responses which contained the trigram "a<>force<>was" were used passively in sentences, whereas the bigram "point<>i" was used actively in sentences. This provides evidence of the importance of active voice in self-explanations positively correlating with student performance while passive-voice sentences correlate negatively with performance.

Some attributes tended to reinforce our intuition regarding students' performance. The word "knew" indicates conceptual understanding and a student's confidence in their problem-solving. When we examined these self-explanation responses, the word "knew" expressed premeditation and certainty. For example, one such self-explanation transcript read, "I knew that by taking a moment about point A that I would cancel out forces at F."

Obvious grammatical errors tended to lead to poor performance. In our manual investigation, we found that the trigram "action<>and<>i" was primarily used in run-on sentences. Consideration of alternative solution paths correlated positively with performance. The bigram "body<>but" was typically used by a student to indicate that there was another way to solve a particular problem.

9. CONCLUSION

In this work, we have demonstrated a novel technique for analyzing students' handwritten self-explanations of their homework solutions. This technique is enabled by our unique dataset of student work. We conducted a study in which thirty students in an undergraduate Mechanical Engineering Statics course provided handwritten self-explanations of the major steps they followed when solving each of their homework problems. The students completed the homework and self-explanations using Livescribe^{TM}Smartpens. These devices produce a digital record of students' handwritten work in the form of time-stamped pen strokes, enabling us to see not only the final ink on the page, but also the order in which it was written.

We compute numerical features from this digital record which characterize the vocabulary used and effort expended in constructing handwritten self-explanations. We applied a heuristic subset selection algorithm to identify the optimal subset of features for predicting homework performance. Using this subset, we computed a linear regression model that predicts students' grades on homework assignments. This model accounts for 19.4% of the variance in the students' performance. While this is a strong correlation, what is more valuable are the insights that can be drawn from the underlying parameters of this model. The coefficient weights of the model may be used to guide manual analysis of the students' self-explanation responses, revealing patterns that provide insights into the types of self-explanation behaviors that are indicative of understanding or lack thereof.

10. REFERENCES

[1] S. Banerjee and T. Pedersen. The design, implementation, and use of the ngram statistics package. *Computational Linguistics and Intelligent Text Processing*, pages 370–381, 2003.

[2] K. Bielaczyc, P. L. Pirolli, and A. L. Brown. Training in self-explanation and self-regulation strategies: Investigating the effects of knowledge acquisition activities on problem solving. *Cognition and instruction*, 13(2):221–252, 1995.

[3] W. B. Cavnar and J. M. Trenkle. N-gram-based text categorization. In *In Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, 1994.

[4] M. T. Chi, M. Bassok, M. W. Lewis, P. Reimann, and R. Glaser. Self-explanations: How students study and use examples in learning to solve problems. *Cognitive science*, 13(2):145–182, 1989.

[5] K. Forbes-Riley, D. Litman, A. Purandare, M. Rotaru, and J. Tetreault. Comparing linguistic features for modeling learning in computer tutoring. In *Proceedings of the 2007 conference on Artificial Intelligence in Education: Building Technology Rich Learning Contexts That Work*, pages 270–277, Amsterdam, The Netherlands, The Netherlands, 2007. IOS Press.

[6] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

[7] M. A. Hall. *Correlation-based Feature Subset Selection for Machine Learning*. PhD thesis, University of Waikato, Hamilton, New Zealand, 1998.

[8] S. Hall and E. A. Vance. Improving self-efficacy in statistics: Role of self-explanation and feedback. *J. Stat. Educ*, 18(3), 2010.

[9] K. S. Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28:11–21, 1972.

[10] K. Kira and L. A. Rendell. A practical approach to feature selection. In D. H. Sleeman and P. Edwards, editors, *Ninth International Workshop on Machine Learning*, pages 249–256. Morgan Kaufmann, 1992.

[11] L. Prevost, K. Haudek, M. Urban-Lurain, and J. Merrill. Deciphering student ideas on thermodynamics using computerized lexical analysis of student writing. 2012.

A meta-learning approach for recommending a subset of white-box classification algorithms for Moodle datasets

C. Romero, J.L. Olmo, S. Ventura
Department of Computer Science
University of Cordoba, Spain
cromero@uco.es, jlolmo@uco.es, sventura@uco.es

ABSTRACT

This paper applies meta-learning to recommend the best subset of white-box classification algorithms when using educational datasets. A case study with 32 Moodle datasets was employed that considered not only traditional statistical features, but also complexity and domain specific features. Different classification performance measures and statistics tests were used to rank algorithms. Furthermore, a nearest neighbor approach was used to recommend the subset of algorithms for a new dataset. Our experiments show that the best recommendation results are obtained when all three types of dataset features are used.

Keywords

Meta-learning, classification, predicting student performance

1. INTRODUCTION

One of the oldest and best-known problems in educational data mining (EDM) [10] is predicting student's performance as a classification task. A wide range of algorithms have been applied to predict academic success and course results. However, selecting and identifying the most adequate algorithm for a new dataset is a difficult task, due to the fact that there is no single classifier that performs best on all datasets, as proven by the No Free Lunch (NFL) theorem [6]. Choosing appropriate classification algorithms for a given dataset is of great importance in practice. Meta-learning has been used successfully to address this problem [12]. Meta-learning is the study of the main methods that exploit meta-knowledge to obtain efficient models and solutions by adapting machine learning and the DM process [4]. Recommendation can be presented in various ways, such as the best algorithm in a set, a subset of algorithms, a ranking of algorithms, or the estimated performance of algorithms. We propose to use several classification evaluation measures and statistical tests to rank algorithms, and a nearest neighbor approach to recommend the subset of best algorithms for a given new dataset.

Meta-learning has been used mainly in general domain and publicly available datasets such as UCI [2]. However, we have not found any papers that tackle algorithm selection using meta-learning in the EDM domain. There is only one related work about using meta-learning to support the selection of parameter values in a J48 classifier using several educational datasets [8]. In the educational domain, the comprehensibility of discovered classification models is an important issue, since they should be interpretable by users who are not experts in data mining (such as instructors, course authors and other stakeholders) so they can be used in decision-making processes. Indeed, white-box DM models based on rules are preferred to black-box DM models such as Bayesian and artificial neural networks, although they are normally more accurate but less comprehensible [11]. On the other hand, statistics and information theory measures [3] and more recently data complexity measures [7] are widely used to characterize datasets in meta-learning. However, we propose to also use domain specific measures to characterize datasets.

The paper is organized as follows: Section 2 introduces the methodology used in this work; Section 3 describes the Moodle educational datasets employed in the experimentation; Section 4 describes the experiments, results, and the model obtained; and finally, conclusions and future works are outlined in Section 5.

2. METHODOLOGY

We propose a meta-learning methodology that consists of two steps (see Figure 1):

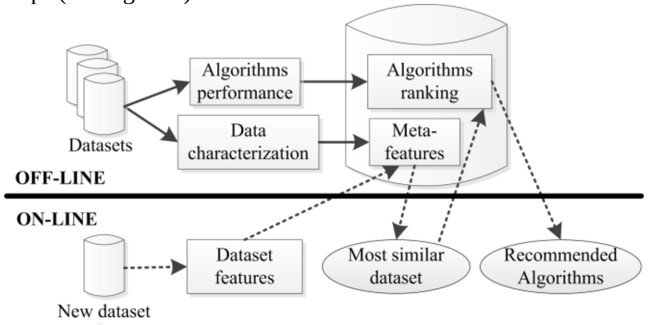


Figure 1. Meta-learning methodology.

- An off-line or training phase for creating the meta-database starting from educational datasets and classification algorithms. On the one hand, we identified important properties for characterizing datasets (statistics, complexity and domain) and developing meta-features. On the other hand, we used white-box classification algorithms (rule-based and decision tree algorithms) to evaluate their performance on all the available datasets. For each dataset, we used a statistical test on several classification evaluation measures to rank and select the subset of algorithms that gave the best performance, in such a way that there were no significant differences, as far as performance is concerned, between all the algorithms in the subset.
- An on-line or prediction phase to recommend a subset of classification algorithms to a new dataset using a nearest neighbor approach. Firstly, when a new dataset appears, its features are compared against all the meta-features in order to find the most similar dataset. Then, the subset of algorithms recommended for the new dataset corresponds to those previously obtained for its nearest neighbor.

3. DATASETS

We used a set of 32 classification educational datasets (see Table 1) about predicting students' final performance starting with Moodle's usage data [10]. As input attributes, these datasets contain a variety of information about the interaction of students in Moodle and the class to be predicted is the final mark (categorical) obtained by students in the courses. All this data was collected from university Computer Science students between 2007 and 2012. For each dataset, 16 features were obtained that can be grouped into the following three types:

- Statistical features (see Columns 2 to 6 in Table 1): the number of instances or students (Ni), the number of numerical attributes (Nna), the number of categorical attributes (Nca), the number of classes or labels of the mark attribute such as Pass/Fail, High/Medium/Low, etc. (Nc), and the imbalance ratio (IR), which is the ratio between instances of the majority class and minority class.
- Complexity features that characterize the apparent complexity of datasets for supervised learning [7], such as the maximum Fisher's discriminant ratio, the overlap of the per-class bounding boxes, the maximum (individual) feature efficiency, the collective feature efficiency (sum of each feature efficiency), the fraction of points on the class boundary, the ratio of average intra/inter class nearest neighbor distance, the leave-one-out error rate of the one-nearest neighbor classifier, the non-linearity of the one-nearest neighbor classifier, the fraction of maximum covering spheres, and the average number of points per dimension. We used DCoL (data complexity library) to obtain all the previous complexity measures [9] from our datasets.
- A domain feature (see the last column in Table 1) that indicates what the specific source of each dataset is, which can either be a Moodle's report, quiz or forum. Report is a general summary about the interaction of each student in Moodle, such as: total time in Moodle, number of accesses/sessions, number of resources viewed, number of assignments done, average score in assignments done, total time spent on assignments, number of activities carried out, total time spent on activities, etc. Quiz is a specific summary about the interaction of each student with quizzes, such as: total time spent on quizzes and each quiz done, number of quizzes answered, number of quizzes passed, average score in quizzes, correctly/incorrectly answered questions, knowledge in each concept evaluated by the quiz, etc. Forum is a specific summary about the interaction of each student with forums, such as: total time spent in forums and each forum, number of messages sent, number of messages read, number of threads created, number of replies received, number of words and sentences written, etc.

4. EXPERIMENTS

An initial experiment was carried out to select a subset of white-box classification algorithms that best predicted the final students' performance for each Moodle dataset. We used only rule-based and decision trees algorithms due to the fact that they provide models that can be easily understood by humans and used directly in the decision-making process.

Dataset	Ni	Nna	Nca	Nc	IR	Domain
Dataset1	98	4	0	2	1.08	Report
Dataset 2	194	0	4	2	1.39	Report
Dataset 3	786	6	0	3	9.8	Quiz
Dataset 4	658	0	6	3	9.1	Quiz
Dataset 5	67	40	0	2	1.23	Quiz
Dataset 6	922	6	0	3	19.27	Quiz
Dataset 7	910	0	6	3	19.24	Quiz
Dataset 8	114	0	11	2	1.19	Forum
Dataset 9	42	0	11	2	6	Forum
Dataset 10	103	0	11	2	1.53	Forum
Dataset 11	114	11	0	2	1.43	Forum
Dataset 12	98	0	6	2	1.91	Forum
Dataset 13	81	6	0	2	1.19	Forum
Dataset 14	33	0	12	2	32	Forum
Dataset 15	82	0	12	2	3.1	Forum
Dataset 16	113	40	0	4	23.5	Quiz
Dataset 17	105	41	0	3	1.06	Quiz
Dataset 18	123	0	10	4	3.89	Quiz
Dataset 19	102	10	0	3	1.06	Quiz
Dataset 20	75	0	8	2	2.12	Report
Dataset 21	52	0	4	2	1.89	Report
Dataset 22	208	10	0	2	3.25	Report
Dataset 23	438	0	10	4	15.41	Report
Dataset 24	421	10	0	4	14.2	Report
Dataset 25	84	6	0	4	5.43	Report
Dataset 26	168	6	0	4	11.25	Report
Dataset 27	136	6	0	4	11.5	Report
Dataset 28	283	0	10	2	1.67	Report
Dataset 29	155	0	10	2	1.21	Report
Dataset 30	72	6	0	4	11	Report
Dataset 31	40	0	10	2	1.2	Quiz
Dataset 32	48	10	0	2	1.8	Quiz

Table 1. Statistics and domain features of the datasets.

The next 19 classification algorithms provided by Weka 3.6 [13] were used:

- **Rule-based algorithms**: ConjunctiveRule, DecisionTable, DTNB, JRip, NNge, OneR, PART, Ridor and ZeroR.
- **Tree-based algorithms**: BFTree, DecisionStump, J48, LADTree, LMT, NBTree, RandomForest, RandomTree, REPTree and SimpleCart.

We executed each algorithm using all the Moodle datasets, which account for a total of 608 executions (19 algorithms * 32 datasets). All algorithms were executed using their default parameters and 10-fold cross-validation.

Several classification performance measures were used to compare algorithm performance [13], such as sensitivity (Sen), precision (Prec), F-Measure (F-M), Kappa (Kap) and the area under the ROC curve (AUC). For instance, Table 2 shows the average values for these measures obtained by each algorithm on dataset1.

Algorithm	Sen	Prec	F-M	Kap	AUC
RConjunctiveRule	0.845	0.869	0.846	0.694	0.852
DecisionTable	0.840	0.866	0.841	0.684	0.840
DTNB	0.851	0.863	0.852	0.701	0.889
JRip	0.840	0.870	0.841	0.685	0.837
NNge	0.742	0.740	0.739	0.461	0.726
OneR	0.845	0.873	0.846	0.695	0.862
PART	0.845	0.869	0.846	0.694	0.843
Ridor	0.851	0.866	0.852	0.702	0.861
ZeroR	0.582	0.339	0.429	0.000	0.485
BFTree	0.835	0.855	0.836	0.672	0.873
DecisionStump	0.856	0.888	0.856	0.716	0.836
J48	0.845	0.869	0.846	0.694	0.847
LADTree	0.830	0.848	0.831	0.662	0.829
LMT	0.840	0.855	0.841	0.681	0.862
NBTree	0.861	0.873	0.862	0.721	0.876
RandomForest	0.840	0.855	0.841	0.681	0.854
RandomTree	0.830	0.848	0.831	0.662	0.838
REPTree	0.861	0.887	0.862	0.725	0.852
SimpleCart	0.840	0.858	0.841	0.682	0.844

Table 2: Performance classification measures for dataset1.

Secondly, in order to find out which algorithms perform best for each dataset taking several classification measures into account, we used the Iman&Davenport non-parametric statistical test [5]. This test was repeated for each of the 32 datasets and produced an ordered list of algorithms with their final rank (average rank of the 19 algorithms over the 5 performance measures), in such a way that the algorithm with the best rank (highest position in each list) is the one that performs best for the measures under consideration. According to the Iman&Davenport test, if the null-hypothesis is accepted, we state that all the algorithms are equivalent, i.e., they have a similar behavior. In contrast, if the null-hypothesis is rejected, we state that there are differences between the algorithms. For the 32 tests performed in our experiment at a significance level of alpha=0.1, the null-hypothesis was rejected, thus indicating that significant differences exist between classifiers.

Therefore, in order to reveal such performance differences, a post-hoc test needs to be carried out. The Bonferroni-Dunn test [5] can be applied, since all the algorithms were compared against a control algorithm (the algorithm with the highest rank), the focus being on all the possible pairwise comparisons among them. The critical value revealed by this test at the same significance level of alpha=0.1 was 9.5331. Therefore, for each dataset, that value was added to the rank of the control algorithm, and the algorithms whose rank belongs to the interval [highest rank, highest rank + critical value] are the set of best algorithms recommended for that particular dataset, given that there are no significant differences between them.

For instance, the set of best algorithms recommended for dataset1 are shown in Table 3, in which the critical interval is [2,2+9.5331]. The remaining 10 algorithms are not recommended due to the fact that their rank is over the upper limit.

Algorithm	Ranking
NBTree	2
REPTree	2.667
DecisionStump	5
DTNB	5.25
Ridor	5.667
OneR	6.333
ConjunctiveRule	8.083
J48	8.417
PART	8.833

Table 3: Ranking of the algorithms recommended for dataset6

Finally, in order to recommend algorithms for a new dataset, we used a nearest neighbor (1-NN) approach [1]. We used the unweighted normalized Euclidean distance to find the closest dataset to the new one. In the case of categorical value (the domain feature), the distance considered was 0 in the case of matching and 1 otherwise. Then, the set of best algorithms previously calculated to the most similar dataset was recommended for the new dataset.

We carried out a second experiment to compare the results obtained when the different types of features that characterize the datasets were used. We noticed that distinct nearest neighbors were obtained for the same dataset depending on the features used. For instance, the nearest neighbors obtained for dataset1 when using different feature combinations are shown in Table 4.

Statistic	Complex	Statisic+ Complex	Statistic+ Complex+ Domain
Dataset13	Dataset11	Dataset11	Dataset22

Table 4: Nearest neighbors for dataset1 depending on the combination of used features.

As can be seen in Table 4, dataset13 (from forum domain) is the most similar to dataset1 (from report domain) when only statistics features (see Table 1) are used, but dataset11 (from forum domain) is the most similar when complex and statistics features are used together, and finally, dataset22 (from report domain) is the most similar when all the features that also take the domain into account are used (see Table 1).

Four separate tests using the hold-one-out method were directed to check which combination of features (by employing only statistical features, only complexity, both statistical and complexity, and also the domain attribute) enables the best recommendation to be obtained. Following this hold-one-out procedure, first we calculated the similarity between each dataset and the remaining 31 datasets to select the most similar dataset. Next, the set of recommended (and previously calculated) algorithms for each dataset is considered as the real output, whereas the set of algorithms of its nearest neighbor is the predicted one. Then, several evaluation measures commonly used in pattern recognition and information retrieval systems (such as search engines and recommender systems) were computed to evaluate the quality of the recommendations. Precision and recall are the metrics employed, which are defined in terms of a set of retrieved documents in an information retrieval domain, but in this work, they are defined in terms of retrieved algorithms:

$$precision = \frac{|\{real\_algorithms\} \cap \{predicted\_algorithms\}|}{|predicted\_algorithms|}$$

$$recall = \frac{|\{real\_algorithms\} \cap \{predicted\_algorithms\}|}{|real\_algorithms|}$$

There is an inverse relationship between precision and recall, in such a way that obtaining higher values of one measure means obtaining lower values for the other. Nevertheless, there is another measure, called F-Measure, which combines both precision and recall and is computed as the harmonic mean of both:

$$F-Measure = 2 \times \frac{precision \times recall}{precision + recall}$$

The F-measure results achieved for the four combinations of features used are shown in a box plot or box-and-whisker diagram that shows the smallest observation (sample minimum), lower quartile (Q1), median (Q2), upper quartile (Q3), and largest observation (sample maximum), as can be seen in Figure 2. As can be observed, better results are obtained when the statistical and complexity features are considered jointly rather than when they are considered independently. Moreover, the best results are reached when the domain attribute is also included.

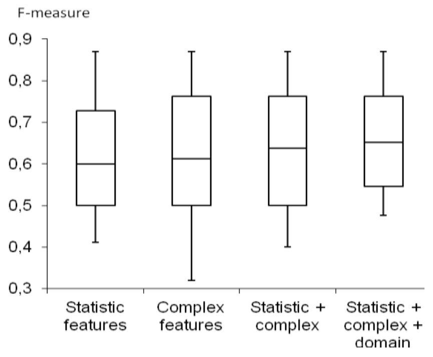


Figure 2. Blox plot of the F-measure.

5. CONCLUSIONS

In this paper, meta-learning has been used to address the problem of recommending a subset white-box classifier from Moodle datasets. Several classification performance measures are used together with several statistical test to rank and select a subset of algorithms. Results show that complexity and domain features used to characterize datasets can improve the quality of the recommendation. For future work, we plan to extend the experimentation, for example, using more datasets, algorithms (including black box models), characteristics, evaluation measures, etc. Future research may employ a greater number of classification datasets from other sources or other kinds of education systems (primary, secondary, higher, special education, …) in which different specific domain features to characterize datasets can be used. A further line of research would be to develop more advanced off-line procedures, such as the employment of several K-NN neighbors instead of the 1-NN, and methods for merging several rankings and subsets of algorithms in neighboring datasets.

6. ACKNOWLEDGMENTS

This work was supported by the Regional Government of Andalusia and the Spanish Ministry of Science and Technology projects, P08-TIC-3720 and TIN-2011-22408, and FEDER funds.

7. REFERENCES

[1] Aha, D., Kibler, D. Instance-based learning algorithms. Machine Learning. 6, 37-66, 1991.

[2] Asuncion, A, Newman, D.J.. UCI Machine Learning Repository, University of California, Irvine, CA, 2007. (http://www.ics.uci.edu/mlearn/MLRepository.html).

[3] Bhatt, N. Thakkar, A. Ganatra, A. A Survey & Current Research Challenges in Meta Learning Approaches based on Dataset Characteristics. International Journal of Soft Computing and Engineering, 2(10), 234-247, 2012.

[4] Brazdil, P., Giraud-Carrier, C., Soares, C. and Vilalta, R. Metalearning: Applications to Data Mining. *Series: Cognitive Technologies*. Springer, 2009.

[5] Demsar, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7 1-30, 2006.

[6] Hämäläinen, W., Vinni M. Classifiers for educational data mining; Handbook of Educational Data Mining. Chapman & Hall/CRC. 2011.

[7] Ho T.K., Basu M. Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):289-300, 2002.

[8] Molina, M.M., Romero, C. Luna, J.M. Ventura. S. Meta-learning approach for automatic parameter tuning: A case study with educational datasets. 5th International Conference on Educational Data Mining, Chania, Greece, 180-183, 2012.

[9] Orriols-Puig A., Macià N. & Ho T.K. Documentation for the data complexity library in C++. Technical report, La Salle - Universitat Ramon Llull, 2010.

[10] Romero, C. and Ventura, S. Data Mining in Education. *Wire Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 3:12-27. 2013.

[11] Romero, C., Espejo, P., Romero, R., Ventura, S. Web usage mining for predicting final marks of students that use Moodle courses. *Computer Applications in Engineering Education*. 21(1): 135-146. 2013.

[12] Song, Q, Wang, G, Wang, C. Automatic recommendation of classification algorithms based on dataset characteristics. *Pattern recognition*. 45, 2672–2689, 2012.

[13] Witten, I. H., Eibe, F. and Hall, M.A. *Data Mining: Practical Machine Learning Tools and Techniques, Third Edition*. Morgan Kaufman Publishers, 2011.

Investigating the Effects of Off-Task Personalization on In-System Performance and Attitudes within a Game-Based Environment

Erica L. Snow
Tempe, AZ, USA
Arizona State University
Erica.L.Snow@asu.edu

G. Tanner Jackson
Tempe, AZ, USA
Arizona State University
Tanner.Jackson@asu.edu

Laura K. Varner
Tempe, AZ, USA
Arizona State University
Laura.Varner@asu.edu

Danielle S. McNamara
Tempe, AZ, USA
Arizona State University
Danielle.McNamara@asu.edu

ABSTRACT

The current study investigates the relation between personalizable feature use, attitudes, and in-system performance in the context of the game-based system, iSTART-ME. This analysis focuses on a subset (n=40) of a larger study (n=126) conducted with high school students. The results revealed a positive relation between students' frequency of interactions with personalizable features and their self-reported engagement and perceived system control. Students who frequently interacted with personalizable features also demonstrated better overall in-system performance compared to students who interacted with these features less often. The current paper adds to the growing literature supporting the potential positive impact that personalizable features have on students' attitudes and performance in adaptive learning environments.

Keywords

Personalization, attitudes, game-based features, off-task behaviors

1. INTRODUCTION

A growing trend in the field of adaptive learning environments has been the study of educational games on users' interest and engagement during learning [1-2]. When games are incorporated into these learning environments, students have demonstrated increased engagement and motivation [3-4]. However, few studies have investigated how features within educational games may lead to off-task behaviors and ultimately influence in-system performance (notable exceptions include [5-6]).

An exploration of the interactive features in educational game environments may allow researchers to identify the aspects of system interfaces that benefit or hinder students' learning. Developers have integrated many types of interactive choice-based features into educational game interfaces, including: personalizable avatars, interactive maps, and customizable background colors. These features have been found to increase student motivation and engagement [7-8]; however, the learning impact of these potentially off-task behaviors is relatively unclear.

Off-task behaviors have been defined as any behavior that does not involve the specific learning task designated to the student [9]. Although off-task behaviors are frequently observed within classrooms [10], tutoring systems [11], and workplaces [12], the impact of these behaviors on learning remains inconclusive [6,9, 13].

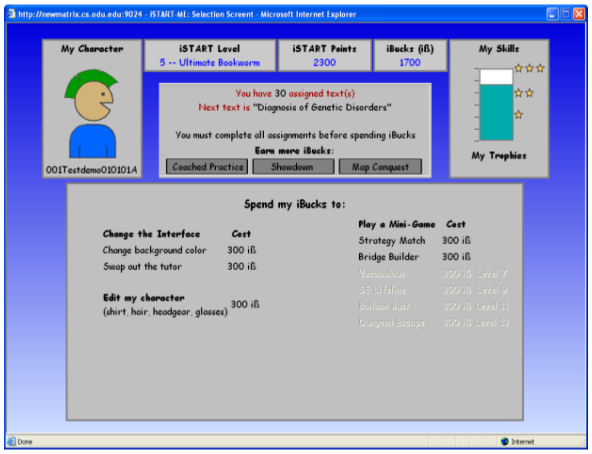
For instance, Rowe, McQuiggan, Robinson, and Lester (2009) found that off-task choices within games were negatively correlated to posttest measures of learning performance. These findings are in line with previous work that has suggested a negative relation between off-task behaviors and student learning [5,9]. However, Rai and Beck (2012) found no relation between students' interactions with off-task features and learning performance. These mixed results render it challenging for researchers to decipher the true cost-benefit ratio of game-based features within educational learning environments.

Although previous work has provided some insight into the impact of off-task behaviors on student learning, they have yet to find a "sweet spot" amongst variables such as students' use of personalizable features that elicit off-task behaviors, in-system performance measures, and student attitudes. The current study investigates these relations within the context of the game-based system, iSTART-ME [14].

1.2 iSTART-ME

The Interactive Strategy Training for Active Reading and Thinking – Motivationally Enhanced (iSTART-ME) system is a game-based learning environment designed to improve students' reading comprehension ability. This system is an extension of a previous version of iSTART, which provides students with instruction and practice using reading comprehension strategies while reading challenging texts [15].

The iSTART-ME interface is controlled through a selection menu where students can read and self-explain texts, personalize avatars, play mini-games, earn points, win trophies, and view their progress in the system (see Figure 1 for menu screenshot). Students earn points and win trophies by playing games. As students accumulate points, they advance through a series of achievement levels. Each subsequent level requires more points to proceed than the previous level. Within iSTART-ME, in-system performance can be measured via students' achievement levels and trophies. Together, these measures represent students' commitment to and quality of strategy practice.


http://newmatrix.cs.odu.edu:9024 - iSTART-ME: Selection Screent - Microsoft Internet Explorer

My Character	iSTART Level	iSTART Points	iBucks (iß)	My Skills
001Testdemo010101A	5 -- Ultimate Bookworm	2300	1700	My Trophies

You have 30 assigned text(s)
Next text is "Diagnosis of Genetic Disorders"

You must complete all assignments before spending iBucks

Earn more iBucks:

Spend my iBucks to:

Change the Interface	Cost
Change background color	300 iß
Swap out the tutor	300 iß
Edit my character (shirt, hair, headgear, glasses)	300 iß

Play a Mini-Game	Cost
Strategy Match	300 iß
Bridge Builder	300 iß
Vocabulous	300 iß Level 7
SE Lifeline	300 iß Level 9
Balloon Bust	300 iß Level 11
Dungeon Escape	300 iß Level 13

Figure 1. Screenshot of iSTART-ME Interface

System points within iSTART-ME also serve as a form of currency (iBucks) that students can use to buy rewards on the main interface. There are two primary incentives that students can unlock with their earned iBucks: mini-games and personalizable features. Mini-games were added to iSTART-ME to provide students with opportunities to practice the reading comprehension strategies in a game-based environment. Within iSTART-ME, mini-games are considered on-task because they are extensions of the overall learning goal of the system. However, the second incentive that students can unlock, personalizable features, is considered off-task, because it is tangential to the learning task.

Within iSTART-ME, personalizable features include changing background colors, editing a pedagogical agent and customizing an avatar. Students can spend their earned iBucks on these features as many times as they choose for a variety of configurations (see Figure 2 for avatar configuration examples). Personalizable features were added to the system interface as a means to increase students' control and engagement in the iSTART-ME environment. With the addition of these potentially distracting features, it is important to investigate the extent to which students' interactions with these elements impact their attitudes and performance within the system.

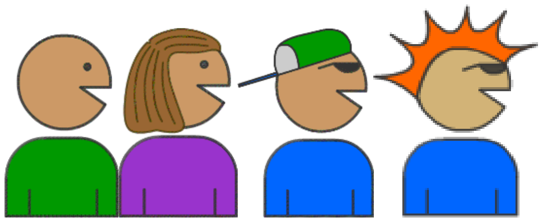


Figure 2. Examples of Avatar Configurations

2. METHODS

Participants in this study included 40 high school students from a mid-south urban environment. The sample included in the current work is a subset of 126 students who originally participated in a larger study that compared three conditions: iSTART-ME, the original version of iSTART, and a no-tutoring control [17]. The current study focuses on the students who were assigned to the iSTART-ME condition. These students had access to the full game-based system in which the game-based interface features were available.

All students completed an 11-session experiment consisting of a pretest, 8 training sessions, a posttest, and a delayed retention test. During the first session, students completed a pretest that included survey measures assessing motivation, prior self-explanation ability, prior reading ability [17], and attitudes toward technology and games. During the following 8 sessions, students engaged with the iSTART-ME interface for a minimum of 1 hour, where they could play games, interact with texts and personalize system features. After training, students completed a posttest, which included measures that were similar to the pretest. Finally, 5 days after the posttest, students returned to complete a retention test, which contained measures similar to the pretest and posttest (i.e., self-explanation and attitudinal measures).

2.1 Self-report measures

Using posttest self-report surveys, we assessed students' attitudes toward the iSTART-ME system (see Table 1 for selected examples). All responses were on a scale of 1 (Strongly Disagree) to 6 (Strongly Agree). Survey measures were combined to create composite scores for boredom, enjoyment, and motivation.

Table 1. Selected Examples of Posttest Self-Report Measures

Dependent Measure	Response Statement	Response Scale
Enjoyment	*"I had fun using the computer system"*	1 - 6
Boredom	*"I felt bored"*	1 - 6
Motivation	*"I was motivated to participate in this study"*	1 - 6
Lack of Control	*"I felt like I had no control over the system"*	1 - 6

1 (Strongly Disagree) to 6 (Strongly Agree)

3. RESULTS

We examined interactions with personalizable features and their relation to students' performance and attitudes during training within iSTART-ME. Using the process data from students' interactions, we calculated the number of times students spent their earned iBucks on personalizable features. We first examined how off-task personalization of any kind (avatar, background theme, or agent) related to in-system performance (achievement levels and trophies won) and posttest attitudinal composite measures (i.e., enjoyment, boredom, and motivation) and a single measure of perceived lack of control. The correlation results in Table 2 indicate that students' number of avatar edits was marginally negatively related to posttest boredom and significantly negatively related to perceived lack of control.

Hence, there was a weak negative relation between students' perceived boredom and lack of control within the system and avatar edits. However, no other personalization feature in the system had a marginal or significant relationship to any of the other dependent variables.

Table 2. Correlations between Frequency of Off-task Personalization Edits and Dependent Measures

Dependent Measure	Avatar Edits (n=30)	Background Theme Edits (n=15)	Pedagogical Agent Edits (n=17)
Achievement Level	.26	.35	.06
Total Trophies Won	.33	.39	-.10
Enjoyment Composite	.11	.07	.14
Boredom Composite	-.35(*M*)	-.13	-.31
Motivation Composite	.19	.15	-.04
Lack of Control	-.36*	-.18	-.21

*p < .05; p<.10 (*M*)

3.1 Differences in avatar interactions

Table 2 reveals that the customizable avatar was the only personalizable feature that showed a significant relation to student attitudes. These results also demonstrate that none of the three personalizable features showed a significant relation to any of the in-system performance measures. Therefore, we focus the remainder of our analyses on students' interactions with the customizable avatar feature. To further explore the relation between avatar editing, attitudes, and in-system performance, we created two categories of students: those who edited their avatar (editors) and those who did not (non-editors). Out of the 40 total participants who completed the iSTART-ME condition, 30 students made at least one edit to their avatar. A median split was performed including only those students who edited their avatar at least once. This median split resulted in 19 students who made at least three or more edits (high editors) and 11 students who performed only one or two avatar edits (low editors). This median split helps to profile students and identify varying patterns of on-line interactions within game-based learning systems.

3.2 Group differences in avatar edits

Differences between high and low avatar editors' attitudes and in-system performance were examined using separate one-way ANOVAs. These analyses revealed a significant difference between high and low editors in terms of overall in-system performance (see Table 3). Compared to high editors, low avatar editors had significantly lower system achievement levels, $F(1,28)=4.22$, $p< .05$, and significantly fewer trophies, $F(1,28)= 5.24$, $p< .05$. These results indicate that students who engaged in more off-task behaviors (i.e., more than two avatar edits) showed significantly better in-system performance relative to students who engaged in fewer off-task behaviors (i.e., only one or two avatar edits).

One potential reason for these editor differences may, in part, be due to the system design. Within iSTART-ME, points are used to unlock various features. Thus, students who earn more points will be able to spend more on off-task features. To examine this issue further, a ratio of points spent to points earned was calculated for each student. A one-way ANOVA on this spending ratio revealed no significant differences between high and low editors, $F(1,28)=.794$, $p=.381$. This finding indicates that high and low editors spent the same relative amount of points throughout their interactions. Although high and low editors spent the same relative amounts, the previous results suggest that high editors were more likely to spend those points on the off-task personalization features, whereas low editors spent their points elsewhere (i.e., they remain on-task).

Separate ANOVAs were also conducted to examine the relation between avatar edits and students' self-reported posttest attitudes toward the system (see Table 3). The results indicated that students who were classified as high editors reported less boredom, $F(1,28)=5.84$, $p<.05$, compared to high editors. Interestingly, low editors reported feeling a higher lack of control within the system, $F(1, 28)=7.62$, $p<.01$, compared to high editors.

Table 3. Mean Overall Performance and Attitudes Scores per Group

Dependent Measure	Low Avatar Editors M (SD)	High Avatar Editors M (SD)
Achievement Level	13.76 (6.25)	18.45 (5.66)
Trophies Won	15.15 (11.77)	36.00 (36.98)
Boredom Composite	2.30 (.93)	1.55 (.59)
Lack of Control	2.36 (1.30)	1.18 (.75)

4. DISCUSSION

The incorporation of educational games within learning environments has demonstrated positive impacts on student engagement [1, 4]. However, many features within these games may promote off-task behaviors. The current study aimed to gain a deeper understanding of the relations among these features, in-system performance, and students' attitudes.

In line with previous work, our results demonstrated significant negative relations among students' interactions with personalizable features, boredom, and lack of control. These results replicate previous work suggesting that personalization potentially augments students' investment and perception of control within a system [7]. ANOVAs revealed distinct differences between high and low avatar editors for their self-reported attitudes and in-system performance. High editors advanced to higher achievement levels and won more trophies compared to low editors. High editors also expressed less boredom and higher levels of perceived control in the system.

In contrast to some prior work [6], the current analyses on off-task behaviors, in-system performance, and student attitudes provide tentative support for the inclusion of personalized features within adaptive learning environments. Interestingly, the most frequently utilized and influential feature was the customizable avatar. One hypothesis for this effect is that creating a personal avatar may increase students' investment within the system [8]. Future work will explore the relation between avatars and investment, along with the potential benefit that this may have on persistence (i.e., students returning to the training environment).

Additionally, further work is needed to investigate how students' motivation and attitudes mediate choices within the system (both on- and off-task) to affect overall system learning outcomes. The current work begins to explore this complex interaction and examines potential features that can engage students over an extended training period while maintaining performance quality. Investigating these dynamic elements within learning environments will advance our understanding of the proper balance between system designs that promote high levels of both engagement and performance.

Acknowledgments

This research was supported in part by the Institute for Educational Sciences (IES R305G020018-02; R305G040046, R305A080589) and National Science Foundation (NSF REC0241144; IIS-0735682). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the IES or NSF

5. REFERENCES

[1] Malone, T., and Lepper, M. 1987. Making learning fun: a taxonomy of intrinsic motivations for learning. In *Aptitude, learning, and instruction*, R. Snow and M. Fair, Eds. Vol. 3. Cognitive and affective process analyses. Erlbaum, Hillsdale, NJ, 223-253.

[2] Moreno, R., and Mayer, R. E. 2005. Role of guidance, reflection, and interactivity in an agent-based multimedia game. *Journal of Educational Psychology*, 97, (2005) 117–128.

[3] Jackson, G. T., and McNamara, D. S. 2011. Motivational impacts of a game-based intelligent tutoring system. In *Proceedings of the 24th International Florida Artificial Intelligence Research Society Conference*, (Palm Beach, FL, May 18-20, 2011) FLAIRS, AAAI, 519-524.

[4] Clark, D., Nelson, B., Sengupta, P., and D'Angelo, C. 2009. Rethinking science learning through digital games and simulations: Genres, examples, and evidence. In *Learning science: Computer games, simulations, and education workshop*, (Washington DC, October 2009). National Academy of Sciences.

[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 Computer-Human Interaction*, (Vienna, Austria, April 24 -30, 2004) CHI, ACM Press, 383–390.

[6] Rowe, J., McQuiggan, S., Robison, J., and Lester, J. 2009. Off-task behavior in narrative-centered learning environments. In *Proceedings of the Workshop on Intelligent Educational Games at the 14th Annual Conference on Artificial Intelligence in Education*, (Brighton, UK, July 06 - 10, 2009). AIED, IOS Press, 99-106.

[7] 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, (1996) 715– 730.

[8] Facer, K., Joiner, R., Stanton, D., Reid, J., Hull R., and Kirk, D. 2004. Savannah: Mobile gaming and learning? *Journal of Computer Assisted Learning*. 20, (2004) 399-409.

[9] Cocea, M., Hershkovitz, A., and Baker, R. S. 2009. The Impact of Off-task and Gaming Behaviors on Learning: Immediate or Aggregate? *In Proceedings of the 14th Annual Conference on Artificial Intelligence in Education,* (Brighton, UK, July 06 -10, 2009). AIED, IOS Press, 507-514.

[10] Wallace, J. C., and Vodanovich, S. J. 2003. Workplace safety performance: conscientiousness, cognitive failure, and their interaction. *Journal of Occupational Health Psychology*, 8, (2003) 316-327.

[11] Baker, S. J. 2007. Modeling and understanding students' off-task behavior in intelligent tutoring systems. In *Proceedings of the ACM Conference on Human Factors in Computer Systems*, (San Jose, CA, April 28 – May 03, 2007).CHI, ACM Press, 1059–1068.

[12] Austin, J. L., and Soeda, J. M. 2008. Fixed-time teacher attention to decrease off-task behaviors of typically developing third graders. *Journal of Applied Behavior Analysis*. 41, (2008) 279–283.

[13] Rai, D., and Beck, J. 2012. Math learning environment with game-like elements: An experimental framework. *International Journal of Game Based Learning*, 2, (2012), 90- 110.

[14] Jackson, G. T., Boonthum, C., and McNamara, D. S. 2009. iSTART-ME: Situating extended learning within a game-based environment. In *Proceedings of the Workshop on Intelligent Educational Games at the 14th Annual Conference on Artificial Intelligence in Education*, (Brighton, UK, July 06 -10, 2009). AIED, IOS Press, 59-68.

[15] McNamara, D. S., O'Reilly, T., Rowe, M., Boonthum, C., and Levinstein, I. B. 2007. iSTART: A web-based tutor that teaches self-explanation and metacognitive reading strategies. In *Reading comprehension strategies: Theories, interventions, and technologies*, D. S. McNamara, Ed. Erlbaum. Mahwah, NJ, 397-420.

[16] Jackson, G. T., and McNamara, D. S. In Press. Motivation and performance in a game- based intelligent tutoring system. *Journal of Educational Psychology*. (in press).

[17] MacGinitie, W., and MacGinitie, R. 1989.*Gates MacGinitie reading tests*. Riverside. Chicago, I

Students' Walk through Tutoring: Using a Random Walk Analysis to Profile Students

Erica L. Snow
Arizona State University
Tempe, AZ, USA
Erica.L.Snow@asu.edu

Aaron D. Likens
Arizona State University
Tempe, AZ, USA
Aaron.Likens@asu.edu

G. Tanner Jackson
Arizona State University
Tempe, AZ, USA
TannerJackson@asu.edu

Danielle S. McNamara
Arizona State University
Tempe, AZ, USA
Danielle.McNamara@asu.edu

ABSTRACT

The purpose of this study was to investigate students' patterns of interactions within a game-based intelligent tutoring system (ITS), and how those interactions varied as a function of individual differences. The analysis presented in this paper comprises a subset (n=40) of a larger study that included 124 high school students. Participants in the current study completed 11 sessions within iSTART-ME, a game-based ITS, that provides training in reading comprehension strategies. A random walk analysis was used to visualize students' trajectories within the system. The analyses revealed that low ability students' patterns of interactions were anchored by one feature category whereas high ability students demonstrated interactions across multiple categories. The results from the current paper indicate that random walk analysis is a promising visualization tool for learning scientists interested in capturing students' interactions within ITSs and other computer-based learning environments over time.

Keywords

Intelligent Tutoring Systems, sequential pattern analysis, random walk analysis, individual differences

1. INTRODUCTION

A growing trend in the field of educational technology has been to use aggregated or summative analysis to trace students' interactions with game-based features inside of Intelligent Tutoring Systems (ITSs) [1-5]. These analyses capture students' interactions with the system overtime and at fixed intervals [3-5]. For example, aggregated analysis on the frequency of students' utilization of game-based features across multiple training sessions found that patterns of interactions varied as a function of individual differences in performance orientation [4]. Similarly, summative methods have been used to investigate how the availability of game-based elements inside of a system impacts students' overall enjoyment [3].

Although aggregated and summative analyses shed some light on users' overall system interactions, those statistical methods cannot trace nuances in students' paths in adaptive learning environments. The current study utilizes sequential pattern analysis to reveal distinct differences in students' propensity to interact with various game-based features.

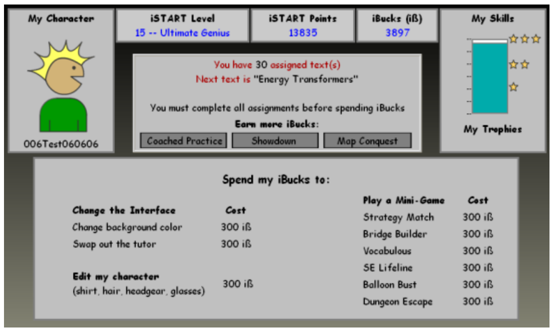
Sequential pattern analysis places an emphasis on fine-grained detail. Therefore, this means of analysis may give researchers a deeper understanding of students' interactions in a system by examining nuances in patterns overtime. Random walk analysis is just one of many available sequential pattern analysis tools (e.g., Euclidean distance and dynamic time warping). A random walk analysis is a mathematical tool that generates a spatial representation of a path [6]. This technique has been used in a variety of domains such as economics [7], ecology [6], psychology [8] and medicine [9-10]. For instance, this method has been used in the study of genetics and aptly renamed DNA walks [9-10]. DNA walks provide researchers with a simple visualization of genome codes and patterns [10]. Overall, random walk analysis has been a useful technique for visualizing the nuances of fine grain patterns in categorical data over time.

The current study employs a random walk analysis in the visualization of students' interactions with the game-based ITS, iSTART-ME. We are particularly interested in examining how students' patterns of interactions with game-based features vary as a function of individual differences across numerous sessions. Investigating these patterns as they unfold over time is expected to provide researchers with a deeper understanding of variations in students' tendency to use game-based features.

1.2 iSTART-ME

The Interactive Strategy Training for Active Reading and Thinking – Motivationally Enhanced (iSTART-ME) is a game-based ITS developed on top of an existing system, iSTART [11]. iSTART was developed to provide instruction and practice in comprehension strategies and improve student comprehension of difficult science texts.

iSTART-ME training includes three initial phases where reading comprehension strategies are *introduced*, *demonstrated*, and *practiced* (phases are discussed in more detail in [1]). A fourth phase includes *extended practice,* where students apply the strategies across numerous texts and multiple sessions. iSTART-ME situates this extended practice within a game-based selection menu (see Figure 1), which includes: generative practice games, personalizable features, achievement screens, and identification mini-games. *Generative practice games* are designed so that students must generate text and practice applying the reading comprehension strategies. *Identification mini-games* provide examples and require students to identify the specific strategy being used. *Personalizable features* are incentives designed to afford students a higher locus of control and investment with the environment. *Achievement screens* allow students to track their progress and performance across the iSTART-ME system. All four features are available during interactions with iSTART-ME, and students are free to choose among the options at any time.


My Character

006Test060606

iSTART Level
15 -- Ultimate Genius

iSTART Points
13835

iBucks (iß)
3897

You have 30 assigned text(s)
Next text is "Energy Transformers"

You must complete all assignments before spending iBucks
Earn more iBucks:

Spend my iBucks to:

Change the Interface	Cost
Change background color	300 iß
Swap out the tutor	300 iß
Edit my character (shirt, hair, headgear, glasses)	300 iß

Play a Mini-Game	Cost
Strategy Match	300 iß
Bridge Builder	300 iß
Vocabulous	300 iß
SE Lifeline	300 iß
Balloon Bust	300 iß
Dungeon Escape	300 iß

Figure 1. Screenshot of iSTART-ME Interface

2. METHODS
2.1 Participants

Participants in the current work (n=40) were a subset of 124 high school students who participated in a study at a large university campus in the Mid-Southern United States [12]. The current analyses focus only on those students who were randomly assigned to interact with the game-based iSTART-ME system (other students in the original study were assigned to an ITS or a no-tutoring control). The students included here consisted of 20 males and 20 females, with an average age of 16 years.

2.2 Procedure

Students in this study completed an 11-session experiment that consisted of a pretest, 8 training sessions within iSTART-ME, a posttest, and a delayed retention test. During session 1, participants completed a pretest to assess their attitudes, motivation, prior self-explanation (SE) quality, vocabulary knowledge, and prior reading ability. SE quality was measured at pretest using the iSTART algorithm, which ranges from 0 (poor) to 3 (good) [13]. This score provides a rough indicator for the amount of cognitive processing involved, and represents the quality of a student's self-explanation [14]. Prior reading ability was assessed using the Gates MacGinitie Reading Test [15]. Students interacted with the iSTART-ME system during sessions 2 through 9. During session 10, students completed a posttest, which included measures similar to the pretest. Finally, five days after the posttest, students returned to complete a retention test, consisting of similar self-explanation and comprehension measures.

2.3 Analysis

The current study employs a random walk algorithm to visualize student interaction patterns across time (sessions 2 through 9). Game-based features were grouped into four distinct categories and each was assigned to a vector on an X, Y scatter plot. Although the current study used only four dimensions, the number of dimensions that can be included when using random walk analyses is relatively unlimited. The walk proceeds by placing an imaginary particle at the origin (0, 0) and, each time a participant interacts with a specific feature, the particle moves in the direction of the vector assignment (see Table 1 for directional assignment).

Table 1. Directional Assignment per Interaction

System Interaction	Directional assignment
Generative Practice Games	-1 on X-axis (move left)
Identification Mini-Games	+1 on Y-axis (move up)
Personalizable Features	+1 on X-Axis (move right)
Achievement Screens	-1 on Y-axis (move down)

Figure 2 is an example of what a walk may look like for a student with four interactions corresponding to the following sequence: 1) generative practice game (move left), 2) identification mini-game (move up), 3) personalizable feature (move right), and 4) a second identification mini-game (move up). These simple rules are used for every interaction a student makes and give us their "walk" through the system.

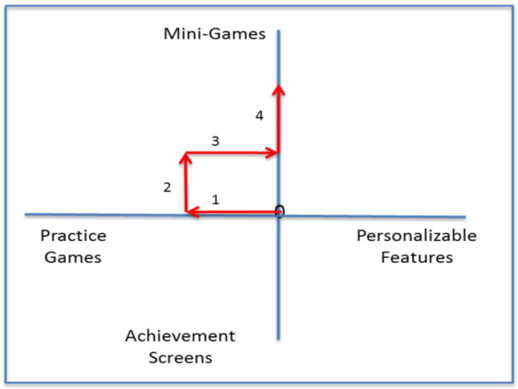


Figure 2. Example of Directional Rules for Walk Sequence

3. RESULTS

The current study examined students' patterns of interactions with game-based features and how they may vary as a function of individual differences. Students' data logs from their eight sessions in iSTART-ME were used to categorize every interaction into one of the four possible game-based feature types: generative practice games, personalizable features, achievement screens, and identification mini-games. The random walk algorithm was then used to construct a unique pattern for each participant (see Figure 3 for one student's complete "walk" pattern).

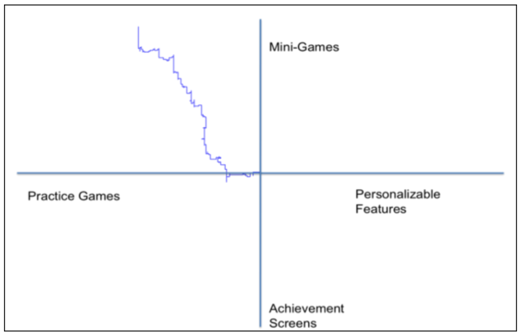


Figure 3. Individual Walk for a Student in the iSTART-ME Study

A slope was calculated as a coarse measure of each student's unique walk pattern. This slope provides a measure of each student's interaction trajectory across the time spent within the game-based portion of iSTART-ME. Slope calculations obscure a portion of the variability in walk patterns. On the one hand, some information is lost in that analysis. On the other hand, this metric provides valuable insight into students' interaction trajectories over time. Utilizing these slopes, we examined the relation between slope magnitude and individual differences in prior reading ability, prior SE quality, and prior vocabulary knowledge (see Table 2). A correlation analysis revealed that the magnitude of walk slopes was negatively related to prior reading ability and prior SE quality. This analysis also indicated a marginally significant relation between slopes and pretest vocabulary knowledge. Students with higher reading and SE quality pretest scores demonstrated a more vertical trajectory in their patterns of interactions. That is, higher ability students were more likely to interact with both generative practice games and identification mini-games and were less likely to hover around the generative practice game function.

Table 2. Correlations between Slope of Students' Walks and Individual Difference Variables

Individual Difference Variables	***Slope (r)***
Prior Reading Ability	-.593**
Prior SE Quality	-.496**
Vocabulary Knowledge	-.297*(M)*
p<.05*; p<.01**; *M*= Marginally Significant, p<.10	

A median split on pretest reading comprehension scores was used to classify students as either high or low ability. A one-way ANOVA examining trajectory differences between high and low reading ability students indicated that high reading ability students had significantly steeper slopes (M = -1.76, SD = 0.96) compared to low reading ability students (M = -0.58, SD = 0.56, $F(1,38)$ = 23.58, p < 0.001[1]. The effect size for this relation (η^2 = 0.38) suggests a moderate to high practical significance [16]. Figure 4 provides a visualization of those differences. In summary, the results indicate that low reading ability students are more likely to use the generative practice game features and are less prone to interact with the other game features.

A similar ANOVA examined differences in walk slopes for students with high and low pretest SE quality scores[1] These results reveal that students with higher quality self-explanations at pretest had significantly steeper slopes (M = -1.62, SD = 1.00) compared to those with low SE quality (M = -0.62, SD = 0.60, $F(1,38)$ = 14.99, p < 0.001). The effect size for this relationship (η^2 = 0.68) suggests a high practical significance [16]. These results indicate that students who generated low quality self-explanations prior to training tend to hover consistently around a specific feature (generative practice games) whereas, high SE quality students seem to interact at a more balanced rate between generative practice games and identification mini-games. Figure 5 provides a visualization of these differences.

[1] Multivariate regression analyses confirmed results of the median-split analyses.

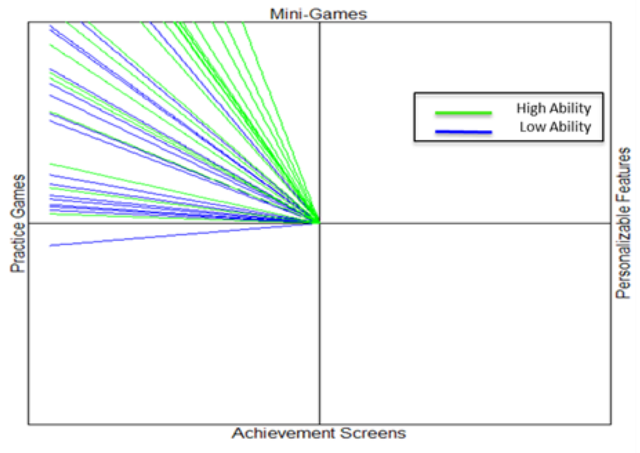


Figure 4. Visualization of Slope Trajectories for High and Low Reading Ability Students

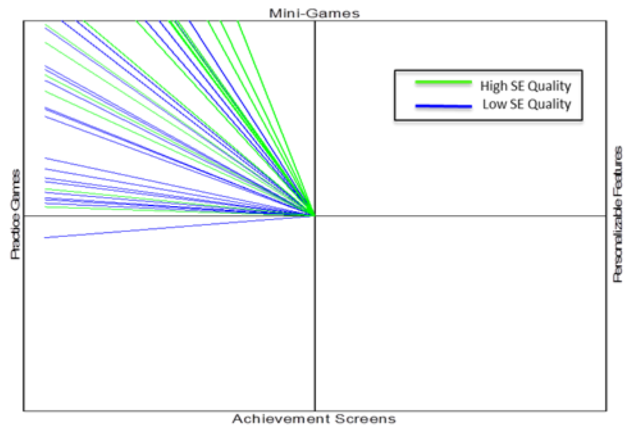


Figure 5. Visualization of Slope Trajectories for Students with High and Low Prior Self-Explanation Quality Scores

4. DISCUSSION

The current study used a random walk analysis to view sequences of patterns in students' interactions within the game-based ITS, iSTART-ME. We suggest that sequential pattern analysis may benefit learning scientists by providing a new method for tracking and viewing students' interactions with game-based features across time. Using the slopes of each student's unique walk we found that there was a relation between a students' trajectory through the system and their prior reading comprehension ability and prior self-explanation (SE) quality scores. Investigating this relation further, we found that students with higher reading ability and higher SE quality scores showed significantly different trajectories compared to low reading ability and those students with lower quality self-explanations. Low ability students tended to interact more with generative practice games, whereas high ability students interacted in a more balanced way with both generative practice games and identification mini-games.

The implications of the current study are promising for researchers in two ways. First we have shown that random walk analysis can be used to view fine-grained patterns of interactions within an ITS. Researchers can use this analysis technique to track students' interactions within a system across time. Although this method is well known and used in many diverse fields [6-9], this is the first time, to our knowledge, that random walk has been used to investigate users' trajectories inside of an adaptive learning environment. Secondly, we have shown that individual differences can be distinguished by using slopes derived from each student's walk. These slopes show us the trajectory of students' interactions and what features are anchoring them. This may be useful for real time analysis of system usage. For instance, if students' patterns of interactions are too stagnant, the system may need to prompt them to interact with a different feature. Random walk analyses may also improve the adaptability of adaptive environments by allowing researchers to monitor and track users' behaviors inside the system.

The current analysis opens the door for a wide range of time series analysis techniques, a full description of which is beyond the scope of this paper. For example, we are currently exploring the benefit that long-range correlations and probability analysis may offer to the study of students' interaction patterns. Future work will also focus on the order in which students interact with features and how much time students spend on each feature. Examining time and order will give us a better understanding of students' patterns of interactions and how these patterns may evolve across time.

5. ACKNOWLEDGEMENTS

This research was supported in part by the Institute for Educational Sciences (IES R305G020018-02; R305G040046, R305A080589) and National Science Foundation (NSF REC0241144; IIS-0735682). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the IES or NSF

6. REFERENCES

[1] Jackson, G. T., Boonthum, C., and McNamara, D. S. 2009. iSTART-ME: Situating extended learning within a game-based environment. In *Proceedings of the Workshop on Intelligent Educational Games at the 14th Annual Conference on Artificial Intelligence in Education*, (Brighton, UK, July 06 -10, 2009). AIED, IOS Press, 59-68.

[2] Malone, T., and Lepper, M. 1987. Making learning fun: a taxonomy of intrinsic motivations for learning. In *Aptitude, learning, and instruction*, R. Snow and M. Fair, Eds. Vol. 3. Cognitive and affective process analyses. Erlbaum, Hillsdale, NJ, 223-253.

[3] Rai, D., and Beck, J. 2012. Math learning environment with game-like elements: An experimental framework. *International Journal of Game Based Learning*, 2, (2012), 90- 110.

[4] Snow, E. L., Jackson, G. T., Varner, L. K., and McNamara, D. S. In Press. The impact of performance orientation on students' interactions and achievements in an ITS. *In Proceedings of the 26th Florida Artificial Intelligence Research Society Conference* (St. Petersburg, FL, May 21 – May 25, 2013). FLAIRS-26. AAAI Press.

[5] Rowe, J., McQuiggan, S., Robison, J., and Lester, J. 2009. Off-task behavior in narrative-centered learning environments. *In Proceedings of the Workshop on Intelligent Educational Games at the 14th Annual Conference on Artificial Intelligence in Education*, (Brighton, UK, July 06 -10, 2009). AIED, IOS Press, 99-106.

[6] Benhamou, S., and Bovet, P. 1989. How animals use their environment: a new look at kinesis. *Animal Behavior*, 38, (1989) 375-383.

[7] Nelson, C. R., and Plosser, C. R. 1982. Trends and random walks in macroeconmic time series: some evidence and implications. *Journal of Monetary Economics*, 10, (1982) 139-162.

[8] Collins, J. J., and De Luca, C. J. 1994. Random walking during quiet standing. *Physical Review Letters*, 73, (1994) 764-767.

[9] Arneodo, M., Arvidson, A., Badełek, B., Ballintijn, M., Baum, G., Beaufays, J., and Prytz, K. 1995. Measurement of the proton and the deuteron structure functions, *Physics Letters B*, 364, (1995) 107-115.

[10] Lobry, J. R. 1996. Asymmetric substitution patterns in the two DNA strands of bacteria. *Molecular Biological Evolution*, 13, (1996) 660–665.

[11] McNamara, D. S., O'Reilly, T., Rowe, M., Boonthum, C., and Levinstein, I. B. 2007. iSTART: A web-based tutor that teaches self-explanation and metacognitive reading strategies. In *Reading comprehension strategies: Theories, interventions, and technologies*, D. S. McNamara, Ed. Erlbaum. Mahwah, NJ, 397-420.

[12] Jackson, G. T., and McNamara, D. S. In Press. Motivation and performance in a game- based intelligent tutoring system. *Journal of Educational Psychology*. (in press).

[13] McNamara, D.S., Boonthum, C., Levinstein, I.B., and Millis, K. 2007. Evaluating self-explanations in iSTART: Comparing word-based and LSA algorithms. In *Handbook of Latent Semantic Analysis,* T. Landauer, D. S. McNamara, S. Dennis, and W. Kintsch, Eds. Erlbaum. Mahwah, NJ, 227-241.

[14] Jackson, G.T., Guess, R.H., and McNamara, D.S. 2009. Assessing cognitively complex strategy use in an untrained domain. In *Proceedings of the 31st Annual Meeting of the Cognitive Science Society* (Amsterdam, The Netherlands, July 29 – August 01, 2009). Cognitive Science Society, Austin, TX, 2164-2169.

[15] MacGinitie, W., and MacGinitie, R. 1989.*Gates MacGinitie reading tests*. Riverside. Chicago, IL.

[16] Cohen, J. 1988. *Statistical power analysis for the behavioral sciences*. Lawrence Erlbaum. Hillsdale, NJ.

From “Events” to “Activities”: Creating Abstraction Techniques for Mining Students’ Model-Based Inquiry Processes

Vilaythong Southavilay*, Lina Markauskaite**, Michael J. Jacobson**

*School of Information Technologies
University of Sydney
vstoto@it.usyd.edu.au

**Centre for Research on Computer Supported Collaborative Learning and Cognition
University of Sydney
lina.markauskaite@sydney.edu.au
michael.jacobson@sydney.edu.au

ABSTRACT

In this paper, we present a technique that we have developed to transform sequences of technical events into more abstract actions and semantic activities. The sequences of more abstract units are then used for discovering patterns of students’ interaction with computer models using heuristic miner. Our proposed approach automatically segments sequences of technical events that occurred during model runs and pauses and, on the basis of the nature of technical events that occurred during model runs and pauses, clusters them into actions. Then, using heuristic rules, it classifies actions into activities. We demonstrate the usefulness of our multilevel abstraction for extracting and exploring characteristic patterns of students’ interaction with computer models. Our study shows that each abstraction level could help to identify distinct characteristics of students’ interaction.

Keywords

Process mining, data pre-processing, multilevel data analysis, model-based learning.

1. INTRODUCTION

It has been acknowledged that model-based inquiry could be a very effective pedagogical approach for teaching and learning complex scientific knowledge. Computer models could help students to engage in first-hand scientific investigations of complex social and natural phenomena and construct deep, authentic understanding of many complex processes, such as climate change [3, 4]. However, a number of studies that investigated model-based learning have found that not all learners succeed achieving desired learning outcomes [1, 8, 9]. They suggested that students’ failure and success to learn from computer models may be related to the differences in how students interact with computer models. For example, studies demonstrated that some students, when they explore computer models, systematically change one parameter after parameter; in contrast, other students approach tasks in more haphazard ways and change different model settings simultaneously [6]. While the former students usually succeed completing given inquiry tasks, the latter students tend to be less successful demonstrating desired learning outcomes. However, methods for exploring students’ model-based inquiry processes have been complicated, usually based on human coding, thus hard to implement in computer systems. Only recently researchers have started to explore the possibilities to extract students’ inquiry characteristics and patterns automatically from the log files [2, 7]. These techniques could help to detect productive and unproductive students’ behaviours and scaffold students’ inquiry automatically.

A traditional approach in exploring learning processes automatically is to use event logs of students’ interaction with computer software as an input to process mining algorithms. However, processes of students’ interaction with computer models tend to be very flexible, unstructured and composed from large numbers of fine-grained technical events. Consequentially, technical events may be too far remote from students’ intended purposeful actions and the identified patterns could be hard to understand and use.

Our goal is to develop a method for identifying semantically more abstract and meaningful students’ actions and activities from lower level technical events recorded in the logs. The sequences of more abstract units then could be used to investigate students’ model-based inquiry patterns. In this paper, we present a multilevel data preprocessing approach that we have created and used in combination with process mining for investigating students’ model-based inquiry strategies. Using real data, we illustrate that by identifying more abstract semantic units and examining their sequences we can gain additional insights into how students interact with NetLogo computer models.

The rest of this paper is organised as follows. Section 2 reviews related work and introduces our approach. Section 3 illustrates the capability of our approach using real event log data. Section 4 presents conclusions.

2. RELATED WORK AND APPROACH

There have been a number of studies that focused on analyzing data logs of less-structured processes. However, the process models extracted from raw data logs using traditional process mining techniques have been usually difficult to interpret. One of recently proposed techniques to deal with this issue is abstraction. Particularly, Bose and Aalst [1] explored a way that identifies the looping constructs in logs and replaces the repeated occurrences of the loops by more abstract entities (aka. activities). They also proposed a technique for discovering sub-processes or common functionalities in the traces and replacing them with more abstract entities. Our technique adopts Bose and Aalst [1] approach and focuses on how event logs could be semantically transformed into more abstract action and activity sequences. However, it interprets looping events and repeated occurrences of events differently from Bose and Aalst [1]. Our technique segments technical event-sequences into “bags of events” and then uses a heuristic-based classifier to automatically identify activities.

2.1 Study Context and Data

Data for this study came from a larger design-based project that investigates how school students learn scientific knowledge about climate change [4]. In total, 90 students from a high girls' school learnt about climate change by exploring computer models. Students completed inquiry tasks in dyads. In this paper, we conducted the analysis of dyads' interactions with the Carbon Cycle model (Figure 1). In total, we recorded 21 event log.

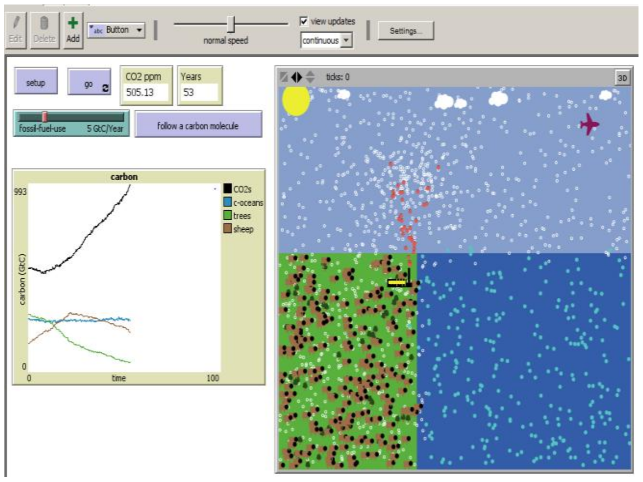


Figure 1. A carbon cycle model

2.2 Pre-processing Logs with Abstractions

Our data pre-processing included three steps. During the first two steps, we abstracted action-sequences from the event-sequences, and in the third step, we further abstracted activity-sequences from action-sequences. The recorded log files comprised process instances. A process instance is a sequence of events in which each event represents an interaction with a model (e.g., a click of "Setup" button).

In the first step, we identified five main categories of technical events: two types of basic model control; tracking of different model elements; change of modelling speed; and change of model parameters (Table 1). In our analysis we regarded dichotomous events as two discrete events (e.g., "Go" was regarded as "Start" and "Stop" of the simulation), whereas all events with continuous parameters we regarded as singular events without parameters (e.g., "Speed" was considered as change of speed without taking into account the direction of change and size). Thus, the pre-processed log files were composed from the sequences made up from seven singular technical events. In this step, process instances (sequences of events) were segmented using "Start," "Stop" and "Setup" events as the main semantic delimiters. As a result, each process instance consisted of non-overlapping segmented sequences of events. The sequences that began with "Start" and ended with "Stop" identified those events that students made during model simulation, including their implementations of planned modelling experiments and more spontaneous interactions with the model. The sequences that began with "Stop" and ended with "Start" distinguished those events that students made when the simulation was paused. We called these two types of sequences "runs" and "pauses," respectively. The execution time of each segmented sequence was computed as a time difference between the end and start of the segment. "Setup" terminated the execution of the model. Thus, semantically, "Setup" events that occurred during runs were similar the sequences of three events "Stop-Setup-Start" and was interpreted as a pause with the execution time of 0 seconds.

In the second step, we identified actions by classifying segmented event-sequences identified in the first step. Each event-sequence was considered to be "a bag of events". In order words, the order of events, except "Start" and "Stop," in each event-sequence was not considered; and all events were indexed using a binary digit. The occurrence of an event was assigned value 1, if the event happened in the sequence at least once; otherwise, the occurrence was assigned value 0 (i.e. the event did not occur). Based on this binary presentation, we clustered all segmented event-sequences into actions. Actions were represented by a string of five digits. Each event, but "Start" and "Stop," was assigned a unique position (see Table 1). The absence or presence of the event within each action was identified by assigning value "0" or "1" for the digit representing that event in the string. For example, simulation speed was represented by the second digit and fossil fuel use was represented by the third digit; thus the sequence "start-01100-stop" represented a modelling action during which a dyad changed the simulation speed and fossil fuel use while the simulation was running. Therefore, each action had a distinct semantic and represented the nature of students' interaction with the model during "runs" or "pauses."

Table 1. Types of interactions with the carbon cycle model

Buttons in the model	Category [type][1]	Technical events [position]	Examples of actions
Go [On/Off]	Model control [Dichotomous]	Start [S] Stop [E]	S-00000-E E-00000-S
Setup [nil]	Model control [Singular]	Setup [1]	S-10000-E
Follow a CO_2 molecule [On/Off]	Tracking [Dichotomous]	Follow molecule – On [4] Follow molecule – Off [5]	S-00010-E S-00001-E
Change of speed [value]	Speed change [Continuous]	Change of speed [2]	S-01000-E
Change fossil fuel use [value]	Parameter change [Continuous]	Change fossil fuel use [3]	S-00100-E

In the third step, activities were created by classifying actions. We identified activities using three features: action types, event types, and duration. Action type specified whether an activity was a "pause" or a "run". Event type specified the nature of events that occurred during each activity and included four heuristic categories: control, interaction, configuration and combined activities. *Control* activities included simple runs, pauses, and reset. They indicated those periods when students controlled the model by simply running, and pausing the simulation or resetting the simulation to the initial conditions. *Interaction* activities indicated those periods when students controlled their interaction with the model by adjusting simulation settings, such as making visible and tracking the behaviour of an individual CO_2 molecule

[1]Type - is the type of a variable in the log file corresponding different types of behaviour: *Singular events* have one discrete value (no parameter); *Dichotomous events* have two possible values (*On* or *Off*); *Continuous events* have a range of values (continuous parameters). **Position – identifies the position of the digit that represents each technical event in the action.

and changing simulation speed. *Configuration* activities included those periods when students changed model variables that impact the modelled phenomenon, such as the rate of CO_2 emissions. *Combined* activities included the combination of model control, interaction and configuration. Interaction and configuration activities also involved model control. To specify the time feature, we used action durations: value 1 indicated that the execution time was longer than 0 seconds; value 0 indicated that the length of an action was 0 seconds. This classification resulted in nine kinds activities (Table 2) from which we created activity logs. The three logs - the event-sequences, action-sequences, and activity-sequences – then were used as inputs to process mining.

Table 2. Classification of actions into activities

Actions[2]	Sec	Activity
S-00000-E	>0	Control_Run
S-010XX-E | S-0X01X-E | S-0X0X1-E	>0	Inter_Run
S-00100-E	>0	Conf_Run
S-011XX-E | S-0X11X-E | S-0X1X1-E	>0	Comb_Run
E-10000-S	=0	Reset_Run
E-X0000-S	>0	Control_Pause
E-X10XX-S | E-XX01X-S | E-XX0X1-S	>0	Inter_Pause
E-X0100-S	>0	Conf_Pause
E-X11XX-S | E-XX11X-S | E-XX1X1-S	>0	Comb_Pause

3. RESULTS

There were 21 event logs that, in total, included 2582 technical events. The event logs were segmented into 1520 event-sequences. Each event-sequence was transformed into "bags of events" with binary indexing. All transformed event-sequences were then clustered into action-sequences and activity-sequences.

In order to demonstrate the capability of our data preparation method, we present and compare process patterns of one dyad created using data at three different levels of abstraction - events, actions and activities. We generated three causal dependency diagrams (nets) from the pre-processed data using the Heuristic Miner algorithm with the default parameters [9, 10]. The analysis and comparison of these three causal nets shows that each of the process models depicts distinct features of students' interaction with the simulation (Figures 2-4).

The event process model shows that "start" and "stop" dominated in students' interaction with the model and each event was recorded in the log 33 times (Figure 2). "Setup" event during which students reset parameters appeared also quite often (20 times); whereas "fossil-fuel-use" and "speed" events were executed only 14 and 11 times, respectively. The model shows that "stop" event plays quite distinct role in the process pattern; and two other events - "start" and "fossil-fuel-use" - have high dependencies on "start" event (0.9). This indicates that the dyad usually took these two actions after stopping the model. "Start" and "stop" events are mutually co-dependent. It means that these students often started the simulation and then stopped it without any further interaction and/or configuration. Nevertheless, "start" and "stop" events form a part of a larger three-event loop of "stop", "fossil fuel use" and "start" with dependencies 0.9 of these three events on each other. It indicates that these students usually configured the model after stopping the simulation. "Start," "speed" and "stop" events form another loop, where "speed" event has a dependency on "start" event (0.5) and "stop" event has a dependency on "speed" (0.7). The dependencies in this event loop, however, are lower, which indicates that the change of speed did not necessary follow the start of the simulation or preceded the stop. Indeed, the extracted process net shows that "speed" also depends on "fossil fuel use," which suggests that the students sometimes changed speed just after changing "fossil fuel use" parameter, thus configured modeling and interaction parameters together. Further, "setup" event has a noticeable dependency on "stop" (0.7), suggesting that the dyad usually reset the simulation after stopping it. However, there is a loop from "setup" event to "setup" indicating that students, for some reasons, sometimes pressed "setup" button multiple times.

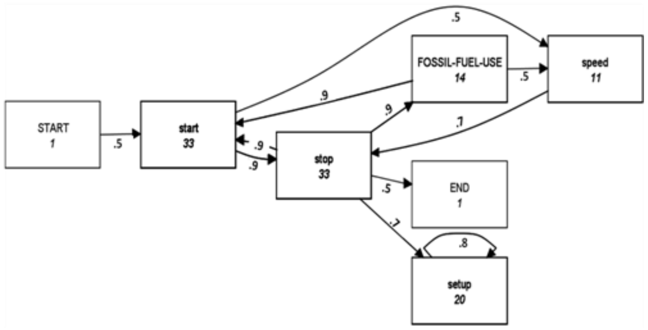


Figure 2. Process model mined from event-sequence

The action process net reveals additional characteristics of students' modeling behavior (Figure 3). The model distinguishes characteristic features of each run and each pause. Therefore, we can see what the dyad did during and in between the execution of the simulation. As Figure 3 shows, the students took only two kinds of actions when the simulation was running. Firstly, the students simply ran the simulation without any interaction and configuration (i.e. S-00000-E). This action dominated in their process model (27 times). Secondly, the students sometimes changed the simulation speed while it was running (i.e. S-01000-E), but this action appeared less frequently (8 times). In contrast, the students took a large variety of different actions between the executions of the simulation. Particularly, there are 7 types of "pause" actions. Among the six actions connected to the simple simulation run, 5 of them are co-dependent. That is, there are both out-going and in-coming dependency arrows to these five actions from the simple simulation run. This model shows that more complex interactions during the runs usually tended to follow more complex model configurations during the pauses, whereas simple interactions during the runs tended to follow more simple model reset or pause actions. It is important to note, that this tendency was not depicted in the process model that was based on the technical events (Figure 2).

[2]X represents either 1 or 0. The position in the sequences of five digits indicates that a certain event occurred during the activity: 1) setup; 2) speed; 3) fossil fuel use; 4) track a CO_2 molecule - on; 5) track a CO_2 molecule - off. Sequences that begin with "start" (S) indicate that action was performed while the model was running; whereas sequences that begin with "stop" (E) indicate actions performed while the simulation was paused.

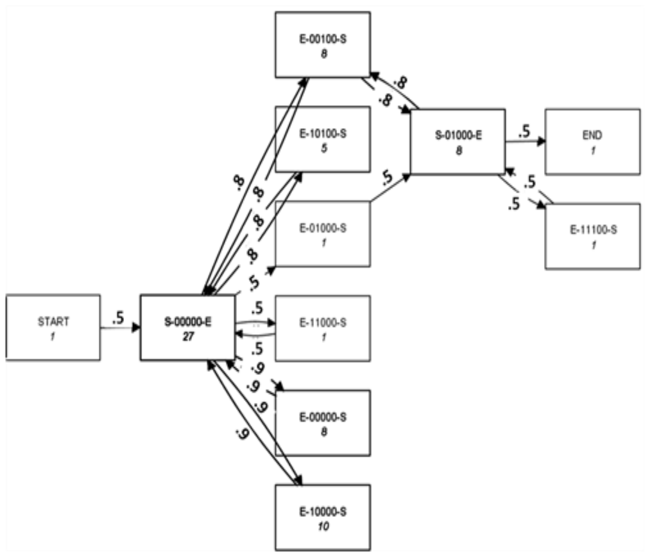


Figure 3. Process model mined from action-sequence

The activity process net provides additional insight into the students' modeling behavior (Figure 4). As we can see, students' activity pattern involves two kinds of model run activities: simple control of the model (27 activities) and interaction with the model while the simulation is running (8 activities). These activities are similar to the actions depicted in the action process model (Figure 3). However, the activity pattern depicts that students' activities during the pauses also form two dominant groups: pauses that involve simple control (16 activities) and pauses that involve configuration of the model (13 activities). The control pauses are strongly co-dependent with just one activity, which is the control of the simulation during model runs (dependency 0.9). In contrast, the pauses that involve model configuration are codependent with both types of simulation runs: simple control and interaction with the simulation while the model was running (dependencies 0.9). This pattern shows that students' active configuration during pauses was followed by a mix of passive observation and more proactive exploration during runs, whereas passive behavior during pauses was always followed by their passive behavior during runs.

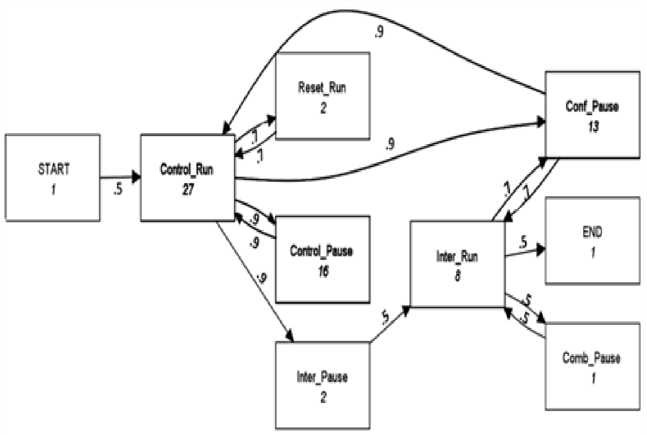


Figure 4. Process model mined from activity-sequence

4. DISCUSSION AND CONCLUSION

The technique described here presents our work in progress. In conclusion, we have created a technique to identify actions and activities based on the combination of events of students' interaction with computer models. We also illustrated that using larger semantic units and examining their sequences we could gain additional insights into how students interact with computer models.

The result presented here demonstrates the value of our technique in extracting patterns of students' interaction with computer models for individual pairs of students. Although a pattern of students' interaction can be extracted for a particular pair of students, there are different patterns for different pairs. In addition, we did not take into account students' performance and learning gain. In future, we are planning incorporate students' performance and learning gain into our study and process analysis.

5. ACKNOWLEDGMENTS

The research reported in this paper was funded by an Australia Research Council Linkage Grant No. LP100100594. We thank Paul Stokes, Nick Kelly, Kashmira Dave and the teachers and students for their invaluable assistance in this study.

6. REFERENCES

[1] Bose, R.P.J.C., and Aalst, W.M.P.V.D. 2009. Abstractions in process mining: A taxonomy of patterns. In *Proceedings of Bussiness Process Management,* 159-175.

[2] Buckley, B.C., Gobert, J.D., Horwitz, P., and O'dwyer, L.M. 2010. Looking inside the black box: Assessing model-based learning and inquiry in Biologica™, *International Journal of Learning Technology,* 5(2), 166-190.

[3] Goldstone, R.L., and Wilensky, U. 2008. Promoting transfer by grounding complex systems principles, *Journal of the Learning Sciences,* 17(4), 465-516.

[4] Kelly, N., Jacobson, M., Markauskaite, L., and Southavilay, V. 2012. Agent-based computer models for learning about climate change and process analysis techniques. In *Proceedings of 10th International Conference of the Learning Sciences, International Society of the Learning Sciences*, Sydney, Australia, 25-32.

[5] Levy, S.T., and Wilensky, U. 2010. Mining students' inquiry actions for understanding of complex systems, *Computers & Education*, 56(3), 556-573.

[6] Mcelhaney, K.W., and Linn, M.C. 2011. Investigations of a complex, realistic task: Intentional, unsystematic, and exhaustive experimenters, *Research in Science Teaching,* 48(7), 745-770.

[7] Pedro, M. S., 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 model. *User-Adapt. Interact.*, 23(1), 1-39.

[8] Thompson, K., and Reimann, P. 2010. Patterns of use of an agent-based model and a system dynamics model: The application of patterns of use and the impacts on learning outcomes, *Computers & Education*, 54(2), 392-403.

[9] Weijters, A.J.M.M., and Ribeiro, J.T.S. 2011. Flexible heuristics miner (FHM). In *CIDM'11,* Eindhoven University of Technology, Eindhoven, 310-317.

[10] Weijters, A.J.M.M., van der Aalst, W.M.P., and Medeiros, A.K.A.D. 2006. Process mining with the heuristics miner algorithm, *Technology,* 166, 1-3

A Comparison of Model Selection Metrics in DataShop

John C. Stamper
Human-Computer Interaction Institute
Carnegie Mellon University
Pittsburgh, PA
john@stamper.org

Kenneth R. Koedinger
Human-Computer Interaction Institute
Carnegie Mellon University
Pittsburgh, PA
koedinger@cmu.edu

Elizabeth A. McLaughlin
Human-Computer Interaction Institute
Carnegie Mellon University
Pittsburgh, PA
mimim@cs.cmu.edu

ABSTRACT

Variations of cognitive models drive many instructional decisions that intelligent tutoring systems currently make. A better knowledge component model will yield better instruction, but how do we identify better cognitive models? One answer has been to create a latent variable version of a cognitive model or a so-called knowledge component (KC) model, then compare different models by how well they predict student performance data. In this research we analyze 1,943 proposed KC models that exist in DataShop (http://pslcdatashop.org) and compare and contrast the different metrics used to measure the quality of predictive fit to the data. All these metrics are designed to avoid over-fitting to the data, including AIC, BIC, and cross validation. We find that AIC is the metric most consistent with all the others and corresponds better with cross validation results than BIC.

Keywords

Model Selection, AIC, BIC, Cross-validation, KC Modeling.

1. INTRODUCTION

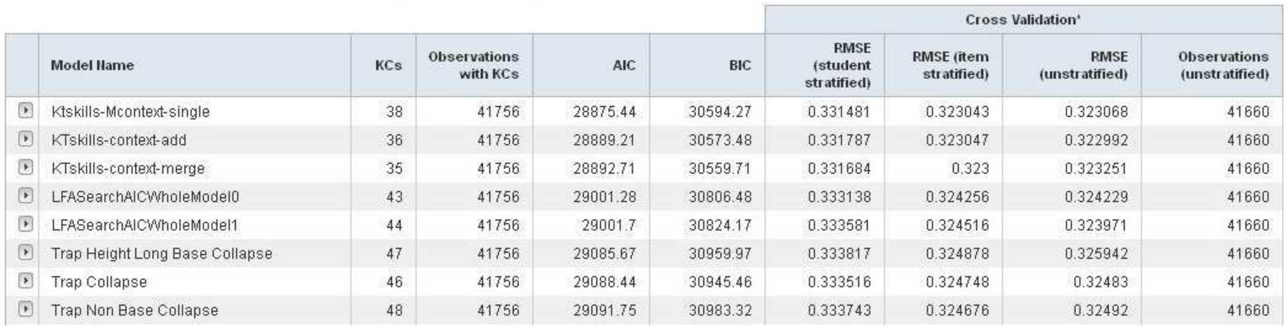
An important area of Educational Data Mining (EDM) is the building and improvement of models of student knowledge. Creating good models are important in the design of adaptive feedback, assessment of student knowledge, and predicting student outcomes [9]. A correct model of student knowledge is consistent with student behavior, such that, it predicts task difficulty and transfer between prior opportunities to practice and learn (via positive or negative feedback and next-step hints) and future opportunities to demonstrate learning (by correct performance). These models are evaluated by how well they predict the student performance on actual student data. To prevent selecting models that overfit the data (and would thus not work well in new contexts), prediction fit is measured using a number of techniques including cross validation, the Akaike information criterion (AIC), and the Bayesian information criterion (BIC). Cross validation is the gold standard for evaluating prediction fit and avoiding over-fitting, but it can take substantial time to run making it undesirable for searching for new models. AIC and BIC are metrics that can be calculated quickly, which makes them desirable when comparing a large number of proposed models, but how adequate are they and which one is better at anticipating cross validation results? This research explores comparisons of AIC, BIC, and various cross validations that are available in DataShop.

DataShop is the world's largest open data repository of transactional educational data collected from a wide variety of online learning environments [10]. The data is fine-grained, with student actions recorded roughly every 10 seconds on average, and it is longitudinal, spanning semester or yearlong courses. As of May 2013, over 400 datasets are stored including over 100 million student actions, which equates to over 250,000 student hours of data. Most student actions are "coded" meaning they are not only graded as correct or incorrect, but are categorized in terms of the hypothesized skills or knowledge components (KCs) needed to perform that action. DataShop stores a widespread selection of educational data from assorted technologies, domains and researchers. STEM subjects are well represented as are languages such as Chinese, English and French. There are also accessible datasets in miscellaneous content areas like reading, psychology, logic and handwriting. The acquisition of student log-data comes from a multitude of sources including intelligent tutors, online-courses and internet games and simulations. The collection methodologies include random controlled experiments, longitudinal studies, and anonymous on-line game playing.

Given the accessibility of data and diversity of applications stored in DataShop's repository, we were interested in exploring the metrics commonly used for model selection and prediction (i.e., AIC, BIC and Cross-validation). In particular, we examined correlations and rank order correlations between the various metrics to determine if one metric stands apart as a better predictor. Next, we examined best model selections for AIC and BIC and how they compared to cross validation best model selection.

A KC is defined as a piece of knowledge that can be applied to solve a specific task. Practically, KCs can be considered generalizations of skills or concepts that form the basis of a cognitive model of student knowledge. A typical step in a problem that a student will solve will include one or more KCs that describe the knowledge that the student is applying. A mapping of KCs to problem steps in a set of instruction forms a KC Model. Multiple mappings can be fit to the same set of student instruction based on the granularity of the KCs that make up each model. Figure 1 shows a screen shot from DataShop listing the KC Models and their evaluation metrics for a dataset called "Cog Model Discovery Experiment Spring 2010."

A KC model can be used to track individual student knowledge or predict student responses based on a statistical representation of the KC Model. In DataShop, the model used to evaluate student learning is called the Additive Factors Model (AFM) [3; 11]. AFM is an extension of item response theory that incorporates a growth or learning term [cf.,6]. AFM is shown in Figure 2. The discrete portion of the student model is represented by q_{jk}, the so-called "Q matrix" [13], which maps hypothesized difficulty or learning factors (the knowledge components or skills) to steps in problems. These factors are hypothesized causes for difficulty (β_k) or for learning improvement as students practice (γ_k). AFM gives a probability that a student i will get a problem step j correct based on the student's baseline proficiency (θ_i), the baseline difficulty (β_k) of the required KCs (q_{jk}), and the improvement (γ_k) in those KCs as the student gets practice opportunities (T_{ik}).



	Model Name	KCs	Observations with KCs	AIC	BIC	Cross Validation¹ RMSE (student stratified)	RMSE (item stratified)	RMSE (unstratified)	Observations (unstratified)
▸	Ktskills-Mcontext-single	38	41756	28875.44	30594.27	0.331481	0.323043	0.323068	41660
▸	KTskills-context-add	36	41756	28889.21	30573.48	0.331787	0.323047	0.322992	41660
▸	KTskills-context-merge	35	41756	28892.71	30559.71	0.331684	0.323	0.323251	41660
▸	LFASearchAICWholeModel0	43	41756	29001.28	30806.48	0.333138	0.324256	0.324229	41660
▸	LFASearchAICWholeModel1	44	41756	29001.7	30824.17	0.333581	0.324516	0.323971	41660
▸	Trap Height Long Base Collapse	47	41756	29085.67	30959.97	0.333817	0.324878	0.325942	41660
▸	Trap Collapse	46	41756	29088.44	30945.46	0.333516	0.324748	0.32483	41660
▸	Trap Non Base Collapse	48	41756	29091.75	30983.32	0.333743	0.324676	0.32492	41660

Figure 1. Screenshot of the KC Models page in DataShop (http://pslcdatashop.org) for the dataset Cog Model Discovery Experiment Spring 2010. Here we can see named models with a different number of KCs in each. Note that all the models with the same number of observations with KCs (41,756 for example) are comparable with each other. DataShop also allows for the user to select the metric on which to sort the models. In this case, the models are sorted by AIC where a lower value is better.

$$\ln\left(\frac{p_{ij}}{1-p_{ij}}\right) = \theta_i + \sum_{k=1}^{K} q_{jk}\beta_k + \sum_{k=1}^{K} q_{jk}\gamma_k T_{ik}$$

Figure 2. In the Additive Factors Model (AFM), the probability student i gets step j correct (p_{ij}) is proportional to the overall proficiency of student i (θ_i) plus for each factor or knowledge component k present for this step j (indicated by q_{jk}), add the base difficulty of that factor (β_k) and the product of the number of practice opportunities this student (i) has had to learn this factor (T_{ik}) and the amount gained for each opportunity (γ_k).

The AFM model can be used to evaluate and predict learning, which can be visualized in DataShop with the use of learning curves. Fig 3 shows a learning curve for a KC called "Find circle circumference." The red line represents the actual student error rate from data collected in the dataset over each opportunity students have to apply the KC. The blue line represents the predicted model derived through AFM. DataShop allows for visual inspection of the KCs and their predicted fit with AFM, which can be used to help identify potential improvements in the KC model when the data and AFM curves do not match [12].

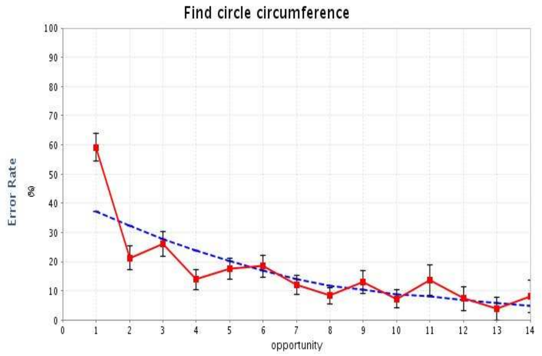


Figure 3. Learning Curve visualization from DataShop showing the KC "Find circle circumference". The y-axis is the error rate and the x-axis is each opportunity students have to apply the KC. The red is actual data and blue line is the predicted value.

When a potential improvement is found and a new model is proposed, it can be imported into DataShop and the system will automatically evaluate the new model against five metrics: AIC, BIC, Student Stratified Cross Validation (SSCV), Item Stratified Cross Validation (ISCV), and Non-Stratified Cross Validation (NSCV). Using these metrics the researcher can make a judgment as to whether the potential model leads to a better fit on the data.

When time is not an issue, cross validation is considered by most to be the best way to score models, but there is no consensus on how the cross validation should be done for educational transaction data. DataShop provides three cross validation measures that are each 10 fold and provides a value for the root mean squared error (RMSE). One measure stratifies the data by student, another by item, and the third is not stratified. While cross validation is considered the best method to score models, it is more time consuming and computationally expensive for large datasets than AIC or BIC. For this reason, when comparing many models we use AIC or BIC to score the models.

One active research area where many models are compared and evaluated against each other is the automated search for improved models. Using the AFM model and datasets in DataShop we have previously implemented an automated search algorithm, Learning Factors Analysis (LFA), for discovering better cognitive models [8]. This algorithm has been successfully applied to DataShop datasets and succeeded in improving existing models. Figure 1 includes two models that were automatically generated and are named "LFASearch…" The LFA search algorithm uses existing KC Models to complete a directed search, which results in labeled models that are easily interpretable by researchers.

AIC and BIC are measures for the goodness of predictive fit of a statistical model. They extend the log-likelihood measure of fit by penalizing less parsimonious models. Unlike the RMSE calculation from cross validation, the values of AIC and BIC have no meaning for an individual model, and are only useful when comparing alternative models built on the same dataset. Within DataShop, this means that models must have the same number of observations tagged with KCs to be comparable. DataShop also has a Model Values page under the Learning Curve tool that has more detailed information on the model metrics (AIC, BIC, and the cross validations), and the inputs used to calculate them (log likelihood, number of parameters, etc.). AIC is a metric for model comparison that trades off the complexity of the estimated model against how well the model fits the data [1]. In this way, it penalizes the model based on its complexity (the number of parameters). The equation for calculating AIC is ***AIC*= 2k – 2 ln(L)**, where k is the number of parameters and L is the likelihood. The equation for BIC is ***BIC*=k ln(n) – 2 ln(L)**, where n is the number of observations, k is the number of parameters, and L is the likelihood. BIC is similar to AIC, but BIC penalizes free parameters more strongly than AIC as can be seen by the formulas and noting that the coefficient of the number of parameters (k) is much larger for BIC (ln(n) for n observations)

Table 1. AIC and BIC correlations against each other and Cross-validation

KC set name	# students	# models	# obs	AIC-BIC correl	AIC-correlation			BIC-correlation		
					SSCV	ISCV	NSCV	SSCV	ISCV	NSCV
Assistments Math 2008-2009 Symb-DFA (302 Students)	302	31	8181	0.666	0.936	0.994	0.989	0.438	0.662	0.630
Assistments Math 2008-2009 Symb-DFA (302 Students)	302	23	4957	0.986	0.956	0.977	0.973	0.961	0.961	0.961
OLI Engineering Statics - Fall 2011 - CMU (74 students)	74	4	71805	0.973	0.967	1.000	0.999	0.882	0.976	0.979
OLI Engineering Statics - Fall 2011 - CMU (74 students)	74	5	37423	0.983	0.989	0.650	0.996	0.999	0.568	0.972
IWT Self-Explanation Study 2 (Fall 2009) (tutors only)	99	13	7094	0.200	0.822	0.945	0.916	0.538	0.198	0.064

than for AIC (2) for any non-trivially sized data set. In general, this means that BIC favors models with less parameters (again more strongly the AIC), and converges to the "true" or correct model [1], however, this does not mean that for BIC to be useful that the "true" model must exist in the set of possible models [2]. Both reduce the chance of over-fitting the data by penalizing for increasing the number of parameters in the model. They are much faster to compute than cross validation and are believed to reasonably predict the results of cross validation, though no systematic investigation of that has been performed, at least, for the kinds of EDM models investigated here. Given that AIC is more lenient, one might suspect it would be more susceptible to favoring models that over-fitted the data. On the other hand, BIC might over penalize more complex models that indeed do capture true variability in the data. Many of the previous efforts to evaluate knowledge component models in EDM have used BIC as the evaluation criteria including Learning Factors Analysis (LFA) [3], Performance Factors Analysis (PFA) [11], and Instructional Factors Analysis (IFA) [4].

2. DATA AND METHOD

DataShop has grown to include almost 400 datasets as of February 2013. One of the fundamental features available in DataShop is the ability to fit different KC Models to a dataset. There are a number of ways KC Models can be generated with DataShop.

1) KC Models can be imported with log data of an initial dataset.
2) KC Models can be exported, modified, and re-imported through DataShop's intuitive user interface (Examples of this can be seen in the DataShop tutorial channel on Youtube [5]).
3) KC Models can be automatically generated by automated search algorithms such as LFA Search[8].
4) Every imported dataset automatically gets 2 models generated by DataShop- the Unique Step Model, which includes a KC for every step, and the Single KC model, which applies the same KC to every step.

Currently, there are 1,943 proposed KC Models in DataShop that were used for this analysis. Two conditions were established for a dataset to be included in the analysis: (1) three or more models with an equal number of observations were required and (2) the number of observations had to be greater than 800. We found 50 datasets within DataShop that met the conditions and 12 of them had more than one grouping of models (10 had 2 sets; 1 had 3 sets and 1 had 4 sets) for a total of 65 comparable KC sets. In addition to the aforementioned diversity of content and technology, the 65 KC sets have a broad range of the number of parameters (9 to 654), models (3 to 48), students (7 to 510), knowledge components (1 to 287), and observations (884 to 95,512). Such variation provides a rich environment for a deep analysis into what might be the best measure for model selection. As shown in Figure 1, DataShop provides a leaderboard of commonly used metrics across models within a dataset. We examined the correlations and rank order correlations for AIC, BIC, and Cross-validation across the 65 KC sets. We chose to report rank order correlations in addition to correlations because it is less sensitive to outliers that may excessively inflate (or, less frequently, lower) a correlation.

3. RESULTS AND DISCUSSION

After running the correlations between the metrics, we found that for the majority of KC sets (44 of 65), AIC and BIC do not agree on which model best fits the data. More importantly, AIC is overwhelmingly the better predictor when compared with cross validation best models (an average 94% match vs. BIC's 33%). To be more precise, of the 44 comparable KC sets, 41 of AIC best models match with SSCV best models vs. 13 for BIC, for ISCV - 41 AIC best models match vs. 14 for BIC, and for NSCV - 42 AIC best models match vs. 16 for BIC. It is noteworthy that the three AIC best models that do not match with at least one cross validation best model have a substantially lower average number of KCs (12) and number of observations (5,941) than the 41 models with a match (average of 53 KCs and 17,374 observations). This appears to be because the AIC implementation in DataShop does not take into account second order Akaike Information Criterion (AIC_C) which has an adjustment for smaller sample sizes in relation to number of parameters [1]. As an example, Table 1 shows a small subset of the 65 comparable KC sets illustrating a strong positive correlation between AIC and

Table 2. Correlations and rank order correlations across the five metrics provided in DataShop (AIC, BIC, SSCV,ISCV and NSCV).

	AIC-BIC	AIC-SSCV	AIC-ISCV	AIC-NSCV	BIC-SSCV	BIC-ISCV	BIC-NSCV	SSCV-ISCV	SSCV-NSCV	ISCV-NSCV
Correlations	0.574	0.824	0.891	0.890	0.522	0.464	0.446	0.812	0.777	0.919
Rank Corr.	0.532	0.817	0.852	0.847	0.478	0.403	0.420	0.760	0.735	0.868

each of the three cross validations regardless of whether the AIC-BIC correlation is strong (rows 2-4), weak (row 5) or average (row 1). In all but one instance, the AIC correlations with cross validation are better than BIC. Table 2 shows the average correlations and rank correlations between AIC, BIC, and the Cross-validations (as stated earlier, three types of ten-fold cross validations are reported in DataShop: student stratified cross validation (SSCV), item stratified cross validation (ISCV) and non-stratified cross validation (NSCV)). From these averages in Table 2, AIC and BIC have correlations with each other of just over 0.5, which makes sense since they often do not agree on the best fitting model. More importantly, AIC is a better predictor than BIC of all three kinds of cross validation. Interestingly, table 2 shows SSCV is better indicated by AIC than the other CV metrics.

Thus, on those grounds, it seems as though AIC is the best single measure. In general, AIC best models average more knowledge components (53 vs. 34) and more parameters (205 vs. 166) than BIC best models. It is not surprising, then, that there is a high level of disagreement between best model selections for AIC and BIC (68% do not match). When comparing the best models of AIC and BIC to the best models of all three types of cross validation, AIC again matches better than BIC (approximately 70% to 10%). This better matching of best models is another strong argument that AIC is a better metric for model selection.

4. CONCLUSION AND FUTURE WORK

Although cross validation is the gold-standard for model selection, it is not a reasonable metric to use for computationally expensive processes, such as inside the LFA search, as it is too time consuming. Efficiency concerns together with uncertainty about which is a better heuristic led us to a detailed comparison of AIC/BIC across datasets and many models. Our evidence points toward AIC as the better predictor of cross validation results.

A possible reason may follow from the fact that AIC favors greater complexity within models than BIC. While the KC models in DataShop are a good approximation of student cognitive processing, it is quite likely that they significantly under-represent the true complexity of student thinking. Thus, rather than the higher bias toward simplicity that is implicit in BIC, it may be that higher complexity is a better prior belief. The true (more rich and complex) cognitive model is most likely outside the space of models that we are searching within and AIC is claimed to be better than BIC in such circumstances [14]. On the one hand, it is a positive sign of maturity of the field of Educational Data Mining that we now have so many datasets and so many alternative KC models that a comparison like this one is possible. On the other hand, it is clear that more and better research is needed to better uncover the true complexity and richness of student thinking.

It is also important to note that AIC and BIC are not the only model selection metrics available, and in the future we hope to explore alternatives for possible inclusion in DataShop. Further, the only statistical model used in this analysis was AFM. While we expect that the results would be similar with other regression based statistical models (such as PFA or IFA), we have implemented a facility in DataShop to accept external analyses, and we plan to score additional statistical models across the DataShop metrics using the external analyses support.

We thank the DataShop team for providing custom reports of the KC Model metrics and Hui Cheng for help with the LFA Search.

5. REFERENCES

[1] Burnham, K., P., Anderson, D., R. Model selection and multimodel inference. a practical information-theoretic approach. New York: Springer; 2002.

[2] Burnham, K. P.; Anderson, D. R. (2004), "Multimodel inference: understanding AIC and BIC in Model Selection", Sociological Methods and Research 33: 261–304.

[3] Cen, H., Koedinger, K. R., & Junker, B. (2006). Learning Factors Analysis: A general method for cognitive model evaluation and improvement. In *Proceedings of the 8th International Conference on ITS*, 164-175. Springer-Verlag.

[4] Chi, M., Koedinger, K., Gordon, G., Jordan, P., and VanLehn, K. (2011). Instructional factors analysis: A cognitive model for multiple instructional interventions. In *Proceedings of the 4th International Conference on Educational Data Mining*. Eindhoven, the Netherlands

[5] DataShop Tutorial 2: Exploring an alternative skill model. In DataShop Youtube Channel. Retrieved 2/25/13. From www.youtube.com/user/datashoptutorials

[6] Draney, K.L., Pirolli, P., & Wilson, M. (1995). A measurement model for complex cognitive skill. In P. Nichols, S.F. Chipman, & R.L. Brennan (Eds.), *Cognitively diagnostic assessment* (pp. 103–126). Hillsdale: Erlbaum.

[7] Koedinger, K.R. & McLaughlin, E.A. (2010). Seeing language learning inside the math: Cognitive analysis yields transfer. In S. Ohlsson & R. Catrambone (Eds.), *Proceedings of the 32nd Conference of the Cognitive Science Society*. (pp. 471-476.) Austin, TX: Cognitive Science Society.

[8] Koedinger, K. R., McLaughlin, E. A., & Stamper, J. C. (2012). Automated Student Model Improvement. *Proceedings of the 5th International Conference on Educational Data Mining*. (pp. 17-24) Chania, Greece.

[9] Koedinger, K. R., Stamper, J. C., McLaughlin, E. A., & Nixon, T. (2013). Using data-driven discovery of better student models to improve student learning. (Submitted). *AIED 2013 - The 16th International Conference on AIED.*

[10] Koedinger, K.R., Baker, R.., Cunningham, K., Skogsholm, A., Leber, B., Stamper, J., (2011) A Data Repository for the EDM community: The PSLC DataShop. In *Handbook of Educational Data Mining*. CRC Press.

[11] Pavlik Jr., P.I., Cen, H., Koedinger, K.R.: Learning Factors Transfer Analysis: Using Learning Curve Analysis to Automatically Generate Domain Models. In *Proceedings of the the 2nd International Conference on Educational Data Mining*, Cordoba, Spain, pp. 121-130 (2009).

[12] Stamper, J. & Koedinger, K.R. (2011). Human-machine student model discovery and improvement using data. In J. Kay, S. Bull & G. Biswas (Eds.), *Proceedings of the 15th International Conference on AIED*, pp. 353-360. Springer.

[13] Tatsuoka, K.K. (1983) Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement*, 20, 345-354.

[14] Yang, Y. (2005). Can the strengths of AIC and BIC be shared? A conflict between model identification and regression estimation. Biometrika 92: 937–950.

Measuring the Moment of Learning with an Information-Theoretic Approach

Brett van de Sande
Arizona State University
PO Box 878809
Tempe, AZ 85287
bvds@asu.edu

ABSTRACT
There are various methods for determining the moment at which a student has learned a given skill. Using the Akaike information criterion (AIC), we introduce an approach for determining the probability that an individual student has learned a given skill at a particular problem-solving step. We then investigate how well this approach works when applied to student log data. Using log data from students using the Andes intelligent tutor system for an entire semester, we show that our method can detect statistically significant amounts of learning, when aggregated over skills or students. In the context of intelligent tutor systems, one can use this method to detect when students may have learned a skill and, from this information, infer the relative effectiveness of any help given to the student or of any behavior in which the student has engaged.

Keywords
data mining, information theory

1. INTRODUCTION
The traditional experimental paradigm for studying student learning is to use a pre-test and post-test combined with two or more experimental conditions. Pre-test and post-test scores can indicate *whether* learning has occurred, but not *when* it may have occurred. At best, one might infer when learning has occurred if an isolated change to the instructional materials or help-giving strategy results in better post-test performance. It is more difficult to infer whether a change in student behavior at some point has resulted in greater learning, since student behavior is largely uncontrolled and must be recorded in some way. In a laboratory setting, these issues can be addressed by careful experimental design, albeit with an accompanying loss of authenticity.

Moving from the laboratory to a more realistic setting, such as a classroom study, presents a challenge since there is necessarily an extended time between any pre-test and post-test. Heckler and Sayre [4] introduce an experimental technique where they administered a test to a different subgroup of students in a large physics class each week during the quarter, cycling through the entire class over the course of the quarter (a between-students longitudinal study). With a sufficiently large number of students (1694 students over five quarters), they were able to produce plots of student mastery of various skills as a function of time, and identify exactly which week(s) students learned a particular skill. However, the shortest time scale that one could imagine for this kind of approach (administering a test in a classroom setting) can, at best, be a day or so. Can we do better?

The use of an intelligent tutor systems (ITS) provides a way forward. In this case, student activity is analyzed and logged for each user interface element change, with a granularity of typically several 10s of seconds. Instead of relying on a distant pre-test or post-test, the experimenter can examine student (or tutor system) activity in the immediate vicinity of the event of interest.

Baker, Goldstein, and Heffernan [1] construct a model that predicts the probability that a student has learned a skill at a particular time based on the Bayesian Knowledge Tracing (BKT) algorithm [3]. BKT gives the probability that the student has mastered a skill at step j using the students performance on previous opportunities to apply that skill. The authors supplement the BKT result with information on student correctness for the two subsequent steps $j+1$ and $j+2$ and infer the probability that the student learned the skill at that step. Finally, they use their model to train a second machine-learned model that does not rely on future student behavior, so it could be run in real time as the student is working

We will address the same problem using an information-theoretic approach. Starting with the Akaike information criterion and a simple model of learning, we use a multi-model strategy to predict the probability that learning has occurred at a given step, and to predict how much learning has occurred. We apply our approach to student log data from an introductory physics course. We find that, for an individual student and skill, detection of learning has large uncertainties. However, if one aggregates over skills or students, then learning can be detected at the desired level of significance.

1.1 Correct/Incorrect steps

Our stated goal is to determine student learning for an individual student as they progress through a course. What observable quantities should be used to determine student mastery? One possible observable is "correct/incorrect steps," whether the student correctly applies a given skill at a particular problem-solving step without any preceding errors or hints. There are other observables that may give us clues on mastery: for instance, how much time a student takes to complete a step that involved a given skill. However, other such observables typically need some additional theoretical interpretation. *Exempli gratia*, What is the relation between time taken and mastery? Baker, Goldstein, and Heffernan [1] develop a model of learning based on a Hidden Markov model approach. They start with a set of 25 additional observables (for instance, "time to complete a step") and construct their model and use correct/incorrect steps (as defined above) to calibrate the additional observables and determine which are statistically significant. Naturally, it is desirable to eventually include such additional observables in any determination of student learning. However, in the present investigation, we will focus on correct/incorrect steps.

What do we mean by a step? A student attempts some number of *steps* when solving a problem. Usually, a step j is associated with creating/modifying a single user interface object (writing an equation, drawing a vector, defining a quantity, *et cetera*) and is a distinct part of the problem solution (that is, help-giving dialogs are not considered to be steps). A student may attempt a particular problem solving step, delete the object, and later attempt that solution step again. A step is an *opportunity* to learn a given Knowledge Component (KC) [6] if the student must apply that KC or skill to complete the step.

For each KC and student, we select all relevant step attempts and mark each step as "correct" (or 1) if the student completes that step correctly without any preceding errors or requests for help; otherwise, we mark the step as "incorrect" (or 0). A single student's performance on a single KC can be expressed as a bit sequence, *exempli gratia* 00101011.

2. THE STEP MODEL

We need to compare the student log data to some sort of model of learning. In another paper [5], we introduced the "step model" and showed that it was competitive with other popular models of learning when applied to individual student log data. It is defined as:

$$P_{\text{step}}(j) = \begin{cases} g, & j < L \\ 1 - s, & j \geq L \end{cases} \qquad (1)$$

where L is the step where the student first shows mastery of the KC, g is the "guess rate," the probability that the student gets a step correct by accident, and s is the "slip rate," the chance that the student makes an error after learning the skill. These are analogous to the guess and slip parameters of BKT [3]. This model assumes that learning occurs all at once, reminiscent of "eureka learning" discussed by [1].

2.1 Method

We examined log data from 12 students taking an intensive introductory physics course at St. Anselm College during summer 2011. The course covered the same content as a normal two-semester introductory course. Log data was recorded as students solved homework problems while using the Andes intelligent tutor homework system [7]. 231 hours of log data were recorded. Each step was assigned to one or more different KCs. The dataset contains a total of 2017 distinct student-KC sequences covering a total of 245 distinct KCs. We will refer to this dataset as student dataset $\mathcal{A}$. See Figure 1 for a histogram of the number student-KC sequences having a given number of steps.

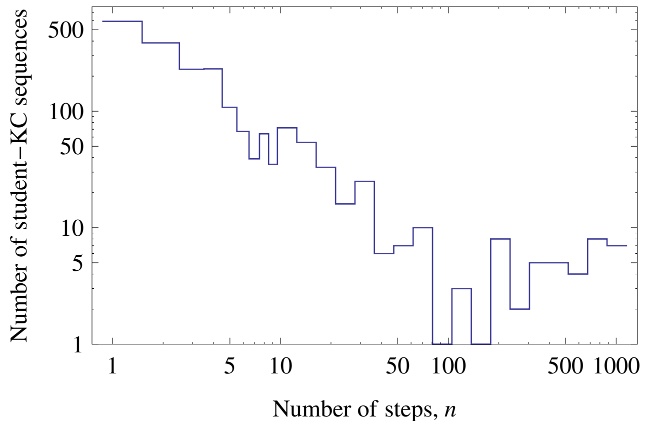


Figure 1: Histogram of number of distinct student-KC sequences in student dataset $\mathcal{A}$ having a given number of steps n.

Most KCs are associated with physics or relevant math skills while others are associated with Andes conventions or user-interface actions (such as, notation for defining a variable). The student-KC sequences with the largest number of steps are associated with user-interface related skills, since these skills are exercised throughout the entire course.

One of the most remarkable properties of the distribution in Fig. 1 is the large number of student-KC sequences containing just a few steps. The presence of many student-KC sequences with just one or two steps may indicate that the default cognitive model associated with this tutor system may be sub-optimal; there has not been any attempt, to date, to improve on the cognitive model of Andes with, say, Learning Factors Analysis [2]. Another contributing factor is the way that introductory physics is taught in most institutions, with relatively little repetition of similar problems. This is quite different than, for instance, a typical middle school math curriculum where there are a large number of similar problems in a homework assignment.

3. MULTI-MODEL APPROACH

We need to determine the step where a specific student has learned a particular skill. Our strategy is to take the step model, $P_{\text{step}}(j)$, and treat L as a constant, yielding a set of n sub-models $P_{\text{step},L}(j)$, one for each value of L. We then fit each of the n sub-models to the student data and calculate an AIC value. Finally, we find the Akaike weighs for each of the sub-models. The Akaike weights give the relative probability that learning occurred at each step.

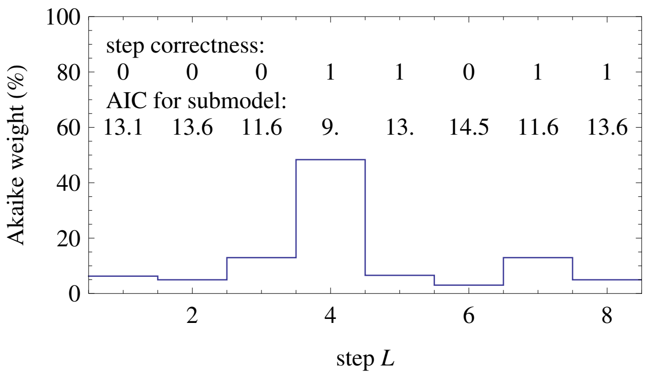


Figure 2: Akaike weights for the sub-models $P_{\mathrm{step,L}}(j)$. This gives the relative probability that the student learned the KC just before step L. The case $L = 1$ corresponds to no learning occurring during use of the ITS.

Let us illustrate this technique with a simple example. Suppose the bit sequence for a particular student-KC sequence is 00011011 (8 opportunities); see Fig. 2. We fit this bit sequence to 8 sub-models of the step model, corresponding to $L \in \{1, 2, \ldots, 8\}$, by maximizing the log likelihood, $\log \mathcal{L}_L$. The associated AIC values are given by $\mathrm{AIC}_L = 2K - \log \mathcal{L}_L$ where K is the number of fit parameters. Note that there are two parameters (s and g) when $L > 1$ and there is only one parameter (s) when $L = 1$. Not surprisingly, the best fit (lowest AIC) corresponds to the first "1" in the bit sequence at step 4. From the AICs, we calculate the Akaike weights

$$w_L = \frac{e^{-\mathrm{AIC}_L/2}}{\sum_{L'} e^{-\mathrm{AIC}_{L'}/2}} \,. \tag{2}$$

The Akaike weight w_L gives the relative probability that sub-model $P_{\mathrm{step},L}(j)$ is, of all the sub-models, the closest to the the model that actually generated the data.

Note that the case $L = 1$ corresponds to the student having "learned the skill" some time before the first step or after the last step. That is to say, the student does not acquire the skill while using the tutor system. Thus, w_1 should be interpreted as the relative probability that no learning has occurred while using the tutor system.

4. WEIGHTED GAIN

Our ultimate goal is to distinguish steps that result in learning from steps that do not. Hopefully, one can use this information to infer something about the effectiveness of the help given on a particular step, or the effectiveness of the student activity on that step.

It is not sufficient to know *when* learning has occurred but one must also determine *how much* learning has occurred. Consider the bit sequence 11011000. When fit to the step model, the best fit will occur at $L = 6$ but this would correspond to a *decrease* in student performance for that skill. In many cases seen in our log data, the change in student performance is almost zero. In order to take this into account, we propose using the Akaike weight w_L times the associated performance gain Δ_L to characterize a step. We define the performance gain $\Delta_L = 1 - \hat{g} - \hat{s}$ where $\hat{g}$ and $\hat{s}$ are the

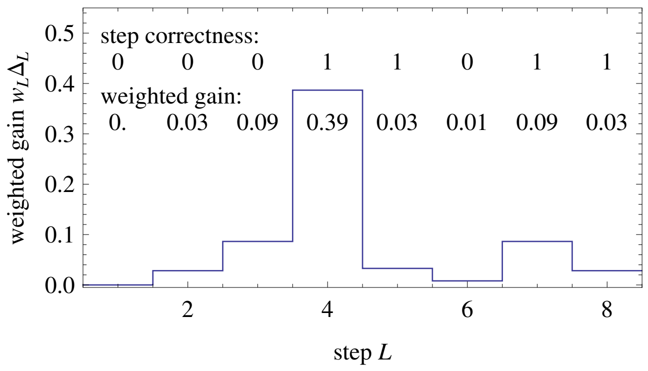


Figure 3: Weighted gain $w_L\Delta_L$ as a function of L for an example bit sequence. The associated quality factor is $Q = 0.66 \pm 0.29$; it is significantly smaller than 1 since the student makes a slip on step 6. Q is significantly greater than zero at the $p = 0.01$ level.

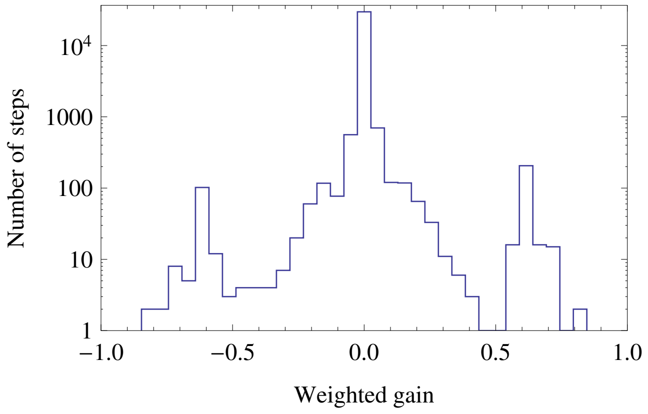


Figure 4: Histogram of weighted gains $w_L\Delta_L$ for all steps in all student-KC sequences of dataset $\mathcal{A}$.

Maximum Likelihood estimators for g and s given by sub-model $P_{\mathrm{step},L}(j)$. For the "no learning" case $L = 1$, we set $\Delta_1 = 0$. We will call $w_L\Delta_L$ the "weighted gain" associated with $P_{\mathrm{step},L}(j)$. A calculation of $w_L\Delta_L$ for an example bit sequence is shown in Fig. 3. Not surprisingly, the largest gain occurs at $L = 4$, corresponding to the first 1 in the bit sequence. The remaining weighted gains are much smaller.

A histogram of $w_L\Delta_L$ for student dataset $\mathcal{A}$ is shown in Fig. 4. We find that the vast majority of steps (29730) have almost zero weighted gain. We also see that there is a significant number of steps with negative gain (988), but there are somewhat more steps with positive gain (1312) .

The fact that there are so many steps with negative gain is symptomatic of bit sequences that are very noisy (a lot of randomness). Indeed, if we compare the histogram for student dataset $\mathcal{A}$ with the histogram for a randomly generated dataset $\mathcal{R}$ (we take $\mathcal{A}$ and randomly permute the steps) we find a similar distribution; see Fig. 5.

What would the distribution look like if the data weren't so noisy? To see this, we generated an artificial "ideal" dataset $\mathcal{I}$ where there were no slips or guesses, but having the same

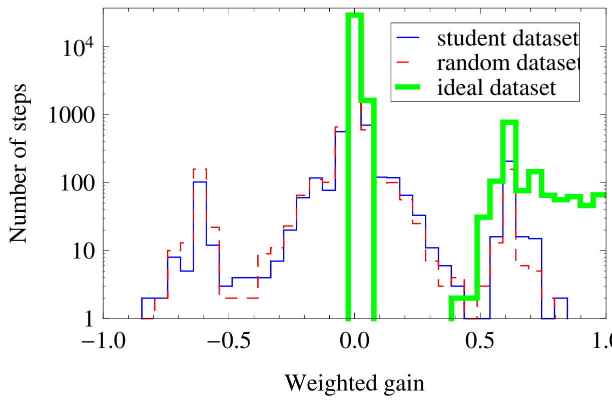


Figure 5: Histogram of weighted gains $w_L\Delta_L$ for the student dataset $\mathcal{A}$, a randomly generated dataset $\mathcal{R}$, and an artificial ideal dataset $\mathcal{I}$.

length distribution as $\mathcal{A}$ (Fig. 1). Thus, the bit sequences in $\mathcal{I}$ all have the form $00\cdots011\cdots1$. In this case, for each student-KC sequence, we expect a single large weighted gain (corresponding to the first 1 in the bit sequence) and the remaining weighted gains to be nearly zero. The resulting distribution of gains is shown in Fig. 5.

We propose to use the following average of the weighted gains as a "quality index" for determining how suitable a dataset is for determining the point of learning for an individual student-KC sequence:

$$Q = \frac{1}{N}\sum_\alpha\sum_L w_L\Delta_L \tag{3}$$

where α is an index running over all N student-KC sequences in a dataset. We use the sample standard deviation of the weighted gains $w_L\Delta_L$ to calculate the standard error associated with Q. An example calculation is shown in Fig. 3.

For the random dataset $\mathcal{R}$, the distribution of Δ_L is symmetric about zero and Q approaches zero as $N\to\infty$. For the "ideal" dataset $\mathcal{I}$, we expect that, when L coincides with the first 1 in the bit sequence, w_L will be nearly one with the associated Δ_L also nearly one so that $Q\to 1$ in the limit of many opportunities. Numerically, we obtain $Q = 0.5240 \pm 0.0003$. The fact that it is smaller than one is due to the large number of student-KC sequences having just a few steps. For the student dataset $\mathcal{A}$, we obtain $Q = 0.0467 \pm 0.0065$, which is small, but significantly larger than zero ($p < 0.001$). Thus, we conclude that one can detect statistically significant learning when applying our method to this student dataset, with the location of that learning given by the Akaike weights w_L.

5. CONCLUSION

We believe that a direct estimate of the moment when a student learns a skill could be very useful for improving instruction, improving help-giving, and understanding student learning. However, the question of whether learning has occurred at a particular step can only be answered in a probabilistic sense: unambiguous "Aha moments" seem to be relatively rare. Using the Akaike Information Criterion, we have introduced a method for determining this probability.

As can be seen in Fig. 5, there is not much difference between our student dataset and a randomly generated dataset. However, the quality index Q which can be used to quantify the size of the signal of learning as well as the size of the background. We see that the quality index $Q = 0.0467 \pm 0.0065$ for the student dataset $\mathcal{A}$ is roughly 10% the size of Q for the ideal dataset $\mathcal{I}$; we interpret this to mean that the "signal" for learning is roughly 10% as big as the "noise." However, the fact that Q for the student dataset is seven standard deviations from zero means that we have detected learning for 2000 student-KC sequences with room to spare. Using the fact that the error is proportional $1/\sqrt{N}$, where N is the number of student-KC sequences, we estimate that we could still detect learning with only 260 student-KC sequences at the $p = 0.01$ level. This gives us an initial estimate for the amount of log data needed to measure the moment of learning, at least for students using the Andes tutor system.

Finally, we see that many of the student-KC sequences are quite short, as shown in Fig. 1. We speculate that this is due to to the way that introductory physics is typically taught, with relatively little reinforcement of specific KCs, emphasizing, instead, more general problem solving meta-skills. If we were to repeat this analysis for high school or grade school math, where there is more repetition, we speculate that there would be significantly fewer KCs with less than 10 opportunities and that detecting when learning has occurred would be significantly easier.

6. ACKNOWLEDGMENTS

Funding for this research was provided by the Pittsburgh Science of Learning Center which is funded by the National Science Foundation award No. SBE-0836012.

7. REFERENCES

[1] R. S. J. D. Baker, A. B. Goldstein, and N. T. Heffernan. Detecting learning moment-by-moment. *Int. J. Artif. Intell. Ed.*, 21(1-2):5–25, Jan. 2011.

[2] H. Cen, Kenneth Koedinger, and B. Junker. Learning factors analysis - a general method for cognitive model evaluation and improvement. In *Proceedings of the 8th international conference on Intelligent Tutoring Systems*, pages 164–175, Jhongli, Taiwan, June 2006. Springer-Verlag Berlin, Heidelberg.

[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, 1995.

[4] A. F. Heckler and E. C. Sayre. What happens between pre- and post-tests: Multiple measurements of student understanding during an introductory physics course. *Am. J. Phys.*, 78(7):768, 2010.

[5] B. van de Sande. Applying three models of learning to individual student log data. Under review, 2013.

[6] K. VanLehn. The behavior of tutoring systems. *Int. J. Artif. Intell. Ed.*, 16(3):227–265, Jan. 2006.

[7] K. Vanlehn, C. Lynch, K. Schulze, J. A. Shapiro, R. Shelby, L. Taylor, D. Treacy, A. Weinstein, and M. Wintersgill. The andes physics tutoring system: Lessons learned. *Int. J. Artif. Intell. Ed.*, 15(3):147–204, Aug. 2005.

Test-size Reduction for Concept Estimation

Divyanshu Vats[1], Christoph Studer[1], Andrew S. Lan[1],
Lawrence Carin[2], Richard G. Baraniuk[1]
[1]Rice University, TX, USA; [2]Duke University, NC, USA
{dvats, studer, mr.lan, larry, richb}@sparfa.com

ABSTRACT

Consider a large database of questions that assess the knowledge of learners on a range of different concepts. In this paper, we study the problem of maximizing the estimation accuracy of each learner's knowledge about a concept while minimizing the number of questions each learner must answer. We refer to this problem as test-size reduction (TeSR). Using the SPARse Factor Analysis (SPARFA) framework, we propose two novel TeSR algorithms. The first algorithm is nonadaptive and uses graded responses from a prior set of learners. This algorithm is appropriate when the instructor has access to only the learners' responses after all questions have been solved. The second algorithm adaptively selects the "next best question" for each learner based on their graded responses to date. We demonstrate the efficacy of our TeSR methods using synthetic and educational data.

Keywords

Learning analytics, sparse factor analysis, maximum likelihood estimation, adaptive and non-adaptive testing

1. INTRODUCTION

A course instructor is naturally interested in estimating how well learners understand certain concepts (or topics) that are relevant to the course. Information about each learner's understanding is useful in (i) providing feedback to instructors to assess whether the material is suitable for the class and (ii) recommending remediation/enrichment for concepts a learner has weak/strong knowledge of. In practice, accurate estimates for each learner's concept knowledge can be extracted automatically by analyzing the responses to a (typically large) set of questions about the concepts underlying the given course (see, e.g., [8] for the details). In order to minimize each learner's workload, however, it is important to reduce the number of questions, or—more colloquially—the *test-size*, while still being able to retrieve accurate estimates of each learner's concept knowledge. In what follows, we refer to this problem as *test-size reduction* (TeSR).

Contributions: We propose two novel algorithms for test-size reduction (TeSR). Our algorithms build on the *SPARse Factor Analysis* (SPARFA) framework proposed in [8], which jointly estimates the question–concept relationships, question intrinsic difficulties, and the latent concept knowledge of each learner, based solely on binary-valued graded response data obtained in an homework, test, or exam. Given the SPARFA model, we leverage theory of maximum likelihood (ML) estimators to formulate TeSR as a combinatorial optimization problem of minimizing the uncertainty in estimating the concept knowledge of each learner. We then propose two algorithms, one nonadaptive and one adaptive, that approximates the combinatorial optimization problem at low computational complexity using a combination of convex optimization and greedy iterations. The nonadaptive TeSR algorithm, referred to as NA-TeSR, reduces the test size in a way that enables accurate concept estimates *for all learners* in a course. The adaptive TeSR algorithm, referred to as A-TeSR, adapts the test questions to *each individual learner*, based on their previous responses to questions. A range of experiments with synthetic data and two real educational datasets demonstrates the efficacy of both TeSR algorithms.

Prior Work: Prior results on test-size reduction build primarily on the Rasch model [1–3, 6, 9], which characterizes a learner using a single ability parameter [11]. In contrast, the SPARFA model used in this paper characterizes a learner using their concept knowledge on multiple latent concepts. In this way, SPARFA models educational scenarios of courses consisting of multiple concepts more accurately. Moreover, we show using experiments in Section 4 that the efficacy of the Rasch model for TeSR is inferior to SPARFA combined with our TeSR algorithms. The problem of selecting "good" questions is related to the sensor selection problem [5, 7], which finds use in environmental monitoring, for example. However, measurements from sensor-networks are typically real-valued, whereas, TeSR relies on discrete measurements.

2. PROBLEM FORMULATION

2.1 SPARFA in a nutshell

Suppose we have a total set of Q questions that test knowledge from K concepts. For example, in a high school mathematics course, questions can test knowledge from concepts like solving quadratic equations, evaluating trigonometric identities, or plotting functions on a graph. For each question $i = 1, \ldots, Q$, let $\mathbf{w}_i \in \mathbb{R}^K$ be a column vector that represents the association of question i to all K concepts. Note that each question can measure knowledge from multiple concepts[1]. The j^{th} entry in $\mathbf{w}_i$, which we denote by w_{ij}, measures the association of question i to concept j. In other words, if question i does not test any knowledge from concept j, then $w_{ij} = 0$. Let $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_Q]^T$ be a sparse,

[1]Solving $x^2 - x = \sin^2(x) + \cos^2(x)$ for $x \in \mathbb{R}$, for example, requires conceptual understanding of both solving quadratic equations as well as trigonometric identities.

non-negative $Q \times K$ matrix, assuming that each question only tests a subset of all concepts. Let $\mu_i \in \mathbb{R}$ be a scalar that represents the intrinsic difficulty of a question. A larger (smaller) μ_i corresponds to an easier (harder) question. Let $\boldsymbol{\mu} = [\mu_1, \ldots, \mu_Q]^T$ be a $Q \times 1$ column vector that represents the difficulty of each question. Finally, let $\mathbf{c}^* \in \mathbb{R}^K$ be a column vector that represents the concept knowledge of a particular learner. *It is this parameter vector that we are interested in estimating accurately.*

To model the interplay between $\mathbf{W}$, $\boldsymbol{\mu}$, and $\mathbf{c}^*$, we use the SPARFA framework proposed in [8]. Let Y_i be a binary random variable that indicates whether question i has been answered correctly or not, indicated by 1 and 0, respectively. More specifically, the SPARFA model assumes that $Y_i \in \{0, 1\}$ admits the following distribution:

$$\Pr(Y_i = 1 \,|\, \mathbf{w}_i, \mu_i, \mathbf{c}^*) = \Phi(\mathbf{w}_i^T \mathbf{c}^* + \mu_i), \qquad (1)$$

where $\Phi(x) = 1/(1 + e^{-x})$ is the inverse logistic link function. In words, (1) says that the probability of answering a question correctly depends on a sparse linear combination of the entries in the concept understanding vector $\mathbf{c}^*$. This sparsity arises because of the assumption that $\mathbf{w}_i$ is sparse, i.e., it only contains a few non-zero entries. Given graded question responses from multiple learners, the factors $\mathbf{W}$ and $\boldsymbol{\mu}$ can be estimated using either the SPARFA-M or SPARFA-B algorithms introduced in [8].

2.2 Test-size reduction (TeSR)

The problem we consider in this paper is the selection of an appropriate subset of $q < Q$ questions so that $\mathbf{c}^*$, a learner's unknown concept understanding vector, can be estimated accurately. We assume that a set of responses from N learners, i.e., a binary-valued matrix $\widetilde{\mathbf{Y}}$, is known *a-priori*; an entry $\widetilde{Y}_{i,j}$ of $\widetilde{\mathbf{Y}}$ refers to whether a learner j answered question i correctly or incorrectly. In many educational settings, such a data matrix can be obtained by looking at past offerings of the same course. As mentioned in Section 2.1, the matrix $\widetilde{\mathbf{Y}}$ can be used to estimate the question to concept matrix $\mathbf{W}$ and the intrinsic difficulty vector $\boldsymbol{\mu}$ using the algorithms proposed in [8].

Suppose, hypothetically, that we choose a subset $\mathcal{I}$ of $q < Q$ questions, and we are given a response vector $\mathbf{y}_\mathcal{I}$. Let $\widehat{\mathbf{c}}$ be an estimate of the unknown concept knowledge vector $\mathbf{c}^*$ that can be computed using standard maximum likelihood (ML) estimators. The *test-size reduction* (TeSR) problem is to choose an appropriate set of questions $\mathcal{I}$ so that the error $\widehat{\mathbf{c}} - \mathbf{c}^*$ is as small possible. Although this problem seems impossible since we do not have access to the response vector $\mathbf{y}_\mathcal{I}$, it turns out that the covariance of the error $\sqrt{q}(\widehat{\mathbf{c}} - \mathbf{c}^*)$ can be approximated by the inverse of the Fisher information matrix [4], which is defined as follows:

$$\mathbf{F}(\mathbf{W}_\mathcal{I}, \boldsymbol{\mu}_\mathcal{I}, \mathbf{c}^*)) = \sum_{i \in \mathcal{I}} \frac{\exp(\mathbf{w}_i^T \mathbf{c}^* + \mu_i)}{(1 + \exp(\mathbf{w}^T \mathbf{c}^* + \mu_i))^2} \mathbf{w}_i \mathbf{w}_i^T \,. \qquad (2)$$

The notation $\mathbf{W}_\mathcal{I}$ refers to the rows of $\mathbf{W}$ indexed by $\mathcal{I}$. Similarly, $\boldsymbol{\mu}_\mathcal{I}$ refers to the entries in $\boldsymbol{\mu}$ indexed by $\mathcal{I}$. Thus, a natural strategy for choosing a "good" subset of questions $\mathcal{I}$, is to minimize the uncertainty (formally, the differential entropy) of a multivariate normal random vector with mean zero and covariance $\mathbf{F}(\mathbf{W}_\mathcal{I}, \boldsymbol{\mu}_\mathcal{I}, \mathbf{c}^*))^{-1}$. Consequently, the optimization problem considered in the remainder of the paper, referred to as the *test-size reduction* (TeSR) problem, corresponds to

$$\text{(TeSR)} \quad \widehat{\mathcal{I}} = \underset{\mathcal{I} \subset \{1,\ldots,Q\}, |\mathcal{I}| = q}{\arg\max} \log\det(\mathbf{F}(\mathbf{W}_\mathcal{I}, \boldsymbol{\mu}_\mathcal{I}, \mathbf{c}^*)) \,.$$

The main challenges in solving (TeSR) are (i) the TeSR problem is a combinatorial optimization problem and (ii) the concept knowledge vector $\mathbf{c}^*$ is *unknown*, so the objective function cannot be evaluated exactly.

Algorithm 1: Nonadaptive test-size reduction (NA-TeSR)

Step 1) First choose K questions by solving

$$\widehat{\mathcal{I}}_{[K]} = \underset{\mathcal{I} \subset \{1,\ldots,Q\}, |\mathcal{I}| = K}{\arg\max} \log\det\left(\mathbf{W}_\mathcal{I}^T \widehat{\mathbf{V}} \mathbf{W}_\mathcal{I}\right) \qquad (3)$$

using the convex optimization, see [7]. The entries of the diagonal matrix $\overline{\mathbf{V}}$ are defined as $\widehat{V}_{kk} = \exp(\widehat{v}_k)$, where $\widehat{v}_i = \frac{1}{N}\sum_{j=1}^N \log\left(\widetilde{Y}_{ij} - \frac{1}{N}\sum_{j=1}^N \widetilde{Y}_{ij}\right)^2$

Step 2) Select questions $K+1, \ldots, q$ in a greedy manner:

$$\widehat{\mathcal{I}}_{j+1} = \underset{i \in \{1\ldots,Q\} \setminus \widehat{\mathcal{I}}_{[j]}}{\arg\max} \widehat{v}_i \mathbf{w}_i^T \left(\mathbf{W}_{\mathcal{I}_{[j]}}^T \widehat{\mathbf{V}}_{\mathcal{I}_{[j]}} \mathbf{W}_{\mathcal{I}_{[j]}}\right)^{-1} \mathbf{w}_i.$$

3. TESR ALGORITHMS

Our proposed algorithms, that are data driven and computationally efficient, for solving TeSR are summarized in Algorithms 1 and 2. Due to space constraints, in what follows, we only present a high level summary of the methods.

Nonadaptive TeSR: Algorithm 1 summarizes a nonadaptive method (NA-TeSR) for solving the TeSR problem. To deal with the problem of the unknown $\mathbf{c}^*$ in (2), we notice that the coefficient of the term $\mathbf{w}_i \mathbf{w}_i^T$ in (2) is simply the variance of a learner in answering a question i. This variance can easily be estimated using the prior student response data $\widetilde{\mathbf{Y}}$. The first step in NA-TeSR is to estimate K questions, where K is the number of concepts involved in the question database. We are able to make use of properties of the determinant to formulate TeSR as a convex optimization problem, which we solve using low complexity methods in [7]. The second step is to select the remaining $q - K$ questions using a greedy algorithm that selects the "best" question iteratively until all q questions have been selected.

Remark 1: Note that when $\mathbf{W}$ is a $Q \times 1$ vector of all ones, the SPARFA model reduces to the Rasch model [11]. In this case, (TeSR) reduces to a problem of maximizing the sum of the variance terms over the selected questions. Thus, all the questions can be selected independently of the others when using the Rasch model. On the other hand, when using SPARFA, since we account for the statistical dependencies among questions, the questions can no longer be chosen independently as it is evident from Algorithm 1.

Adaptive TeSR: Our second algorithm, A-TeSR, is designed for the situation where one can iteratively and individually ask questions to a learner and then use the responses to *adaptively* select the next "best" question based on the previous responses. Such an approach is often referred to as *computerized adaptive testing* [12].

Algorithm 2: Adaptive test-size reduction (A-TeSR)

```
Choose $K$ questions $\mathcal{I}_{[K]}$ as in Step 1 of Algorithm 1.
Acquire graded learner responses $\mathbf{y}_{\mathcal{I}_{[K]}}$.
for $j = K+1, \ldots, q$ do
    Compute the ML estimate $\widehat{\mathbf{c}}$ using $\mathbf{y}_{\mathcal{I}_{[j-1]}}$
    if $\widehat{\mathbf{c}}$ exists then
        Find $\mathcal{I}_j$ using Step 2 of Algorithm 2 by replacing $\widehat{v}_k$
        with $\mathbb{V}\mathrm{ar}[Y_i|\widehat{\mathbf{c}}]$.
    else
        Find $\mathcal{I}_j$ using Step 2 of Algorithm 2 by searching
        only amongst questions so that $\widehat{\mathbf{c}}$ will most likely
        exist in subsequent iterations.
    Acquire graded learner responses $\mathbf{y}_{\mathcal{I}_j}$.
```

The main idea behind A-TeSR is to use NA-TeSR until a maximum likelihood estimate (MLE) $\widehat{\mathbf{c}}$ of $\mathbf{c}^*$ can be computed. Then, we use $\widehat{\mathbf{c}}$ to evaluate the objective function of the TeSR problem and keep updating $\widehat{\mathbf{c}}$ as the learner responds to adaptively chosen questions. The main challenge of such an adaptive algorithm is the fact that a solution may not exist for certain patterns of the graded response of a given learner when computing the MLE. Thus, we would like our proposed adaptive algorithm to select questions such that the MLE can be computed using less number of questions than the nonadaptive algorithm. To this end, whenever the MLE does not exist, we choose the next question (using a simple modification of Step 2 of NA-TeSR) in such a way that the MLE may exist with higher probability in each subsequent iteration.

Remark 2: Just as in the case of NA-TeSR, A-TeSR reduces to an adaptive Rasch model-based method when $\mathbf{W}$ is a $Q\times 1$ vector of ones; see [3] for examples of such algorithms. The main differences when using the SPARFA model for selecting questions, as opposed to using the Rasch model, are that the condition for the MLE to exist changes and in each iteration we estimate a multidimensional concept vector as opposed a scalar parameter.

4. EXPERIMENTAL RESULTS

Baseline algorithms: We compare NA-TeSR and A-TeSR to four baseline algorithms.

- NA-Rasch and A-Rasch: Nonadaptive and adaptive methods that use the Rasch model to select questions. See Remark 1 and 2 for more details.
- Greedy: Iteratively selects a question from each concept until the required number of q questions has been selected. If all questions from a given concept have been exhausted, then Greedy skips to the next concept to select a question. Note that this approach completely ignores the intrinsic difficulty of a question when performing TeSR.
- Oracle: Uses the true underlying (but in practice unknown) vector $\mathbf{c}^*$ to solve the TeSR problem. Note that the oracle algorithm is not practical and is only used to characterize the performance limits of TeSR.

Performance measure: We assess the performance of the algorithms using the root mean-square error (RMSE), defined as $\mathsf{RMSE} = \|\hat{\mathbf{c}} - \mathbf{c}^*\|_2$. Although $\mathbf{c}^*$ is known for synthetic experiments, for real data, we assume that the ground truth is the concept vector estimated when asking all Q available questions.

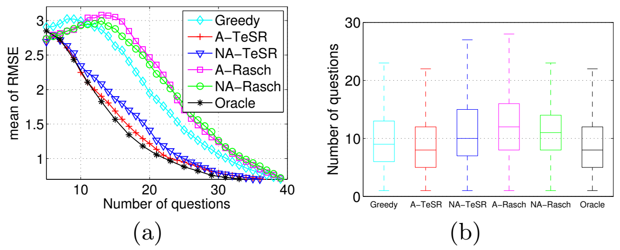


Figure 1: TeSR methods for synthetic data.

Methods: In the experiments shown next, we assume that a matrix $\mathbf{Y}$ is given that contains graded responses of Q questions from M students. As mentioned in Section 2, for real data, we use SPARFA-M [8] to estimate $\mathbf{W}$, $\boldsymbol{\mu}$, and the ground truth concept values of each learner. For each learner, we apply the baseline and our proposed TeSR algorithms using $\mathbf{W}$ and a training data $\widetilde{\mathbf{Y}}$ obtained after removing the responses of the learner from the matrix $\mathbf{Y}$. To show the performance of our TeSR algorithms, we report the mean of the RMSE evaluated over all M learners.

MLE convergence: As mentioned in Section 3, the MLE may not exist for certain patterns of the response vectors. In the case of inexistent ML estimates, we make use of the sign of the ML estimates (since each value in $\widehat{\mathbf{c}}$ will either be ∞ or $-\infty$) to compute the RMSE. We then assign each entry in $\hat{\mathbf{c}}$ to the worst (for $-\infty$) or best (for $+\infty$) value obtained from a prior set of learners who have taken the course. In our simulations, these worst and best concept values are computed using the training data $\widetilde{\mathbf{Y}}$.

Synthetic Data: We generated a sparse 50×5 matrix $\mathbf{W}$ that maps 50 questions to 5 concepts. There were roughly 30% non-zero entries in $\mathbf{W}$ with the non-zero entries chosen from an exponential random variable with parameter $\lambda = 2/3$. Each entry in the intrinsic difficulty vector $\boldsymbol{\mu}$ was generated from a standard normal distribution. We assumed 25 learners whose concept understanding vectors were again generated from a standard normal distribution. For each $\mathbf{Y}$, we computed the reduced test-size with $q = 5, 6, \ldots, 44$.

Figure 1(a) shows the mean value of the RMSE over 100 randomly generated response vectors $\mathbf{Y}$. Note that the mean RMSE is taken over all 25 learners. We observe that NA-TeSR and A-TeSR are superior to the baseline algorithms A-Rasch, NA-Rasch, and Greedy. This observation suggests that the Rasch model is not an appropriate model for selecting questions for the purpose of test-size reduction in courses having more than one underlying concept.

Algebra test dataset: The first dataset was obtained by a high school algebra test administered on Amazon's Mechanical Turk (see [8] for more details). This dataset contains no missing data and consists of responses from 99 learners on 34 questions.

We used SPARFA-M assuming that there are $K = 3$ latent concepts. The estimated concept–question matrix $\mathbf{W}$ contains roughly 40% non-zero values. Figure 2(a) shows the

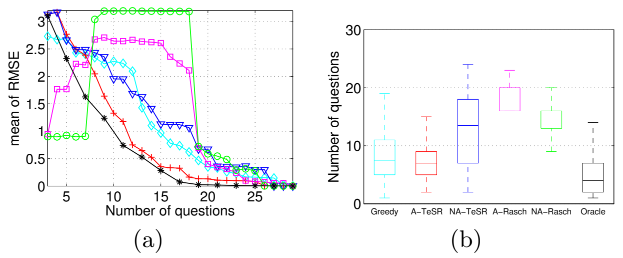


Figure 2: Mechanical Turk algebra test with 3 concepts; see Figure 1(a) for the legend.

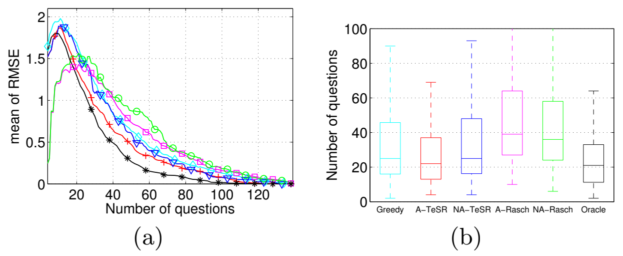


Figure 3: AssisMENT system data with 4 concepts; see Figure 1(a) for the legend.

mean RMSE over 78 learners[2]. We see similar trends as in the synthetic experiments. The main difference is that the performance of Rasch-based algorithms is much worse when compared to the synthetic data case. As we will explain later, this behavior is mainly because the ML estimates for the Rasch model did not converge for $q < 17$. Interestingly, for $q < 7$, the mean RMSE of NA-TeSR is much lower when compared to other algorithms. This behavior can be addressed to the fact that we deal with convergence failures of the ML estimates. Furthermore, we note that the mean performance of Greedy is better, in some regimes, than NA-TeSR. However, this gain in performance, for some questions, comes at the cost of slightly higher variability in the estimation of concept knowledge.

ASSISTment system dataset: The second real educational dataset corresponds to response data obtained from the ASSISTment system that was studied in [10]. The original data contained responses from 4354 learners on 240 questions. There are a large number of missing responses in this dataset. In order to get a dataset with a sufficient number of observed entries, we focused on a subset of 219 questions answered by 403 learners. The resulting trimmed $\mathbf{Y}$ matrix has roughly 75% missing values. Figures 3(a) shows the associated results and we observe trends that are similar to the algebra test dataset.

How many questions are needed? Another interesting measure to evaluate the performance of the TeSR algorithms is the number of questions needed for the ML to converge. Intuitively, this measure signifies the number of questions needed to get accurate estimates of each learner's concept knowledge. Figures 1(b)–3(b) show box plots of the number of questions needed for the ML estimates to converge for each algorithm and for each dataset considered here. Each box corresponds to the 25th and 75th percentiles over all learners in a class. We see that the A-TeSR algorithm is the fastest to converge amongst all the practical algorithms (the oracle algorithm is not practical since it utilizes information about the unknown concept vector of interest $\mathbf{c}^*$).

[2]For some of the questions, the ML estimate did not exist when using all the 34 questions; hence, the ground truth could not be computed.

Acknowledgements

Thanks to Andrew Waters for inspiring discussions and Daniel Calderon for administering the Algebra Test on Amazon's Mechanical Turk. DV thanks the Institute for Mathematics and its Applications (IMA) for financial support. This work was supported by the National Science Foundation under Cyberlearning grant IIS-1124535, the Air Force Office of Scientific Research under grant FA9550-09-1-0432, the Google Faculty Research Award program, and the Swiss National Science Foundation under grant PA00P2-134155.

5. REFERENCES

[1] S. Buyske. *Applied optimal designs*, chapter Optimal design in educational testing, pages 1–16. John Wiley & Sons Inc, 2005.

[2] H. Chang and Z. Ying. A global information approach to computerized adaptive testing. *Applied Psychological Measurement*, 20(3):213–229, 1996.

[3] H. Chang and Z. Ying. Nonlinear sequential designs for logistic item response theory models with applications to computerized adaptive tests. *The Annals of Statistics*, 37(3):1466–1488, Jun. 2009.

[4] L. Fahrmeir and H. Kaufmann. Consistency and asymptotic normality of the maximum likelihood estimator in generalized linear models. *The Annals of Statistics*, 13(1):342–368, 1985.

[5] D. Golovin, M. Faulkner, and A. Krause. Online distributed sensor selection. In *Proc. ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2010.

[6] U. Graßhoff, H. Holling, and R. Schwabe. Optimal designs for the Rasch model. *Psychometrika*, pages 1–14.

[7] S. Joshi and S. Boyd. Sensor selection via convex optimization. *IEEE Transactions on Signal Processing*, 57(2):451–462, 2009.

[8] 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*, Nov. 2012, submitted.

[9] W. J. Linden and P. J. Pashley. *Elements of adaptive testing*, chapter Item selection and ability estimation in adaptive testing, pages 3–30. Springer, 2010.

[10] Z. Pardos and N. Heffernan. Modeling individualization in a Bayesian networks implementation of knowledge tracing. *User Modeling, Adaptation, and Personalization*, pages 255–266, 2010.

[11] G. Rasch. *Probabilistic Models for Some Intelligence and Attainnment Tests.* Studies in mathematical psychology. Danmarks paedagogiske Institut, 1960.

[12] W. J. van der Linden and C. A. W. Glas. *Computerized adaptive testing: Theory and practice.* Springer, 2000.

Reading into the Text: Investigating the Influence of Text Complexity on Cognitive Engagement

Ben Vega
University of Memphis
365 Innovation Drive
Memphis, TN 38152
bavega@memphis.edu

Shi Feng
University of Memphis
365 Innovation Drive
Memphis, TN 38152
sfeng@memphis.edu

Blair Lehman
University of Memphis
365 Innovation Drive
Memphis, TN 38152
balehman@memphis.edu

Arthur C. Graesser
University of Memphis
365 Innovation Drive
Memphis, TN 38152
graesser@memphis.edu

Sidney D'Mello
University of Notre Dame
384 Fitzpatrick Hall
Notre Dame, IN 46556
sdmello@nd.edu

ABSTRACT

Boredom and disengagement have been found to negatively impact learning. Therefore, it is important for learning environments to be able to track when students disengage from a learning task. We investigated a method to track engagement during self-paced reading by analyzing reading times. We propose that a breakdown in the relationship between reading time and text complexity can reveal disengagement. A discrepancy (or decoupling) between attention resources and text complexity was computed via the absolute difference between reading times and the text's Flesch-Kincaid Grade Level, a measure of text complexity. As expected, decoupling varied as a function of text complexity. We also found that text complexity differentially impacted decoupling profiles for different types of participants (i.e., high vs. low comprehenders, fast vs. slow readers). These results suggest that decoupling scores may be a viable method to track disengagement during reading and could be used to trigger interventions to help students re-engage with the text and ultimately learn the material more effectively.

Keywords

Engagement, boredom, reading, text complexity

1. INTRODUCTION

It is widely acknowledged that engagement in a learning task is a necessary (but not sufficient) condition to achieve learning gains. There is also data to support this assumption. For example, student engagement was found to positively correlate with learning during interactions with an intelligent tutoring system (ITS) called AutoTutor, whereas boredom negatively correlated with learning [1]. Given this relationship, learning environments should seek to maximize engagement and minimize boredom and disengagement.

A variety of methods have been used to track student engagement during learning. These include body movements, facial expressions, aspects of language and discourse, self-reports, and observations by trained judges [2-4]. While these measures focus on the *affective* dimension of engagement, here we propose a method to track *cognitive* engagement during a reading task [5]. We posit that student engagement can be measured and tracked through a comparison of reading times and text complexity. The dance between reading times and text complexity can either align (e.g., the text becomes more difficult and reading times increase) or misalign (e.g., the text becomes more difficult but reading times do not reflect that). When discrepancies between reading times and text complexity occur, it may be indicative of students disengaging from the current learning task because they are not appropriately allocating resources to meet task demands.

Our work is grounded in previous research that has shown that reading times are robustly predicted by such language and text characteristics as word length and frequency, sentence length, and other discourse characteristics [6-7]. However, there is a lack of research that uses reading time measures to assess engagement in reading at a fine grained level. In the present paper we investigate the relationship between reading times and text complexity as assessed via Flesch-Kincaid Grade Level scores [8]. The present study investigates the discrepancy (or decoupling) between reading times and text complexity as a new method to track student engagement during a self-paced reading task.

2. METHOD

2.1 Participants

There were 64 participants in the present study who were recruited from Amazon's Mechanical Turk™ (AMT). AMT acts as a mediator between researchers and individuals to allow people to complete psychological tasks online for monetary compensation. Participants were limited to native English speakers of at least 18 years of age. On average, it took participants 33 minutes to complete the study and they were compensated US $4 for their participation. Past research suggests AMT is a reliable and valid source for collecting experimental data [9].

2.2 Materials

2.2.1 Texts
The texts were adapted from the electronic textbook that accompanies the *Operation ARA!* ITS with conversational agents [10]. ARA helps students learn about research methodology through electronic texts, conversations with agents, and critiquing flawed science. Each text discussed one of four research methods topics: causal claims, dependent variable, experimenter bias, and replication. A text began with a real world situation to ground the research methods concept being discussed. The text then continued with explanations and examples that suggest more generalized uses for the concept. Each text was approximately 1500 words long. Order of texts was counterbalanced across participants with a Latin Square.

2.2.2 Knowledge Assessment
Assessment of research methods knowledge was conducted after participants read each of the four texts. Each assessment consisted of six multiple-choice questions pertaining to the research methods concept in the text. The questions were developed using the Graesser-Person question asking taxonomy [11] specifically targeting logical, causal, or goal-oriented reasoning.

2.3 Procedure
Participants signed an electronic consent form and read the instructions for the self-paced reading task. Self-paced reading was adopted for this task to eliminate any pressures from time constraints. Participants were then presented with the first of four texts. A sentence-by-sentence reading paradigm was used in which texts were presented one sentence at a time and participants pressed the space bar to move on to the next sentence. Reading times were collected for each individual sentence from each of the four texts. After participants read the first text, they were presented with the knowledge assessment for the research methods concept in that text. Participants then began the second text and repeated this pattern for all four texts.

3. RESULTS & DISCUSSION
The analyses are divided into two sections. First, we discuss how the decoupling score was computed. Second, we explore the relationship between the decoupling score and text complexity.

3.1 Decoupling Score
The decoupling score was computed as a measure of the degree to which participants were appropriately allocating their attention based on text characteristics. In other words, as the text became more difficult, did participants spend more time reading the text? To compute this score each text was divided into overlapping groups of three sentences (triplets) such that sentences 1, 2, and 3 were one triplet, sentences 2, 3, and 4 were a second triplet and so on.

For each triplet, two values were used to compute the decoupling score. First, the total reading times for the three sentences in a triplet were summed and then standardized (i.e., converted to a z-score) on the subject level. Second, the Flesch-Kincaid Grade Level (FKGL) was computed for each triplet. The FKGL ranges from grades 1-12 and assesses the difficulty of a text based primarily on sentence length and the number of syllables. The FKGL was then also standardized (i.e., converted to a z-score) based on the TASA corpus, as computed by the Coh-Metrix text analysis tool [12].

The decoupling score was computed using the standardized reading time and FKGL scores for each triplet by taking the absolute difference between the two scores. Thus, in the present analysis, we are only focusing on the magnitude of decoupling and not the direction of decoupling (i.e., allocating too much or too little time based on text complexity).

3.2 Decoupling & Text Complexity
We investigated how decoupling scores varied as a function of text complexity, as assessed by FKGL. To investigate this relationship we plotted the decoupling score (y-axis) as a function of the standardized FKGL (x-axis) (see Empirical in Figure 1). FKGL scores were divided into ten equal partitions and the average is plotted in Figure 1. It is possible that the observed curve in Figure 1 is an artifact of the computations used to create the decoupling score. To address this issue, we constructed a randomly shuffled surrogate of the corpus. In this surrogate corpus, the FKGL score for each triplet was preserved, however, the ordering of the reading times was randomized. Ten surrogate corpora were constructed for each participant. Decoupling scores were then computed for each surrogate corpus and the average was used for the Shuffled curve in Figure 1.

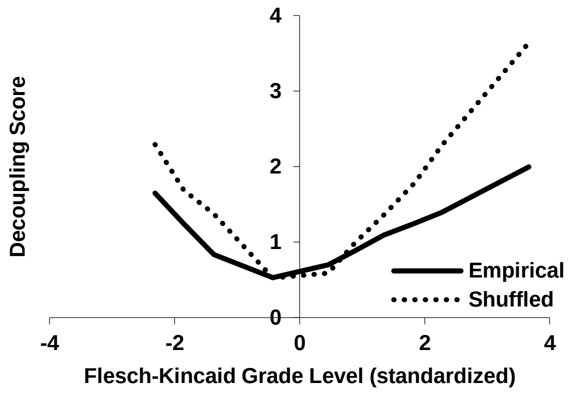


Figure 1. Decoupling scores as a function of text complexity

As can be seen in Figure 1, the curvilinear shape does partially appear to be an artifact of the computations used in the present analyses. However, the two curves were not identical. We investigated the differences between the two curves by conducting a 2 (curve: empirical or shuffled) x 10 (FKGL partition) repeated measures ANOVA. The ANOVA revealed that there were significant main effects comparing the two curves [$F(1,63) = 4.55$, $p < .001$, $\eta^2 = .343$] and the 10 partitions [$F(9,567 = 715$, $p < .001$, $\eta^2 = .919$] as well as a significant curve × partition interaction [$F(9,567) = 3.03$, $p < .001$, $\eta^2 = .639$]. Post hoc analyses with Bonferroni correction showed that the empirical curve and shuffled curve significantly differed at all partitions, suggesting that the empirical curve does differ from chance.

A closer examination of the empirical curve shows that when text complexity is near the mean level of complexity, i.e. FKGL (standardized) = 0, decoupling is less than when compared to the extremes of text difficulty (i.e., very easy or very hard). One way of interpreting this pattern is that the participants read at a pace appropriate for sentences with average complexity, but failed to adjust their reading speed in accordance with the more extreme levels of text complexity. This would result in the participants either spending too much or too little time on the easier and the harder portions of the texts.

It could be the case, however, that the relationship between text complexity and decoupling is obscured when all participants are combined into one group. To investigate this potential issue, we divided participants based on reading speed (fast, slow) and comprehension (i.e., score on knowledge assessment; high, low). Participants were divided via a median split for reading speed and comprehension, resulting in four groups: fast reader-high comprehender (FR-HC, *N* = 16), fast reader-low comprehender (FR-LC, *N* = 17), slow reader-high comprehender (SR-HC, *N* = 19), and slow reader-low comprehender (SR-LC, *N* = 12). Scores for the 4 groups are plotted in Figures 2-5.

ANOVAs showed that there were significant main effects and interaction terms for all four groups (*p's* < .05), with the exception that the curve main effect for the FR-LC group was not significant (*p* = .973). These poor readers were essentially insensitive to text complexity, as would be expected.

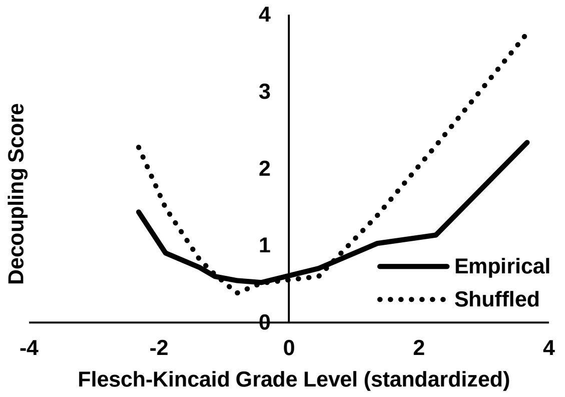


Figure 2. Decoupling scores as a function of text complexity for the SR-HC group

The SR-HC group was most sensitive to the more complex portions of the text. That is when the text was more difficult, this group had less decoupling than chance (i.e., Shuffled curve in Figure 2). On the other hand, the FR-LC group did not vary their reading behavior based on the text complexity. This can be seen in the close proximity of the Empirical and Shuffled curves in Figure 3.

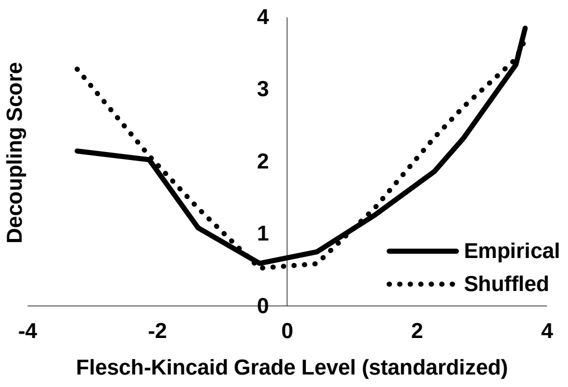


Figure 3. Decoupling scores as a function of text complexity for the FR-LC group

The FR-HC showed greater decoupling than chance (see Figure 4) at almost all levels of text complexity. This suggests that participants in this group could have been extremely vigilant to the text complexity, spending much less time on easy texts and much more time on difficult text segments. However, it is difficult from the present data to determine why this group of participants had these decoupling patterns.

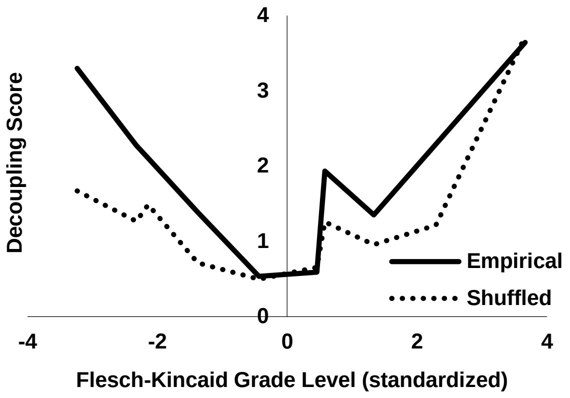


Figure 4. Decoupling scores as a function of text complexity for the FR-HC group

The SR-LC group had less than expected decoupling at extreme levels (easy or difficult) (see Figure 5). In other words, the more the text complexity deviated from the mean in either direction the decoupling was less than expected. This pattern may indicate that participants in this group needed the complexity level of the text to be more explicit or obvious for them to adapt their reading behavior. However, the overall decoupling score for this group did increase at the extremes, although this was still less than chance (see Shuffled curve in Figure 5). Unfortunately, it is somewhat difficult to interpret these results without knowing the direction of decoupling.

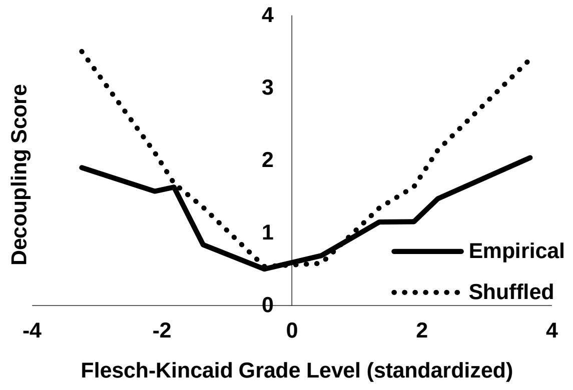


Figure 5. Decoupling scores as a function of text complexity for the SR-LC group

4. CONCLUSION

This study investigated a new method of measuring and tracking cognitive engagement during reading. The decoupling score was derived from the absolute difference between reading times and text complexity. We propose that this measure assesses cognitive engagement because if readers are engaged with the text, then their reading times should be adjusted based on text complexity. In other words, as the text becomes easier, reading times should become relatively faster and, conversely, as the text becomes more difficult reading times should become relatively slower. We found evidence that the relationship between reading time and text complexity did seem to reveal patterns of disengagement. Moreover, we found that the relationship between decoupling and the complexity of the text varies based on individual differences in reading speed and comprehension.

Despite these promising initial findings, we were not able to completely explain the patterns of decoupling for all types of participants. In particular, the relationship between decoupling pattern and comprehension scores was not clearly revealed in the differences between the empirical data and the shuffled surrogate corpus for participants classified as fast reader-high comprehenders. This highlights a limitation of using Flesch-Kincaid Grade Level to assess text complexity. Flesch-Kincaid assesses text complexity at a rather shallow level. It may be the case that more nuanced measures of text complexity will be able to shed more light on how decoupling impacts comprehension. Thus, in future work we plan to investigate more differentiated measures of text complexity, such as narrativity, syntactic simplicity, referential cohesion, word concreteness, and situation model cohesion using Coh-Metrix [12]. We are specifically targeting cohesion because past research has shown that cohesion and breakdowns in cohesion impact learning as well as interact with prior knowledge [13].

Student engagement over the course of a learning experience is a vital issue. This paper provides insight on how text complexity can factor into cognitive engagement levels and a possible measure for it. More importantly, this measure may be capable of tracking students' cognitive engagement across a span of text by simply using reading times and text characteristics (e.g., complexity). This measure of cognitive engagement could then be used to create texts that adapt in complexity level to increase cognitive engagement and maximize learning.

5. REFRENCES

[1] D'Mello, S. & Graesser, A. 2012. Dynamics of affective states during complex learning. *Learning and Instruction*, 22, 145-157.

[2] D'Mello, S., & Graesser, A. 2010. Multimodal semi-automated affect detections from conversational cues, gross body language, and facial features. *User Modeling and User-Adapted Interaction*, 20, 147-187.

[3] Jackson, S., & Marsh, H. 1996. Development and validation of a scale to measure optimal experience. The Flow State Scale. *J. of Sport & Exercise Psychology*, 18, 17-35.

[4] Baker, R., D'Mello, S., Rodrigo, M., & Graesser, A. 2010. 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. *Int. J. of Human-Computer Studies*, 68, 223-241.

[5] Fredricks, J., Blumenfeld, P., & Paris, A. 2004. School engagement: Potential of the concept, state of the evidence. *Review of Educational Research*, 74, 59-109.

[6] Haberlandt, K., & Graesser, A. 1985. Component processes in text comprehension and some of their interactions. *J. of Experimental Psychology: General*, 114, 357-374.

[7] Graesser, A., & McNamara, D. 2011. Computational analyses of multilevel discourse comprehension. *Topics in Cognitive Science*, 3, 371-398.

[8] Klare, G. 1974. Assessing readability. *Reading Research Quarterly*, 10, 62-102.

[9] Mason, W., & Suri, S. 2012. Conducting behavioral research on Amazon's Mechanical Turk. *Behavior Research Methods*, 44, 1-23.

[10] Halpern, D., Millis, K., Graesser, A., Butler, H., Forsyth, C., & Cai, Z. 2012. Operation ARA: A computerized learning game that teaches critical thinking and scientific reasoning. *Thinking Skills and Creativity*, 7, 93-100.

[11] Graesser, A., & Person, N. 1994. Question asking during tutoring. *American Educational Research Journal*, 31, 104-137.

[12] Graesser, A., McNamara, D., & Kulikowich, J. 2011. Coh-Metrix: Providing multilevel analyses of text characteristics. *Educational Researcher*, 40, 223-234.

[13] Ozuro, Y. Dempsey, K., McNamara, D. 2009. Prior knowledge, reading skill, and text cohesion in the comprehension of science texts. *Learning and Instruction*, 19, 228-242.

Using Students' Programming Behavior to Predict Success in an Introductory Mathematics Course

Arto Vihavainen, Matti Luukkainen, Jaakko Kurhila
University of Helsinki
Department of Computer Science
P.O. Box 68 (Gustaf Hällströmin katu 2b)
Fi-00014 University of Helsinki
{ avihavai, mluukkai, kurhila }@cs.helsinki.fi

ABSTRACT
Computer science students starting their studies at our university often fail their first mandatory mathematics course, as they are not required to have a strong background in mathematics. Failing can also be partly explained by the need to adjust to a new environment and new working practices. Here, we are looking for indicators in students' working practices that could be used to point out students that are at risk of failing some of their courses, and could benefit from an intervention. We present initial results on how freshman students' programming behavior in an introductory programming course can be used to predict their success in a concurrently organized introductory mathematics course. A plugin in students' programming environment gathers snapshots (time, code changes) from students actual programming process. Gathered snapshots are transformed to data items that contain features indicating e.g. deadline-driven mentality or eagerness. Our results using Bayesian networks indicate that we can identify students with a high likelihood of failing their mathematics course already at a very early phase of their studies using only data that represents their programming behavior.

Keywords
code snapshots, programming behavior, first-year challenges

1. INTRODUCTION
Students face a number of challenges during the first term of their studies at a higher education institution [4]. They need to find a place within a new community, and adjust their learning strategies and styles to fit the requirements of their chosen study. The multitude of challenges is often unforeseen and surprising for the students, and many end up failing some of their first courses. However, if a student has good study habits, she is more likely to succeed in her studies [12].

Computer science (CS) students typically start their studies with introductory programming courses and mathematics. Although mathematics is typically a minor subject, it is frequently considered as the basis of CS. Especially algorithms-related courses require a fair amount of mathematical maturity. As such, it is important that students fare well in their mathematics studies.

There are several systems that gather snapshots from students' programming process, e.g. [13, 15]. Currently, the research that utilizes snapshots has focused on e.g. modeling how students solve a specific exercise (see e.g. [11, 8]), and how e.g. compilation errors and successes can be used to predict success in a programming course (see e.g. [9, 14]). However, to our knowledge, there has not been much attention on how the students' *study practices* and *behavior*, especially their use of time extracted from snapshots, reflect on their results on current and parallel courses.

In our work, we are investigating students' programming process and seeking indicators of bad study habits that could be used to highlight students at-risk of failing some of their first term studies. In this article, we describe how we can identify students that are at risk of failing a 14-week introductory mathematics course using only their programming behavior from a parallel introductory programming course. Our initial results are promising, and with a population of 52 students, we are able to accurately predict the mathematics course success after only four weeks of programming.

This paper is organized as follows. In section 2, we briefly describe our educational arrangements as well as the tool that enables recording snapshots from students' programming process. Section 3 describes the data and the feature generation process. Section 4 describes the methodology used for analysing the data and presents our results, and Section 5 outlines future work.

2. CONTEXT, PEDAGOGY AND TOOLS
The academic year at the University of Helsinki is split into four seven-week teaching periods. Each period starts and ends simultaneously throughout the University, and each period is followed by a one week intermission before the start of the next period. The last week of each period is usually devoted to course exams.

The first two periods for CS majors are packed with mandatory courses: Introduction to Programming Part I (*7 weeks, period 1*), Introduction to Programming Part II (*7 weeks, period 2*), Software Modeling (*7 weeks, period 2*), that are offered by the Department of CS. In addition, students are expected to enroll into Introduction to University Mathematics (*14 weeks, periods 1 and 2*) that is organized by the Department of Mathematics and Statistics.

Both courses Introduction to University Mathematics and the Introduction to Programming Part I are organized using the Extreme Apprenticeship method (XA) [16], which is a modern interpretation of apprenticeship-based learning [5, 6]. XA values students' personal effort and intensive interaction between the learner and the advisor, and emphasizes deliberate practice [7] that aims towards mastering the craft.

As a craft can only be mastered by practising it, XA-based courses contain lots of exercises. For example, during the first week of their programming course, CS freshmen work already on tens of programming tasks.

As XA stresses activity to be as genuine as possible, the students start working with industry-strength tools from day one. We use NetBeans, which is an open source IDE (integrated development environment), bundled with an automated assessment service called Test My Code (TMC) [15], which is used to download and submit exercises; moreover, TMC is used to run tests on the students' code in order to verify the correctness of an exercise.

In addition to the assessment capabilities, on student's permission, TMC gathers data from students' programming process. Currently, a snapshot is taken whenever a student saves her code, compiles the code, or pastes code into the IDE. Each snapshot contains student id, timestamp, source code changes and possible configuration modifications.

3. DATA AND FEATURES

During the Fall 2012, we gathered over 48 000 snapshots from 52 students that participated to both Introduction to Programming Part I and Introduction to University Mathematics. The 52 students include only those that had more than 100 snapshots, i.e. they had not disabled the snapshot gathering mechanism at an early part of the course and had put at least some effort to solving the exercises. Out of the 52 students, 28 passed the mathematics course, and 24 failed it, while 43 passed the programming course. Although some students passed the courses later in a separate exam, we currently only consider their success in the actual course.

Students' snapshots are aggregated to describe weekly median, average, minimum, maximum and standard deviation of their working in the following dimensions:

1. hour of working
2. minutes to deadline
3. minutes between sequential snapshots
4. edit distance between sequential snapshots.

The first two aggregate statistics are generated by analysing the snapshot time and its distance from the deadline of the exercise that the student is currently working on. Aggregate statistics three and four are generated by sorting the snapshots based on their time and comparing the code changes and snapshot timestamps between sequential snapshots. Due to the large number of snapshots and the relatively small amount of code changes between each snapshot, we calculate the edit distances using an extension of the Ukkonen's algorithm by Bergel and Roach [2].

In addition to the above statistics, we generate weekly:

- minutes spent programming
- % of programming done during night (22-07)
- number of compilation errors
- number of style- and programming-related issues.

The number of compilation errors is gathered by compiling the program code for each snapshot, and gathering statistics out of the code compilation results. For identifying the style- and programming-related issues (e.g. wrong indentation, too long methods, copy-paste code, variables shadowing variables, and infinite loops), we utilize Checkstyle [3] and FindBugs [1].

If a student has not worked on the programming exercises during a specific week, values are entered as empty values. In our data, 21 of the students skipped at least one week of programming, and 9 of them ended up failing the mathematics course. We have purposefully left out the number of exercises each student has completed from the features, as our focus is on analysing the students' working behavior during the course.

3.1 Features

Each of the aggregated value is considered as a *feature*, and each week adds over 30 features that represent students' behavior during the specific week. Let us consider some of the features in more detail.

Figure 1 displays probability densities[1] for the standard deviation of snapshot time distances to deadline. Students that are more likely to fail the mathematics course have been working on the exercises during a smaller time period, e.g. during a single "crunch", while the students that are more likely to pass the mathematics course have worked on the exercises during several days.

The same "crunch" effect is visible in Figure 2, which displays probability densities for the maximum minutes between snapshots during week 3. Here, the students that take a larger pause (over 3.5 days) while working on the exercises are more likely to pass the mathematics course.

Figure 3 displays the amount of programming done during nights during week 6. The students that did not program between 22-07 hrs are slightly more likely to fail the mathematics course, and in our data, all the students that programmed more than 70% of their time during night passed the mathematics course.

[1]Note that the figures are plotted in R, and the used bandwidth for the density function is the default "nrd0". If a value is missing, i.e. has a value NA, it is removed from the dataset prior to plotting.

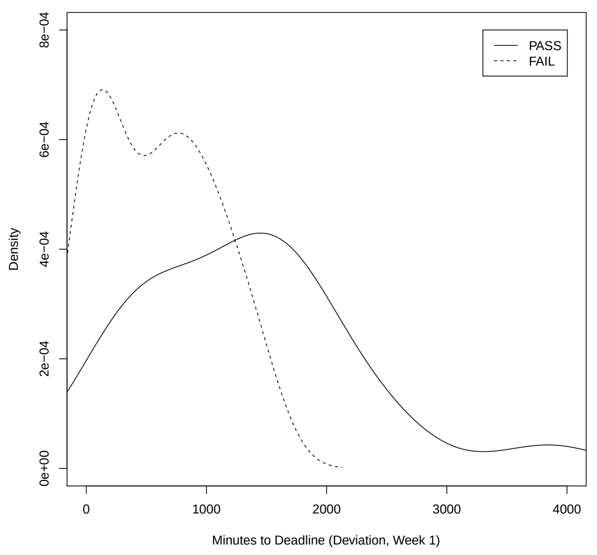


Figure 1: "Minutes to Deadline (Deviation, Week 1)" displayed using probability densities for groups that have passed and failed the course Introduction to University Mathematics.

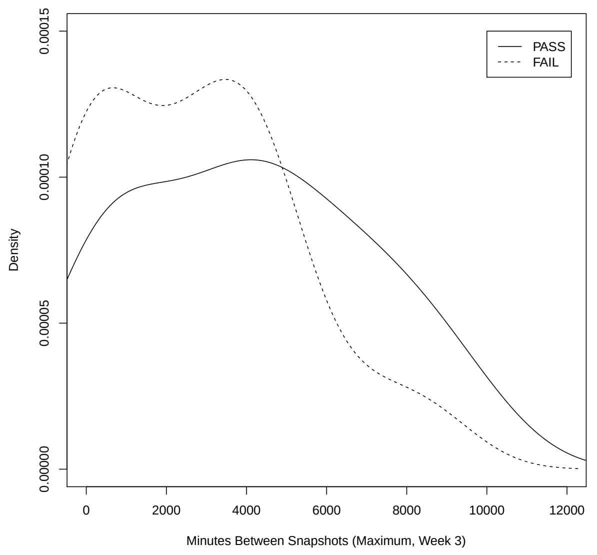


Figure 2: "Minutes Between Snapshots (Maximum, Week 3)" displayed using probability densities for groups that have passed and failed the course Introduction to University Mathematics.

4. METHODOLOGY AND RESULTS

We consider identifying the students' that are likely to succeed or fail their introductory mathematics course a classification problem. The course result (pass/fail) is used as the class to predict, and each feature vector contains the aggregated values from a student's snapshots. In total there are

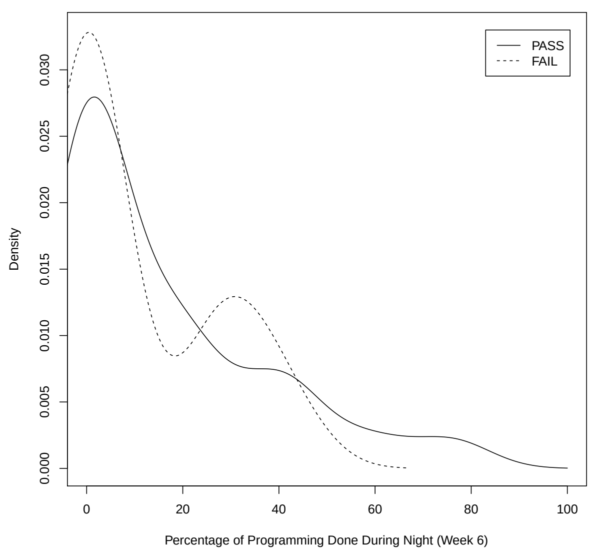


Figure 3: "Percentage of Programming Done During Night (Week 6)" displayed using probability densities for groups that have passed and failed the course Introduction to University Mathematics.

52 feature vectors with class labels, one for each student. We utilize a non-parametric Bayesian network tool called B-Course [10] for both modeling the dependencies between the features in the data, and for building the classifiers.

Validation is performed using student level leave-one-out cross-validation, and if a feature value is missing, i.e. a student has not programmed during a specific week, B-Course ignores the value when calculating the class probabilities.

We have built 12 separate classifiers for our data. Half of the classifiers are built using data from only the students' behavior, without results from static and dynamic code analysis, i.e. compilation errors and style- and programming-related issues, and the other half include static and dynamic code analysis results. Each of the classifier represents students' behavior up to a specific week in the programming course. The features for separate weeks are currently the same; students' weekly behavior is aggregated to feature values.

The following results contain accuracy, precision, recall, and the weighted harmonic mean of precision and recall (F-measure). Precision, recall, and F-measure are calculated assuming that we are predicting success in the mathematics course.

Results for the classifiers that did not include code analysis results are described in Table 1. Already after one week of programming, we are able to identify students' that are likely to fail the mathematics course with an 84.6% accuracy. After 5 weeks of programming, the accuracy is 98.1%.

Table 2 describes the results where dynamic and static code analysis results are included in addition to the students' be-

Week	Accuracy	Precision	Recall	F-Measure
1	84.6 %	85.7 %	85.7 %	85.7 %
2	88.5 %	92.3 %	85.7 %	88.9 %
3	92.3 %	92.3 %	92.3 %	92.3 %
4	94.2 %	100 %	89.3 %	94.3 %
5–6	98.1 %	100 %	96.4 %	98.2 %

Table 1: Results for data that includes students' programming behavior, and excludes compilation errors and style- and programming-related issues.

havior. After a single week of programming, the accuracy is `88.5%`, and at the end of fourth week, we are able to identify the students' at risk with `100%` accuracy.

Week	Accuracy	Precision	Recall	F-Measure
1	88.5 %	92.3 %	85.7 %	88.9 %
2	94.2 %	96.3 %	92.9 %	94.5 %
3	96.2 %	96.4 %	96.4 %	96.4 %
4–6	100 %	100 %	100 %	100 %

Table 2: Results for data that includes students' programming behavior as well as compilation errors and style- and programming-related issues.

5. DISCUSSION AND FUTURE WORK

With our current dataset, we are able to predict the students' success and failure in a 14-week introductory mathematics course already after a few weeks of their studies based on their programming behavior. Our current data indicates that computer science freshman that have a tendency to "crunch" their exercises and start working close to the deadline are at a higher risk of failing their introductory mathematics course than the students that work during a longer time interval.

There is a need for intervention at an early part of the at-risk students' studies, which would direct the students towards more successful learning styles. However, we do not know if there is a direct causality between the working habits, and cannot tell if our subjects would really perform better by e.g. simply starting to work on their assignments earlier. Our current number of samples (52) is relatively small, and we need more data from future students.

Our current plan is to evaluate the students' working process during Fall 2013, and perform intervention(s) to a subset of the population that our current classifier indicates being at risk. We are also seeking more descriptive behavioral indicators from the programming data in addition to our current features. Overall, we are not only interested in a single course or a single semester, but the students' success in their whole studies.

6. REFERENCES

[1] N. Ayewah, D. Hovemeyer, J. D. Morgenthaler, J. Penix, and W. Pugh. Using static analysis to find bugs. *Software, IEEE*, 25(5):22–29, 2008.

[2] H. Berghel and D. Roach. An extension of Ukkonen's enhanced dynamic programming ASM algorithm. *ACM Trans. Inf. Syst.*, 14(1):94–106, Jan. 1996.

[3] O. Burn. Checkstyle homepage. *URL http://checkstyle.sourceforge.net/*, 2001–2012.

[4] M. R. Clark. Negotiating the freshman year: Challenges and strategies among first-year college students. *Journal of College Student Development*, 46(3):296–316, 2005.

[5] A. Collins, J. Brown, and A. Holum. Cognitive apprenticeship: Making thinking visible. *American Educator*, 15(3):6–46, 1991.

[6] A. Collins and J. G. Greeno. Situative view of learning. In V. G. Aukrust, editor, *Learning and Cognition*, pages 64–68. Elsevier Science, 2010.

[7] K. A. Ericsson, R. T. Krampe, and C. Tesch-romer. The role of deliberate practice in the acquisition of expert performance. *Psychological Review*, pages 363–406, 1993.

[8] J. Helminen, P. Ihantola, V. Karavirta, and L. Malmi. How do students solve parsons programming problems?: an analysis of interaction traces. In *Proceedings of the 9th annual international conference on International computing education research*, ICER '12, pages 119–126, New York, NY, USA, 2012. ACM.

[9] M. C. Jadud. Methods and tools for exploring novice compilation behaviour. In *Proceedings of the 2nd international workshop on Computing education research*, ICER '06, pages 73–84, New York, NY, USA, 2006. ACM.

[10] P. Myllymäki, T. Silander, H. Tirri, and P. Uronen. B-course: A web-based tool for bayesian and causal data analysis. *International Journal on Artificial Intelligence Tools*, 11(03):369–387, 2002.

[11] C. Piech, M. Sahami, D. Koller, S. Cooper, and P. Blikstein. Modeling how students learn to program. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education*, SIGCSE '12, pages 153–160, New York, NY, USA, 2012. ACM.

[12] S. B. Robbins, K. Lauver, H. Le, D. Davis, R. Langley, and A. Carlstrom. Do psychosocial and study skill factors predict college outcomes? a meta-analysis. *Psychological bulletin*, 130(2):261–288, 2004.

[13] J. Spacco, D. Hovemeyer, and W. Pugh. An eclipse-based course project snapshot and submission system. In *Proceedings of the 2004 OOPSLA workshop on eclipse technology eXchange*, eclipse '04, pages 52–56, New York, NY, USA, 2004. ACM.

[14] E. S. Tabanao, M. M. T. Rodrigo, and M. C. Jadud. Predicting at-risk novice java programmers through the analysis of online protocols. In *Proceedings of the 7th international workshop on Computing education research*, ICER '11, pages 85–92, New York, NY, USA, 2011. ACM.

[15] A. Vihavainen, M. Luukkainen, and M. Pärtel. Test my code: An automatic assessment service for the extreme apprenticeship method. In *2nd International Workshop on Evidence-based Technology Enhanced Learning*, pages 109–116. Springer, 2013.

[16] A. Vihavainen, M. Paksula, M. Luukkainen, and J. Kurhila. Extreme apprenticeship method: key practices and upward scalability. In *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education*, ITiCSE '11, pages 273–277, New York, NY, USA, 2011. ACM.

Poster Presentations (Regular Papers)

Do students really learn an equal amount independent of whether they get an item correct or wrong?

Seth Adjei[1], S. M. A. Salehizadeh[2], Yutao Wang[3], Neil Heffernan[4]
Department of Computer Science
Worcester Polytechnic Institute
100 Institute Road, Worcester, MA
[saadjei[1], ssalehizadeh[2], yutaowang[3], nth[4]]@wpi.edu

ABSTRACT

The field of educational data mining has been paying attention to Knowledge Tracing (KT) for a long time. Corbett and Anderson assumed the amount of learning that students do does not depend on whether students get items right or wrong. Ohlsson and others argued that the student should learn more from a previous incorrect performance. We decided to investigate a Bayes Network similar to KT but that allows us to have learning rates that are different according to whether students get items correct or not. While the idea of allowing learning rates from previous incorrect performances to be higher seems intuitive, our experiments showed that this way does not always lead to better predictions. Of course reasoning from a null result is dangerous, our contribution is that this intuitive idea is not one that other researchers should waste time in working on, unless they come up with a different model from the model we used (which is the naïve way of modifying KT).

Keywords:
Knowledge Tracing, Bayesian Networks, Learn from performance, Tutoring Strategies

1. INTRODUCTION

The field of Education Data mining has depended to a large extent on the model that was developed by Corbett and Anderson [2] and enhanced by a number of authors for predicting student performance. Over the years many new models have been built to improve upon the prediction accuracy of KT.

Wang and Heffernan have also shown that better predictions are achieved with the inclusion of additional parameters relating to the skills and groups to which a student belongs. [4]
Standard KT makes a number of assumptions, including the fact that the rate of learning is constant and that the transition from one knowledge state to the other is not dependent of previous performance. [2] Other researchers have introduced different models that seem to deal with this anomaly with the KT model [7], whiles some have compared these different models to determine which best predicts student performances considering prior performance. [6] Ohlsson theorized that humans in general are able to learn from their previous error performance, especially in situations where there is an explanation for the cause of the error. [1] Ohlsson further reiterated that in order to avoid repeating an incorrect action the knowledge behind the action must be changed. It is therefore intuitive that the student will learn from the previous performance. In this paper we present yet another modification of the KT model which considers the previous performance of a student on a particular item.

2. LP Model

We considered a new assumption for the KT model as follows: *"Students can learn from their previous observed performance once there is some tutoring associated with the wrong performance."* This resulted in a new model which we call "Learn from Performance" (LP) model and present in Fig.1b).

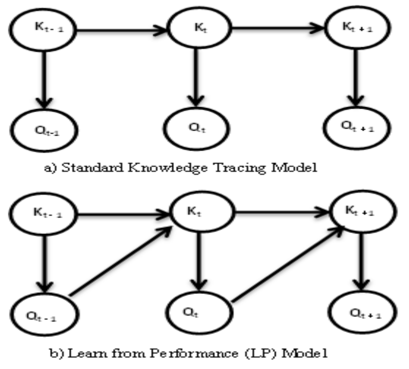


Figure.1 Student Performance Models (a) KT (b) LP

To account for learning after performance, our new model, LP, introduces one link from performance at time *t-1* (Q_{t-1}) to knowledge at time *t* (K_t). This modification therefore introduces two (2) additional parameters to the model. These parameters are *learn_from_correct (LC)*, and *learn_from_incorrect (LinC)*. *LC* is the probability that the student gained some additional knowledge from having answered the most previous question in the skill correct. This is especially so in the case where the student sees a different variation of questions relating to a given skill. *LinC* is the probability that the student has learnt something new from performing poorly from the most previous question. There is also the probability that the student had the knowledge prior to the previous performance but answered the previous question wrong and hence he lost it. In other words this is the probability of shallow learning or *confusion*.

3. METHODOLOGY AND RESULTS

To evaluate our model we used both real data from the ASSISTments Tutoring System [3] and simulated data. The Bayes Net Toolbox for Matlab [5] was used to implement the standard KT as well as the proposed LP model.

3.1 Simulated Data

In order to evaluate the LP model and to ensure that we were not over-fitting the model, we decided to test the model's strength using simulated data from both KT and LP. The networks' parameter values were set to ground truth values.

To evaluate the ability of the LP model to see if it is effective and can learn back its parameters, we generated data for 200 typical students from the LP Bayesian Network and used both models to predict the performance of students. In the simulated experiment we performed a 5-fold cross validation on both models and computed the performance metrics (MAE, RMSE and AUC).

The results indicated that LP learned most of its parameters: *prior, lean_from_incorrect (LinC), guess* and *slip* back to an appreciable degree. The *learn_from_correct (LC)* parameter, however, does not seem to get close to the ground truth values that were used in generating the simulation data for LP. Table 1 shows that LP outperformed KT reliably using LP generated data. However there were mixed results when KT generated data was used.

Table 1. Simulation Performance Comparison Results with LP Generated Data (1000 samples)

	MAE		RMSE		AUC	
Fold	KT	LP	KT	LP	KT	LP
1	0.326	**0.270**	0.399	**0.380**	0.707	**0.832**
2	0.335	**0.271**	0.409	**0.382**	0.779	**0.820**
3	0.336	**0.284**	0.409	**0.391**	0.798	**0.816**
4	0.351	**0.285**	0.426	**0.391**	0.798	**0.827**
5	0.334	**0.275**	0.405	**0.381**	0.762	**0.834**
Mean	0.336	**0.277**	0.410	**0.385**	0.794	**0.826**
P-value	**<0.05**		**<0.05**		**<0.05**	

3.2 Real Data Experiments

Given mixed results in simulation experiments, we further tested the model using real data to confirm our observation. The data set we used is from the ASSISTments Tutoring System. We randomly chose twenty (20) skills that have a minimum of 1000 rows of problem logs. Each row represented the student's performance of a given Skill Builder problem and the number of times the student has encountered problems of that nature including the current opportunity.

We employed the Expectation Maximization function that comes with the Bayes Net Toolbox within Matlab. We split the data randomly into five equal folds for each skill. We then performed a five-fold cross validation of the predictions for each skill and for each model.

Table 2 displays the MAE, RMSE and AUC values for each skill and for each model.

Table 2. Skill Prediction Performance Comparsion Results

SKILL	MAE		RMSE		AUC	
(#)Name	*KT*	*LP*	*KT*	*LP*	*KT*	*LP*
(1) Box and Whisker	0.349	**0.268**	**0.423**	0.487	**0.681**	0.500
(9) Stem and Leaf Plot	0.394	**0.321**	**0.447**	0.490	**0.599**	0.583
(10) Table	0.294	**0.187**	**0.386**	0.424	**0.539**	0.453
⋮	⋮	⋮	⋮	⋮	⋮	⋮
(58)Addition Whole Numbers	0.195	**0.116**	**0.313**	0.330	**0.557**	0.500
Mean	0.350	**0.287**	**0.415**	0.462	**0.611**	0.569
p-values	**<0.05**		**<0.05**		>0.05	

The results in Table 2 indicated that LP has better MAE values than KT. However, KT reliably outperformed LP with RMSE and not reliably so with AUC. These results show that the results do not indicate any better performances than

4. CONCLUSION

Ohlsson theorized that students do learn from their previous error performance, especially in the case where explanation of the reasons for the error is provided. On the basis of this, we developed a new naïve Bayes Network model that allows the amount of learning to increase when users get an item wrong. Our experiments with real and simulated data showed that we do not get better prediction of student performance with this proposed LP model than the standard KT model. Hence we conclude from our experiments that the assumption that Corbett and Anderson made was justifiable even though not intuitive according to Ohlsson. Our contribution is that this intuitive idea is not one that other researchers should waste time in working on, unless they come up with a different model from the model we used.

The code and data for these experiments are available at: http://users.wpi.edu/~saadjei/

REFERENCES

[1]. Ohlsson, S. (1996). Learning from performance errors. Psychological Review, 103(2), 241–262

[2]. Corbett, A. T., & Anderson, J. R. (1995). "Knowledge tracing: modeling the acquisition of procedural knowledge". User Modeling and User-Adapted Interaction, 4, 253–278

[3]. ASSISTments Tutoring System, www.assistments.org

[4]. Wang, Y., Heffernan, N.T. (2012). "The Student Skill Model". In Proceedings of the 11th International Conference on Intelligent Tutoring Systems, pp. 399-404, 2012

[5]. Murphy, K (2006). Bayes Net Tool Kit for Matlab

[6]. Yue Gong, Joseph E. Beck, Neil T. Heffernan (2011): How to Construct More Accurate Student Models: Comparing and Optimizing Knowledge Tracing and Performance Factor Analysis. I. J. Artificial Intelligence in Education 21(1-2): 27-46

[7]. Pavlik Philip, Cen Hao, Koedinger K. R. :(2009) Performance Factors Analysis--A New Alternative to Knowledge Tracing In Proceeding of the 2009 conference on Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling (2009), pp. 531-538

Analysis of students clustering results based on Moodle log data

Angela Bovo
Andil
Université Toulouse 1
IRIT
angela.bovo@andil.fr

Stéphane Sanchez
Université Toulouse 1
IRIT
stephane.sanchez@irit.fr

Olivier Héguy
Andil
olivier.heguy@andil.fr

Yves Duthen
Université Toulouse 1
IRIT
yves.duthen@irit.fr

ABSTRACT
This paper describes a proposal of relevant clustering features and the results of experiments using them in the context of determining students' learning behaviours by mining Moodle log data. Our clustering experiments tried to show whether there is an overall ideal number of clusters and whether the clusters show mostly qualitative or quantitative differences. They were carried out using real data obtained from various courses dispensed by a partner institute using a Moodle platform. We have compared several classic clustering algorithms on several group of students using our defined features and analysed the meaning of the clusters they produced.

Keywords
clustering, Moodle, analysis, prediction

1. INTRODUCTION

1.1 Context of the project
Our project aims to monitor students by storing educational data during their e-learning curriculum and then mining it. The reasons for this monitoring are that we want to keep students from falling behind their peers and giving up.

This project is a research partnership between a firm and an university. The partner firm connects our research with its past and current e-learning courses, hence providing us with real data from varied trainings.

All available data comes from a Moodle [5] platform where the courses are located. Moodle's logging system keeps track of what materials students have accessed and when. We then mine through such logs.

1.2 Clustering as a means of analysis
Clustering is the unsupervised grouping of objects into classes of similar objects. In e-learning, clustering can be used for finding clusters of students with similar behaviour patterns. In the example of forums, a student can be active or a lurker [1, 7]. These patterns may in turn reflect a difference in learning characteristics, which may be used to give them differentiated guiding [2] or to predict a student's chance of success [3]. They may also reflect a degree of involvement with the course, which, if too low, can hinder learning. The data contained in Moodle logs lends itself readily to clustering, after a first collecting and pre-processing step [6].

Our aim with this analysis will be to determine if there is an overall ideal number of clusters and whether the clusters show mostly qualitative or quantitative differences. We chose clustering, which is unsupervised, in order to better reflect the natural structure of our data. Because of this choice, the outcome of our experiments will not be directly relevant to the success of the students, but will rather reflect the differences in their usage of the LMS.

2. FEATURES CHOSEN TO AGGREGATE THE DATA
We have tried to aggregate the Moodle log data into a list of features that could capture most aspects of a student's online activity. The features we have selected are: the login frequency, the date of last login, the time spent online, the number of lessons read, the number of lessons downloaded as a PDF to read later, the number of resources attached to a lesson consulted, the number of quizzes, crosswords, assignments, etc. done, the average grade obtained in graded activities, the average last grade obtained, the average best grade obtained, the number of forum topics read, the number of forum topics created, and the number of answers to existing forum topics. For every "number of x" feature, we actually used a formula that would reflect both the distinct and total number of times that this action had been done. All of our features are normalized, with the best student for each grade obtaining the grade of 10, and others being proportionally rescaled.

3. EXPERIMENTAL METHOD

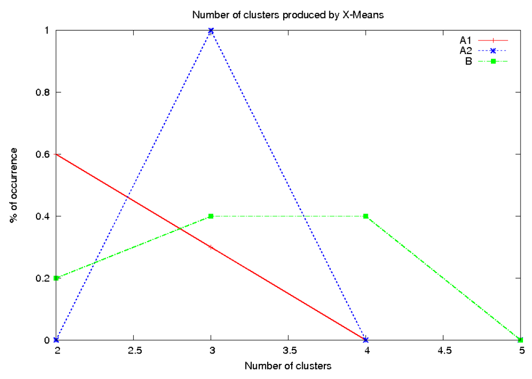


Figure 1: Sample clustering result with X-Means

All our experiments were performed using Weka [4], and converting the data first into the previously described features, then transforming these features into Weka attributes and instances. We can then view the feature data for each of these clusters or students in order to analyse the grouping.

In order to test the accuracy of the obtained clusters, we used the 10-fold cross-validation method, which directly outputs a mean error, which we show in figure 1. We have averaged over a few runs with different randomizing seeds. We executed the following clustering algorithms provided by Weka: Expectation Maximisation, Hierarchical Clustering, Simple K-Means, and X-Means. X-Means chooses a best number of clusters, which we show. The other algorithms take as an input parameter the required number of clusters. These numbers will be comprised between 2 and 5, based on the X-Means result.

We have selected 3 different trainings: two classes of a same training, which we will call Training A1 and A2, and a totally different training B. Training A1 has 56 students, A2 has 15 and B has 30. Both A1 and A2 last about a year while B lasts three months.

4. CLUSTERING RESULTS

4.1 Best number of clusters

The following figure shows the results of the four algorithms used on each of our three datasets. The first shows the frequency at which the X-Means algorithm proposed a given number of clusters. The other three graphs show the error for a given number of clusters for K-Means, Hierarchical clustering and Expectation Maximisation. We can see that all algorithms generally agree on at most 2 or 3 clusters.

4.2 Meaning of the clusters

To our surprise, the clusters observed for all three trainings did not show anything more relevant than a simple distinction between active and less active students, with variations according to the chosen number of clusters. We did not, for instance, notice any group that would differ from another simply by their activity on the forum.

To explain this, we offer the following possible reasons. Firstly, we have a relatively small number of students in each training (between 15 and 56), which may mean less variety in behaviour. Secondly, this training may be targeted towards a relatively homogeneous audience in terms of age, professional training, and habitual use of IT. Thirdly, a vicious circle effect can happen of the forum, because if few people use it, other students have less incentive for using it.

Hence, in about all observed clusters, the students were only quantitatively differentiated by a global activity level. It is also to be noticed that when the number of clusters was too large, clusters containing only one student, the most or least active of his training, tended to form. This phenomenon might be a good indicator that the number of clusters is too high without the help of a comprehensive study.

However, the fact that all differences were proportional also means that the student's activity level was also correlated to the grades they obtained in graded activities (which were not evaluative). This seems to indicate that in our trainings, using a quantity of activity is sufficient to help identify students in trouble, which is our global aim.

5. CONCLUSIONS AND FUTURE WORK

This paper proposes comprehensive and generic features that can be used for mining data obtained from Moodle courses. These features are then used to conduct a clustering of the data, using several algorithms, followed by an analysis, which seems to show very little qualitative difference in behaviour between students. It seems that a single feature, a kind of index of their global activity, would be almost sufficient to describe our data. This is also shown by the very little (2 to 3) number of clusters that is sufficient for describing our data. We propose several explanations for this surprising result, such as the small dataset, the homogeneity of our students and a vicious circle effect. However, the results mean that using our features or computing a quantity of activity could be enough to monitor students and notice which ones run a risk of failure.

6. REFERENCES

[1] M. Beaudoin. Learning or lurking? Tracking the "invisible" online student. *The Internet and Higher Education*, (2):147–155, 2002.

[2] J. Lu. Personalized e-learning material recommender system. In *Proc. of the Int. Conf. on Information Technology for Application*, pages 374–379, 2004.

[3] M. López, J. Luna, C. Romero, and S. Ventura. Classification via clustering for predicting final marks based on student participation in forums. In *Educational Data Mining Proceedings*, 2012.

[4] Machine Learning Group at the University of Waikato. Weka 3: Data mining software in java, 2013. http://www.cs.waikato.ac.nz/ml/weka.

[5] Moodle Trust. Moodle official site, 2013. http://moodle.org.

[6] C. Romero, S. Ventura, and E. García. Data mining in course management systems: Moodle case study and tutorial. *Computers and Education*, pages 368–384, 2008.

[7] J. Taylor. Teaching and Learning Online: The Workers, The Lurkers and The Shirkers. *Journal of Chinese Distance Education*, (9):31–37, 2002.

Mining the Impact of Course Assignments on Student Performance

Ritu Chaturvedi and C. I. Ezeife*
School of Computer Science, University of Windsor
Windsor, Canada N9B 3P4
rituch@uwindsor.ca and cezeife@uwindsor.ca

ABSTRACT

Educational model for higher education has shown a drift from traditional classroom to technology-driven models that merge classroom teaching with web-based learning management systems (LMS) such as Moodle and CLEW. Every teaching model has a set of supervised (e.g. quizzes) and/or unsupervised (e.g. assignments) instruments that are used to evaluate the effectiveness of learning. The challenge is in preserving student motivation in the unsupervised instruments such as assignments as they are less structured compared to quizzes and tests. The research applies association rule mining to specifically find the impact of unsupervised course work (e.g. assignments) on overall performance (e.g. exam and total marks).

1. INTRODUCTION

Research has shown that society is gradually drifting from the most common teacher-centered classroom teaching model to a hybrid educational model that combines classroom teaching and technology such as internet [2]. Some of the recent technology-based systems are web-based courses, learning content management systems (LMS), adaptive and intelligent web-based educational systems (Intelligent tutoring systems (ITS)) [2] and more recent online systems such MOOC (Massive open online course).Such web-based courses gather student data using activities such as quizzes, exams and assignments to measure their cognitive ability and additional data to measure other factors that could influence learning such as number of times a student has visited a webpage. The objective of this paper is to study the direct and indirect impact that unsupervised tasks (e.g. assignments) have on final marks and grades using association rule mining (ARM). ARM is an unsupervised learning method that looks for hidden patterns in data. An impact on the overall mark is considered to be direct (if the student achieves a score x in the assignment, a 10% of x contributes to the overall mark).An impact on final exam is considered to be indirect – student performing well in the assignments understands the course concepts well and hence performs proportionately well in the final exam. The motivation behind this research is to offer constructive suggestions to educators to help them effectively decide the optimal number of course assignments and the amount of weight that should be given to them (e.g. giving 15% weight to the assignments as opposed to 10%).

2. RELATED WORK

Data mining techniques such as classification, clustering and association rule mining have been used to provide guidance to students and teachers in activities such as predicting student's performance and failure rate, discovering interesting patterns among student attributes and finding students who have a low participation in collaborative work [1,2]. Even though a lot of work exists on mining educational data, no one so far has attempted to analyze the impact that assignments have on student performance in a course. In this paper, we mine student data from courses offered in the University of Windsor, Canada to study this impact.

* This research was supported by the Natural Science and Engineering Research Council (NSERC) of Canada under an operating grant (OGP-0194134) and a University ofWindsor grant.

3. THE PROPOSED SYSTEM FOR MINING STUDENT LEARNING

3.1Learning Management System Overview

CLEW (Collaboration and Learning Environment Windsor) is an LMS developed by University of Windsor and is used to support teaching and learning in face-to-face, distance and blended courses (www.uwindsor.ca/clew). Each course has a set of objectives and a set of supervised (e.g. tests) and unsupervised (e.g. assignments) instruments. Students can use discussion board and chat rooms to collaborate with peers or post concerns.

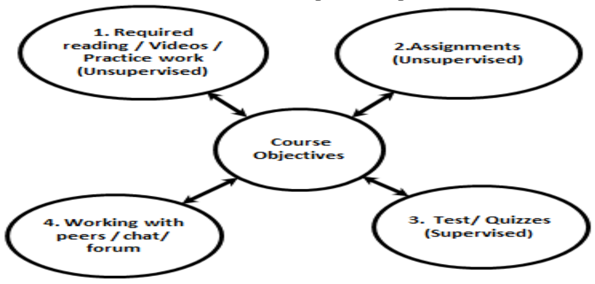

1. Required reading / Videos / Practice work (Unsupervised)
2.Assignments (Unsupervised)
Course Objectives
4. Working with peers / chat/ forum
3. Test/ Quizzes (Supervised)

Figure 2: Assessment instruments in courses on CLEW

3.2 Proposed algorithm Mine_Learning (ML):

This section first presents a formal algorithm (Mine_Learning (ML) for mining impact of assignments on student performance before providing more detailed description of each step.

```
Algorithm Mine_Learning (ML)
Input: an excel file with student marks (individual assignments, average assignment, final, total), number of visits to CLEW
Output: association rules generated for the input dataset, a model predicting if a student can pass a course given assignment and final marks
  1.  Prepare and clean the student marks.
  2.  Transform input data into sets of items to get marks_items.
  3.  Call the Apriori algorithm with marks_items to get marks Frequent patterns and Marks association rules.
  4.  Interpret the importance of generated mark association rules using confidence and support.
  5.  Apply SVM to define the Pass/Fail classification model
```

Figure 3 : Proposed Mine_Learning algorithm

Step 1: Data Cleaning, preparation and preprocessing of marks: Missing marks were replaced by zeroes. A row with 0 as the final exam mark (possibly because the student did not write the final exam) was deleted. Attributes such as ID that were of no significance to this study were removed. All marks were converted to percentages (out of 100) to be consistent.

Step2: Data transformation: Each row of input data is transformed into a set of items as a preparatory step for the Apriori algorithm. For example, let's assume that a student achieves {85, 80, 78} respectively as average assignment, final exam and overall marks. These marks are converted into items {R1, R5, R10} based on the rules defined in table 1. Table 1 defines rules R1 – R4 for assignment marks, R5 – R8 for final exam, R9 – R12 for overall marks and R13 – R15 for number of visits made to CLEW. Rules 16 onwards for individual assignments use the same categories as assignment_average and are not shown here. The transformed data as shown in table 2 has attributes TID (transaction Id), Num_att (number of attributes), RuleAA (Assignment_average transformed as 1 - 4), RuleFM (Final_exam transformed as 5 - 8), RuleTM (Overall_mark transformed as 9 -12) and RuleV (Number of visits as 13 - 15). 'R' has been removed from the transformed items for convenience.

Table 1: Rules used to transform marks to items

Rule Identifier	Rule
R1	**Assignment_average> 85**
R2	**70 <Assignment_average<= 85**
R3	**50 <= Assignment_average<= 70**
R4	**Assignment_average< 50**
R5	**Final_exam_mark> 85**
R6	**70 <Final_exam_mark<= 85**
R7	**50 <= Final_exam_mark<= 70**
R8	**Final_exam_mark< 50**
R9	**Overall_mark> 85**
R10	**70 <Overall_mark<= 85**
R11	**50 <= Overall_mark<= 70**
R12	**Overall_mark< 50**
R13	**1 <Number_of_visits<=100**
R14	**100 <Number_of_visits<=200**
R15	**Number_of_visits> 200**

Table 2: Output of step 2

TID	Num_att	RuleAA	RuleFM	RuleTM	RuleV	RuleA1	RuleA2	RuleA3	RuleA4	RuleA5	RuleA6
1	10	2	7	11	16	20	25	31	33	36	40
2	10	1	7	10	16	20	24	30	32	37	40
3	10	4	7	11	16	22	25	30	33	39	43
4	10	2	7	10	16	20	25	31	33	36	41
5	10	3	7	11	17	23	26	30	33	36	43
6	10	4	6	11	16	22	27	31	35	37	43
7	10	3	7	11	16	22	25	31	35	36	41
8	10	2	7	10	16	20	25	31	33	36	40
9	10	2	7	10	16	20	26	29	33	36	41
10	10	1	7	10	17	20	24	28	32	37	40
.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....	.....

Step3: Apply Apriori algorithm to the dataset obtained as output of step 2 to generate rules such as R4 =>R 8, R4 =>R12, R6 =>R1 and so on, as shown in table 3.

Step4: The rules generated in step 3 are then manually interpreted using the rules' confidence and support values to answer questions such as 'Does average assignment mark have a favorable impact on Final / Overall marks?' or 'How important is it for students to visit the CLEW site frequently?'.

Step 5: An SVM model is created to classify student data (from step 1) as Pass / Fail based on total marks. A total mark of >=50 is labeled as Pass; < 50 is labeled as Fail.

Table 3:Sample rules generated

Rule	Antecedent	Consequent	Support	Confidence
1	Assignment_average < 50	Final < 50	24	63
2	Assignment_average < 50	Total < 50	30	79
3	70 < Final <= 85	Assignment_average>85	16	56

4. RESULTS

Experiments were conducted on student data of two semesters in two Computer Science courses: 'Programming in C for Beginners'(code 106F and 106W) and 'Key concepts for end-users' (code 104F). A relative weight of 30% was assigned to assignments in 106F, 10% in 106W and 50% in 104F.Threshold used for support were 15 and 20 and for confidence was 50 since experiments indicated that lowering minimum support and confidence values increased the number of rules generated substantially. An analysis of all rules generated asserts our hypothesis that assignment marks have an impact on the student's overall and final marks. The confidence of such rules for course 106F is much higher compared to 106W, which implies that allocating a 30% weight to the assignments (as in 106F) has a higher impact than 10 % (106W). Similarly, rules generated for the last 5 assignments in 106F had 100% confidence in depicting that a student who scores > 85 in assignment also scores > 85 in final exam and in the overall mark, and a similar trend was observed with assignment marks <50. Some rare rules such as one found in the dataset 106W (Assignment1 > 85 => Final <50) can be attributed to the fact that assignment1, being the first one, was either too simple or marking was too lenient. Rules generated for all assignments of 104S are uniformly indicative of the fact that achieving a score of > 85 in the assignment and final exam ascertains a total mark in the range of >=70 and < 85. SVM model for all the three courses using average assignment marks predicted a student's chance of passing the course with more than 95% accuracy. However, accuracy using individual assignment marks was 81%, 87% and 97% for 106W, 106F and 104S respectively.

5. CONCLUSIONS

This research presents how useful the association rules mined using Apriori algorithm on student data are in extracting hidden patterns about course assessment instruments such as assignments and final exam. Teachers can take informed decisions using such patterns and use them in improving their curriculum and strategy of teaching a class. The assertion that assignment marks have a direct correlation with final exam and overall marks can be a motivating factor for them to perform well in the assignments.

6. FUTURE WORK

Moving forward, we propose to mine more student attributes that could impact their learning such as their personalities and meta-cognitive skills. Other instruments such as social aspects using chat rooms, discussions and forums that are prevalent in today's web-based courses can also be mined to study their impact on learning.

7. REFERENCES

[1] Anwar, M., and Ahmed, N. Knowledge mining in supervised and unsupervised assessment of students performance. In 2nd International Conference on Networking and Information Technology, 2011, 29-36.

[2] Minaei-Bidgoli, B. Data mining for a web-based educational system. PhD thesis, Michigan State University, 2004.

Mining Users' Behaviors in Intelligent Educational Games Prime Climb a Case Study

Alireza Davoodi
Department of Computer Science
The University of British Columbia
2366 Main Mall, Vancouver, BC, V6T1Z4, Canada
+1 (604) 8225108
davoodi@cs.ubc.ca

Samad Kardan
Department of Computer Science
The University of British Columbia
2366 Main Mall, Vancouver, BC, V6T1Z4, Canada
+1 (604) 8225108
skardan@cs.ubc.ca

Cristina Conati
Department of Computer Science
The University of British Columbia
2366 Main Mall, Vancouver, BC, V6T1Z4, Canada
+1 (604) 8224632
conati@cs.ubc.ca

ABSTRACT
This paper presents work on applying clustering and association rule mining techniques to mine users' behavior in interacting with an intelligent educational game, Prime Climb. Through such behavior discovery, frequent patterns of interactions which characterize different groups of students with similar interaction styles are identified. The relation between the extracted patterns and the average domain knowledge of students in each group is investigated. The results show that the students with significantly higher prior knowledge about the domain behave differently from those with lower prior knowledge as they play the game and that pattern could be identified early during the interactions.

Keywords
Intelligent Educational Games, Behavior Discovery, Association Rule Mining

1. INTRODUCTION
Many Adaptive educational systems apply data mining techniques to answer the need for understanding and supporting varying learning styles, capabilities and preferences in students[1, 2, 3]. Along this line of research, we concentrate on understanding how students interact with Prime Climb (PC) as an adaptive educational game and whether there is a connection between behavioral patterns and attributes (for instance higher knowledgeable vs. lower knowledgeable students) in the students. Developing an interactive environment in which more number of students can learn the desired skills requires a pedagogical agent which maintains more accurate understanding of individual differences between users and provides more tailored interventions. For instance, if a pedagogical agent is capable of identifying whether a group of students have higher domain knowledge than the other group, it can be possible to leverage such information to construct a more accurate user model and intervention mechanism.

Behavioral discovery has been vastly used in educational systems but there is limited application in educational games like PC in which educational concepts are embedded in the game with minimum technical notation to maximize game aspects (i.e. engagement) of the system. In PC, students follow an exploratory mechanism to explore and understand the methods and practice them. This paper describes the first step toward leveraging students' behavioral patterns into building more effective adaptive edu-game. The ultimate goal is devising mechanisms for making abstract high level meaning from raw interaction data and leveraging such understanding for real-time identification of characterizing interaction styles to enhance user modeling and intervention mechanism in an edu-game like Prime Climb.

2. Prime Climb Intelligent Edu-game
Prime Climb (PC) is an intelligent educational game for students in grades 5 and 6 to practice number factorization skills. In PC, the player and his/her partner climb a series of 11 mountains of numbers by pairing up the numbers which do not share a common factor [4]. There are two main interactions of a player with PC:
Making Movements**:** A player makes one or more movements at each time, by clicking on numbered hexagons on the mountains.
Using Magnifying Glass Tool**:** The magnifying glass (MG) tool is always available for the user to benefit from. The MG is used to show the factor tree of a number on the mountains; it is located in the top right corner of the game.

3. Data Collection/User Representation
For behavior discovery, we used the student's interaction data with the first 9 mountains of 43 students who completed at least 9 levels (mountains) of Prime Climb. Each user is represented by a vector of features. Some of the features are shown in Table 1. Each feature is a measure computed based on user's interactions with one or more mountains. In this paper we provide the results for two feature sets.

Mountains-Generic-Movement(1-9) features: Contains features calculated based on the users' movements behavior on mountains 1 to 9. A "*mountain-generic*" feature is a feature which is calculated across all mountains not individual mountains.
Mountains-Generic+Specific-MG+Movement(1-4) features: Contains features calculated base on user's movement and MG-usage behaviors on mountains 1 to 4. A "*mountains-specific*" feature is measured based on data from an individual mountain.

Table 1: Some Features used for behavior discovery

Movement Features
Sum/Mean/STD number of correct/wrong moves across mountains
Sum/Mean/STD of time on [correct/wrong] moves across mountains
Mean/STD length of sequence of correct/wrong moves
Mean/STD time spend per sequence of correct/wrong moves
Magnifying Glass (MG) Features
Sum/Mean/STD of MG Usage
Mean/STD number of [correct/wrong] movements per each MG usage

4. Clustering and Association Rule Mining
Prior to performing clustering, feature selection mechanism is applied to filter out irrelevant features [5]. Then, the optimal number of clusters is determined as the lowest number suggested by C-index, Calinski and Harabasz[4] and Silhouette [6] measures of clustering validity. Next, the GA *K*-means (*K*-means for short) clustering algorithm [1], which is a modified version of GA *K*-means [7], is applied to cluster the users into an optimal number of clusters.

Table 2: Extracted Rules for Mountains-Generic-Movement(1-9)

Rules for Cluster 1[HPK]: (Size: 10/43 = 23.26%)
Mean-Time-on-Movements(1-9) = Higher, [6/6=100%]
Mean-Time-Spent-On-Correct-Movements-On-Mountains(1-9) = Higher, ([5/5=100%])
Rules for Cluster 2[LPK]: (Size: 33/43 = 76.74%)
Mean-Time-On-Movements(1-9) = Lower, [33/37=89.19%]
○ STD-Time-On-Wrong-Correct-Moves(1-9) = Lower, [33/35=94.29%]
Mean-Time-On-Consecutive-Wrong-Movements(1-9) = Lower, [31/35=88.57%]
○ STD-Time-On-Movements(1-9) = Lower, [31/33=93.94%]
○ STD-Time-On-Correct-Movements(1-9) = Lower, [31/33=93.94]

The Hotspot algorithm is used to extract the rules for each discovered cluster. The clusters are then compared for statistical difference on a measure called *cluster's prior knowledge*:

$$\boldsymbol{Cluster's\ prior\ knowledge} = \frac{\sum_{student \in cluster} \boldsymbol{pre\_test(student)}}{\boldsymbol{Cluster's\ size}}$$

$pre\_test(student)$ is the student's score on a pre-test taken before playing PC. The max, average and standard deviation of the scores across the students are 15, 11.7 and 3.29 respectively.

Behavior Discovery on Mountains-Generic-Movement(1-9) Set: Feature selection mechanism selected 18 features out of original 30 features. The optimal number of clusters was found to be 2. The result of a t-test showed that there is a statistically significant difference between the prior knowledge of cluster 1 of students (*higher prior knowledge (HPK) group*) (*M*=13.0 , *SD*=2.0) and cluster 2 of students (*lower prior knowledge (LPK) group*) (*M*=11.3 , *SD*=3.45), *p*=.03 and *cohen-d*=.53. Table 2 shows the rules extracted for each cluster using the Hotspot algorithm. Each bulleted item in Tables 2 and 3 shows an extracted rule. "Higher" and "Lower" are the *bin*s. We considered two bins in this study. The bin shows whether the value of the feature is located in the higher or lower portion of the feature values across students. The cut-off point for splitting a range of feature's values to 2 ranges of lower and upper ranges is calculated specifically for the feature in each extracted rule by the Hotspot algorithm. In front of each rule is a fraction whose numerator and denominator respectively shows the number of students in the cluster and total students on which the rule applies.

The extracted rules show that the students belonging to the HPK cluster, spent more time on movements and correct movements across 9 mountains. This could indicate that the HPK students were more involved in the game and spent more time before making a movement. In contrast, the group of LPK students spent lower time on making movements as well as wrong movements. This could be an indication of less involvement in the game by the LPK group. The other patterns show a lower standard deviation on time spent on making movements and correct movements for LPK group. This indicates that this group of students showed a consistent pattern of lack of engagement in the game.

Behavior Discovery on Mountains-Generic+Specific-MG+Movements(1-4) Set: This feature set only employs interaction data from the first 4 mountains. Such feature set is mainly valuable for constructing an online classifier to classify students to different classes based on their interaction with the game during the gameplay. Table 3 shows the discovered clusters and extracted rules. The result of the t-test shows a statistically significant difference between cluster1's *prior knowledge* (*M*=13.28, *SD*=1.58) and cluster2's *prior knowledge* (*M*=11.39 , *SD*=3.4), *p*=.02, *cohen-d*=.60. Also, around 16% of students belong to HPK cluster and 84% belong to the LPK group.

Table 3: Extracted Rules for Mountains-Generic+Specific-MG+Movements(1-4)

Cluster 1[HPK]: (Size: 7/43 = 16.28)
Mean-Time-On-Movements(4) = Higher, (100% [5/5])
Mean-Time-On-Correct-Movements(3) = Higher, (100% [3/3])
Cluster 2[LPK]: (Size: 36/43 = 83.72%)
Mean-Time-On-Correct-Movements(1-4) = Lower, (100% [35/35])
Mean-Time-On-Movements(1-4)=Lower, (100% [34/34])

This result is very similar to the results when data from all 9 mountains is included. Similar patterns can be seen when more interaction data from upper mountains is included in patterns analysis.

5. CONCLUSION/FUTURE WORK

This paper discusses behavior discovery in PC. To this end, different sets of features were defined. The features were extracted from interaction of students with PC in the form of making movements from one numbered hexagon to another numbered hexagon and usages of the MG tool. In order to identify frequent patterns of interaction in groups of students, firstly a feature selection mechanism was applied to select more relevant features from set of all features. Then a K-Means clustering was applied to cluster the students into optimal number of clusters. Once clusters were built, the Hotspot algorithm of Association Rule Mining is applied on the clusters to extract frequent interaction patterns. Finally the clusters were compared to each other on their cluster's *prior knowledge*. When interaction data from all 9 mountains is included in behavior discovery, it was found that the students with higher prior knowledge were more engaged in the game and spent more time on making movements. On the contrary, the students with lower prior knowledge, spent less time on making movements, indicating that they were less involved in the game. Behavior discovery also was conducted on truncated sets of features in which only a fraction of interaction data was included. The results showed that using the interaction data from the first four mountains resulted in groups of students that are statistically different on their prior knowledge.

As for future work, an online classifier will be built which identifies frequent patterns of interaction in the students and classify them into different groups in real time and leverages such information to build a more personalized user model and adaptive intervention mechanism in PC.

6. REFERENCES

[1] Kardan, S., and C. Conati., 2011, A Framework for Capturing Distinguishing User Interaction Behaviours in Novel Interfaces. EDM 2011, Netherlands, 159-168

[2] Judi, M., Baldwin, J., 2012, Identifying Successful Learners from Interaction Behaviour, EDM 2012, Greece, 160-163

[3] Lopez, M. I., Romero, C., Ventura, S., Luna, J.M., 2012, Classification via clustering for predicting final marks starting from the student, participation in Forums, EDM 2012, Greece, 148-152

[4] Manske, M. , Conati, C., 2005, Modelling Learning in an Educational Game. AIED 2005, 411-418

[5] Guyon, I., & Elisseeff, A. 2003. An introduction to variable and feature selection. *J. of Machine Learning Research*, *3*, 1157–1182.

[6] Milligan, G. W., & Cooper, M. C. 1985. An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, *50*(2), 159–179.

[7] Kim, K. Ahn, H. 2008. A recommender system using GA k-means clustering in an online shopping market. Expert Syst. Appl. 34, 2, 1200-1209.

Bringing student backgrounds online: MOOC user demographics, site usage, and online learning

Jennifer DeBoer[1]
jdeboer@mit.edu

Glenda S. Stump[1]
gsstump@mit.edu

Daniel Seaton[1]
dseaton@mit.edu

Andrew Ho[2]
Andrew_Ho@gse.harvard.edu

David E. Pritchard[1]
dpritch@mit.edu

Lori Breslow[1]
lrb@mit.edu

[1]Massachusetts Institute of Technology
77 Massachusetts Avenue
Cambridge, MA 02139

[2]Harvard Graduate School of Education
455 Gutman Library, 6 Appian Way
Cambridge, MA 02138

ABSTRACT

MOOCs gather a rich array of click-stream information from students who interact with the platform. However, without student background information, inferences do not take advantage of a deeper understanding of students' prior experiences, motivation, and home environment. In this poster, we investigate the predictive power of student background factors as well as student experiences with learning materials provided in the first MITx course, "Circuits and Electronics." We focus on a group of survey completers who were given background questions, and we use multiple regression methods to investigate the relationship between achievement, online resource use, and student background. Online course providers may be able to better tailor online experiences to students when they know how background characteristics mediate the online experience.

Keywords

MOOCs, social capital, regression, online learning

1. RATIONALE

Massive Open Online Courses (MOOCs) enroll thousands of students through numerous learning platforms. edX, which offered its first class "Circuits and Electronics" (6.002x), in the spring of 2012, drew students from nearly every country in the world [5]. The courses' free, open-enrollment structure appeals to a diverse set of students who can use learning resources as they please. But while on-campus instructors have the opportunity to learn about their students—their experience growing up, their educational background, etc.—as they interact with them in the classroom, MOOC instructors do not have that luxury.

MOOC students, however, are unique individuals. They are motivated by different incentives to sign up for the course, they come from different home environments, and they speak different languages. As part of a study on 6.002x, we gathered additional background information on students in the first edX class in order to understand the simultaneous impact of time spent on different course components and a student's background characteristics.

2. RESEARCH QUESTION

To provide a more complete picture of the student factors that relate to achievement, we ask, "What student factors predict higher achievement, all else equal?" In other work, we have explored the impact of resource use (e.g., watching videos, reading the textbook) on achievement in our analyses, but here we investigate whether the inclusion of demographic background factors as covariates changes the relationship between resource use and achievement.

3. CONCEPTUAL FRAMEWORK

We apply a social capital lens to our work. While the inputs of formal schooling are important, education researchers have also noted the importance of the acculturation and social preparation to which students are differentially exposed prior to entering school [see, for example, 3, 6]. Once students enroll, the norms and behaviors they understand as beneficial may be differentially rewarded within the school system [1, 8]. The tools that serve more privileged groups of students in traditional settings (e.g., linguistic capital, knowledge of cultural references, highly educated role models) may also be relevant for online learning.

Distance learning classes have largely been characterized by learners looking for flexibility, cost-savings, and a familiar electronic platform [9, 5]. A larger portion of these students has been female, and many of them have been "non-traditional" students. While this suggests the online context may better *support* students who are underserved in traditional STEM classrooms, initial results from more recent online learning programs suggest there are no differential gains for underserved subgroups [2]. In fact, another study notes that the "achievement gap" between traditionally higher- and lower-performing students may actually be widened due to an online course experience [10].

4. DATA

Data come from the students in 6.002x who completed the exit survey. While the survey was announced specifically for course completers, the link to the survey was open on the website. We find ~800 completers did not receive a certificate in the course. It is important to note these data were gathered using matrix sampling to mitigate non-response due to survey fatigue. We impute missing responses using chained equations [7]. Missingness ranges from 59% to 85% of the over 7,000 students who completed the survey.

Figure 1 illustrates the spread of total points awarded to survey completers using a "partial credit" model. In this model, the number of points students received for a right answer was dependent upon the number of attempts they made. (Students were allowed unlimited attempts to answer homework and lab questions, but only three attempts on the midterm or final.) The bimodal distribution illustrates the "two-population" nature of our sampling frame: some students who earned a certificate, and others who followed the course but did not earn a certificate.

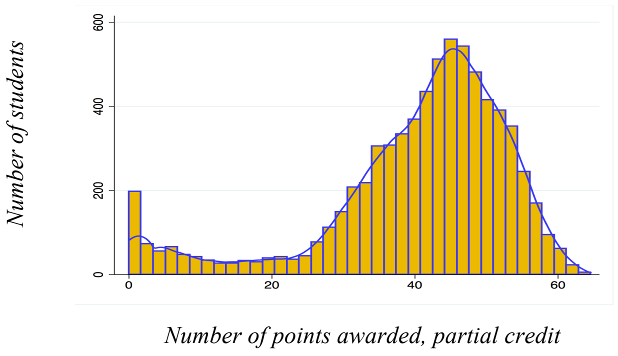


Figure 1.Survey completer numbers across performance levels

5. RESULTS

For the subset of students completing surveys, once we control for key student background information, we immediately find the impact of certain resources on students' total score to be diminished. In addition, when we add controls such as score on the first homework (proxy for prior ability), and when we control for the students' country, we find that we remove more bias. This may result, for example, from correlation between initial score and subsequent study strategies such as referencing previous homeworks or viewing relevant questions on the discussion forum.

Table 1. Additive models predicting partial credit score, OLS[1]

Covariate	"Naïve" estimate	Control for first HW	Controls for country (not listed)
Homework	4.41 (0.28) ***	3.70 (0.26) ***	3.71 (0.27) ***
Labs	0.61 (0.34)	0.45 (0.32)	0.52 (0.32)
Lecture problems	0.26 (0.10) **	0.09 (0.09)	0.10 (0.09)
Lecture videos	0.38 (0.13) **	0.17 (0.13)	0.09 (0.13)
Tutorials	0.11 (0.06)	0.09 (0.06)	0.07 (0.06)
Book	-0.26 (0.07) ***	-0.23 (0.07) **	-0.24 (0.07) ***
Wiki	-0.64 (0.07) ***	-0.60 (0.07) ***	-0.58 (0.07) ***
Discussion board	0.30 (0.12) **	0.39 (0.11) ***	0.33 (0.11) **
Female	-1.11(1.80)	-1.20 (1.88)	-1.12 (1.88)
Parent engineer	2.16 (0.81) **	1.86 (0.79) **	1.91 (0.79) **
Worked with other offline	2.05 (0.66) **	1.96 (0.65) **	2.11 (0.71) **
Teach EE	-0.01(0.56)	0.08 (0.58)	0.26 (0.58)
Took diff. equations	4.72 (0.56) ***	4.44 (0.48) ***	4.56 (0.55) ***
First HW		0.57 (0.03) ***	0.55 (0.03) ***

5.1 Course resources

For survey completers, time spent on homework was a consistent significant predictor of a higher overall score. Even controlling for initial performance, spending more time on the homework was related to gains of approximately 1/3 of a standard deviation on the total points for the class (a small-to-medium effect size in educational research). More time spent on the discussion board was also related to a higher score. However, more time spent on the book or the course wiki was related to lower achievement.

5.2 Initial score

We construct a control covariate of "firstpoints," a rough proxy for initial proficiency with material relevant to the course. In the second and third models given above (additive), the inclusion of this control alters the significance of the covariates for time spent on lecture problems and time spent on lecture videos. This may indicate that students who come into the course with different abilities use these resources in different ways. However, including this control does not change which demographics are significant.

5.3 Demographics

The impact of key demographic background factors is consistent across models, including the fully specified model with all covariates, which allows for a fixed-effect for the student's country of access. Individual factors such as gender and whether the student teaches electrical engineering are not related to achievement. On the other hand, some background factors are strongly related to performance. Specifically, having taken differential equations predicted a higher score, even controlling for the first assignment. Similarly, students who reported offline collaboration also scored higher. This might reflect the same positive role of collaboration as participation in the discussion forum.

6. DISCUSSION AND IMPLICATIONS

The inclusion of demographic variables for MOOC users adds significant, practically important covariates to predictions of achievement based on individual information. While online course creators may already have a wealth of student data from click-stream information, solely predicting performance based on observed behaviors misses important explanatory factors and a deeper understanding of why students may behave in different ways or experience differential utility of online resources. As MOOC offerings grow, course designers may further study how to tailor the online experience and support the diverse backgrounds of a world of students.

7. ACKNOWLEDGMENTS

Funding was provided by NSF Grant, DRL-1258448. We also acknowledge the support of MIT's RELATE group and edX.

8. REFERENCES

[1] Anyon, J. 2008. *Social Class and School Knowledge,* in The Way Class Works, L. Weis (ed). Routledge, chap. 13.

[2] Bowen, W.G., Chingos, M.M., Lack, K.A., & Nygren, T.I. 2012. Interactive Learning Online at Public Universities: Evidence from Randomized Trials, ITHAKA, May 22, 2012.

[3] Bourdieu, P. 1977. *Cultural Reproduction and Social Reproduction,* in Power and Ideology, J. Karabel & A.H. Halsey (eds), Oxford Press, Chap. 29.

[4] Dziuban, C., Moskal, P., Brophy, J., Shea, P. 2007. Student satisfaction with asynchronous learning, *Journal of Asynchronous Learning Networks, 11*(1), 87-95.

[5] edX. 2012. Frequently asked Questions. F*rom: [edx.org/faq]*.

[6] Lareau, A. 2003. Unequal Childhoods: Class, Race, and Family Life. University of California Press.

[7] Little, R.J.A. & Rubin, D.B. 2003. Statistical Analysis with Missing Data. *Technometrics*, 45 (4), pp. 364-365.

[8] Meyer, J. 1977. The Effects of Education as an Institution, *American Journal of Sociology*, 83(1): 55-76.

[9] Park, J.-H., & Choi, H. J. 2009. Factors influencing adult learners' decision to drop out or persist in online learning. *Educational Technology & Society, 12*(4), 207-217.

[10] Xu, D. & Jaggars, S.S. 2013. *Adaptability to Online Learning: Differences Across Types of Students and Academic Subject Areas.* CCRC, Working paper No 5.

[1]Note: resource use covariates given in log-seconds. As noted above, partial credit is given for multiple attempts at a question, though results are consistent when full points were awarded regardless of the number of attempts. This was the policy used by the instructors in determining grades.
** p <0.05, ** p<0.01, ***, p<0.001*

Detecting Player Goals from Game Log Files

Kristen E. DiCerbo
Pearson

kristen.dicerbo@pearson.com

Khusro Kidwai
University of Southern Maine

khusro.kidwai@maine.edu

ABSTRACT

In this paper we describe the development of a detector of *seriousness of pursuit* of a particular goal in a digital game. As gaming researchers attempt to make inferences about player characteristics from their actions in open-ended gaming environments, understanding game players' goals can help provide an interpretive lens for those actions. This research uses Classification and Regression Tree methodology to develop and then cross-validate features of game play and related rules through which player behavior about pursuing a goal of completing a quest can be classified as serious or not serious.

Keywords

Detector, goal, stealth assessment, games, educational data mining.

1. INTRODUCTION

Many recently-developed online learning environments provide open spaces for students to explore. At the same time, there is growing interest in stealth assessment [5], or the use of data resulting from students' every day interactions in a digital environment to make inferences about player characteristics.

This use of data from natural activity in open-ended environments presents a challenge for interpretation. Much of the evidence we wish to use to assess skill proficiency and player attributes assumes that individuals are working towards the goal of completion of sub-tasks or levels within a game. However, game players often appear to be pursuing differing goals [2], which provide different lenses for interpreting player behavior based on data in log files for games. For example, behavior might be categorized as "off-task" if a player is pursuing a quest but "on-task" if a player has a goal of exploring the environment. If we are interested in using evidence contained in game log files to assess constructs such as persistence, we have to be careful not to identify a player as lacking persistence when in fact they were very persistently pursuing a different goal.

This paper describes the creation of a detector for a specific goal—serious pursuit of completion of quests in a game. The approach taken in this paper builds on research regarding detectors for gaming the system [1], which use machine learning to identify features and rules to classify behavior into discrete categories. In this paper, the focus was on whether a player in the online game Poptropica® is seriously pursuing the goal of completing quests in the game. The ability to successfully categorize players based on whether or not they are pursuing the goal of quest completion is likely to help with interpretation of other actions players pursue in the game. This paper discusses the selection of possible features or indicators of goal seriousness, the process of detector creation, and the analysis of the effectiveness of the detector in correctly classifying play.

2. DESCRIPTION OF THE ENVIRONMENT

Poptropica® is a virtual world in which players explore "islands" with various themes and overarching quests that players can choose to pursue. Players choose which islands to visit and the quests generally involve completion of 25 or more steps (for example, collecting and using assets) that are usually completed in a particular order. Apart from the quests, players can talk to other players in highly scripted chats (players can only select from a pre-determined set of statements to make in the chat sessions), play arcade-style games head-to-head, and spend time creating and modifying their avatar.

Like with most online gaming environments, the Poptropica® gaming engine captures time-stamped event data for each player. On an average day actions of over 350,000 Poptropica® players generate 80 million event lines.

3. DETECTOR DEVELOPMENT

Prior to building a machine detector of goal-seriousness, it was necessary to establish a human-coded standard from which the computer could learn and verify rules. A total of 527 clips were coded by two raters as being either "serious" or "not serious" about the goal of completing a quest. Cohen's Kappa [3] between the two raters for the full set of non-training clips was .72; all disagreements were discussed until accord was reached.

Elements of the log files hypothesized to be indicative of goal directedness were identified as features including: (1) total number of events completed on the island, (2) total amount of time spent on the island, (3) total number of events related to quest-completion, (4) number of locations (scenes) visited on the island, (5) number of costumes tried on, and (6) number of inventory checks. The number of costumes and number of inventory checks were hypothesized to be negatively correlated to completing quests.

Researchers employed a Classification and Regression Tree (CART) methodology to create the detector. The process of the creation of decision trees begins with the attempt to create classification rules until the data has been categorized as close to perfectly as possible, however, this can result in overfit to the training data. The software then tries to "relax" these rules, in a process called "pruning" to balance accuracy and flexibility to new data. This research employed the J48 algorithm [4] for pruning. The results of the analyses were evaluated using (1) precision, (2) recall, (3) Cohen's kappa, and (4) A'.

4. RESULTS

The final decision tree is displayed in Figure 1. Each branch provides classification rules and an ultimate classification decision at the end. The red boxes end paths that indicate individuals non-serious about the goal of quest completion while the green boxes indicate serious goal directed behavior. So, for example, following the left-most path, we find that people who visit 4 or fewer scenes and complete 2 or fewer quest events were classified as not seriously goal-directed. This rule correctly classified 335 clips and misclassified 18 of the 353 total clips that followed this pattern. Following other branches reveals different rules, all leading to classifications of seriously or not seriously goal-directed.

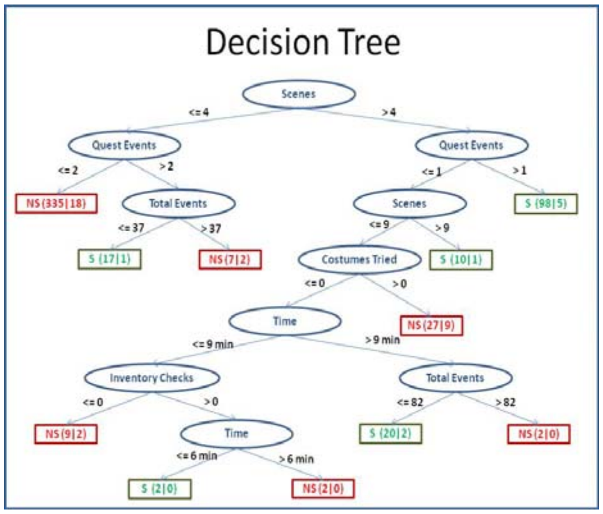


Figure 1. Final Decision Tree

Cross-validation was completed by having half the sample serve as the training sample and the other half as the test sample, and then switching the halves. The detector achieved good performance under cross-validation. Human raters identified 167 of the 527 clips as indicating serious goal-directed behavior. The detector identified 151 clips as serious, out of which 119 agreed with the human raters and 32 did not (see Table 1). This resulted in a precision score of .79 and a recall score of .71. The Kappa value was .63, indicating that the accuracy of the detector was 63% better than chance. The A' was .93, indicating that the detector could correctly classify whether a clip contained serious goal-directed behavior 93% of the time.

Table 1. Correctness of Detector Classification

		Detector	
		Not Serious	Serious
Human	Not Serious	328	32
	Serious	48	119

In order to further investigate where the detector had difficulty with accuracy, we compared places where the detector disagreed with the raters to places where the raters had initially disagreed with each other. Although the human raters eventually came to an agreement about the classification of the clip, their initial disagreement was likely an indication of an ambiguous clip. As displayed in Table 2, when the human raters agreed, the detector also agreed with them 89% of the time. However, when the human raters disagreed, the detector disagreed with their final rating 49% of the time.

Table 2. Cross-tabulation of computer and human agreement

	Humans Disagree	**Humans Agree**
Computer/ Humans Disagree	49.15% (N=29)	10.90% (N=51)
Computer/ Humans Agree	50.85% (N=30)	89.10% (N=417)

5. DISCUSSION AND CONCLUSIONS

The purpose of this paper was to describe the creation of a detector of seriousness of goal pursuit in an online game. The goal a player is pursuing in a game or other open-ended online environment can help provide context and important interpretation of player actions within the system. The results here suggest that an automated detector can be created that can reliably identify whether a participant is seriously pursuing a goal of completing a game quest. The methodology discussed in this paper opens up the possibility of gaming engines detecting and prompting players to adjust their approach (for example, becoming more goal directed) in real time.

We suggest that this type of analysis of player goals may have advantages over other attempts to measure and assess constructs based on player behavior captured in log files. Algorithms such as the one discussed in this paper allow for efficient categorization of thousands of players and millions of actions that would not be humanly feasible. In other games, a similar process could be followed by 1) identifying potential goals, 2) identifying potential indicators of those goals, 3) hand coding a small set of log files as pursuing on of those goals or not, and 4) carrying out the Classification and Regression Tree analysis. The identification of player goals can help us understand player actions in games and extend our ability to make inferences about player characteristics.

6. ACKNOWLEDGMENTS

Our thanks to Ryan Baker and Ilya Golden who served as mentors during this project's inception at the Pittsburgh Science of Learning Center LearnLab Summer School 2012.

7. REFERENCES

[1] Baker, R.S.J.d., Corbett, A.T., Koedinger, K.R., and Roll, I. 2006. Generalizing detection of gaming the system across a tutoring curriculum. *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, 402-411.

[2] Bartle, R. 1996. Hearts, clubs, diamonds, spades: Players who suit MUDs. J.MUD Res, 1, 19.

[3] Cohen, J. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1), 37-46.

[4] Quinlan, R. 1993. C4.5 Programs for Machine Learning. Morgan Kaufman, San Mateo, CA.

[5] Shute, V. 2011. Stealth assessment in computer-based cames to support learning. In *Computer Games and Instruction*, S. Tobias & D. Fletcher, Eds. Information Age Publishers, Charlotte, NC, 503-524.

A Prediction Model Uses the Sequence of Attempts and Hints to Better Predict Knowledge: Better to Attempt the Problem First, Rather Than Ask for a Hint

Hien D. Duong Linglong Zhu Yutao Wang Neil T.Heffernan
Department of Computer Science
Worcester Polytechnic Institute
100 Institute Road, Worcester, MA
hdduong@wpi.edu lzhu@wpi.edu yutaowang@wpi.edu nth@wpi.edu

`

ABSTRACT

Intelligent Tutoring Systems (ITS) have been proven to be efficient in providing students assistance and assessing their performance when they do their homework. Many research projects have been done to analyze how students' knowledge grows and to predict their performance from within intelligent tutoring system. Most of them focus on using correctness of the previous question or the number of hints and attempts students need to predict their future performance, but ignore how they ask for hints and make attempts. In this paper, we build a Sequence of Actions (SOA) model taking advantage of the sequence of hints and attempts a student needed for previous question to predict students' performance. We used an ASSISTments dataset of 66 students answering a total of 34,973 problems generated from 5010 questions over the course of two years. The experimental results showed that the Sequence of Action model has reliable predictive accuracy than Knowledge Tracing.

Keywords

Knowledge Tracing, Educational Data Mining, Student Modeling, Sequence of Action model.

1. INTRODUCTION

Understanding student behavior is crucial for Intelligent Tutoring Systems (ITS) to improve and to provide better tutoring for students. For decades, researchers in ITS have been developing various methods of modeling student behavior using their performance as observations. One example is the Knowledge Tracing (KT) model (Corbett and Anderson, 1995), which uses a dynamic Bayesian network to model student learning. But KT focuses attention on students' performance of correctness, ignoring the process a student used to solve a problem. Many papers have shown the value of using the raw number of attempts and hints (Feng, Heffernan and Koedinger, 2009, Wang, Heffernan 2011). However, most EDM models we are aware of (with one notable exception of Ben Shih, et al. (2012)) have ignored the sequencing of action.

Consider a thought experiment. Suppose you know that Bob Smith asked for one of the three hints and makes one wrong answer before eventually getting the question correct. What if someone told you that Bob first made an attempt then had to ask for a hint compared to him first asking for a hint and then make a wrong attempt? Would this information add value to your ability to predict whether Bob will get the next question correct? We suspected that a student who first makes an attempt might be a better student.

In this work, we define a Sequence of Action (SOA) model that uses the information about the action sequence of attempts and hints for a student in previous question to better predict the correctness of next question. In SOA, students' sequences of actions are divided into five categories: One Attempt, All Attempts, All Hints, Alternative Attempt First and Alternative Hint First. The results of tabling methods indicate that it is better to attempt the problem first rather than ask for a hint. Another highlight of this paper is that we used the next question's percent correct from the tabling method as a continuous variable to fit a binary logistic regression model for SOA. The experimental results show that the SOA outperforms KT in all three metrics (MAE, RMSE, AUC).

2. Sequence of Action Model

2.1 Tabling Method

There are many different sequences of actions. Some students answered correctly only after one attempt and some students kept trying many times. Some students asked for hints and made attempts alternatively, which we believe that they were trying to learn by themselves. In the data, there are 217 different sequences of actions. We divided them into five bins: (1) One Attempt: the student correctly answered the question after one attempt; (2) All Attempts: the student made many attempts before finally get the question correct; (3) All Hints: the student only asked for hints without any attempts at all; (4) Alternative, Attempt First: the students asked for hints and made attempts alternatively and made an attempt at first; (5) Alternative, Hint First: the students asked for hint and made attempts alternatively and asked for a hint first.

We used 34,973 problem logs of sixty-six 12-14 year-old, 8th grade students participated in one class from ASSISTments, which is an online tutoring system giving tutorial assistance if a student makes a wrong attempt or asks for help. Questions in each problem set are generated randomly from several templates and there is no problem-selection algorithm used to choose the next question. Table 1 shows the sequence of action division and some examples in each category, and the correct percent of next question from tabling method (Wang, Pardos and Heffernan2011). Notice that each sequence ends with an attempt because in ASSISTments, a student cannot continue to next question unless he or she fills in the right answer of the current problem. In Table 1, 'a' stands for answer and 'h' stands for hint. For example, 'aha' indicates a student makes an attempt and then asks for a hint before finally types the right the answer.

Table 1. Sequence of Action Category and Examples

Sequence of Action Bin	Examples	Next Question Correct Percent
One Attempt (a)	a	0.8339
All Attempts (a+)	aa, aaa, …, aaaaaaaaaaaa	0.7655
All Hints (h+)	ha, hha,…, hhhhhhha	0.4723
Alternative, Attempt First (a-mix)	aha, aahaaha,…, aahhhhaaa	0.6343
Alternative, Hint First (h-mix)	haa, haha,…, hhhhaha	0.4615

From the tabling results, shown in Table 1, we can see that the percent of next-question-correct is highest among students only using one attempt since they master the skill the best. They can correctly answer the next question with the same skill. For students in All Attempts category, they are more self-learning oriented, they try to learn the skill by making attempts over and over again. So they get the second highest next-question-correct percent. But for students in the All Hints category, they do the homework only relying on the hints. It is reasonable that they don't master the skill well or they don't even want to learn, so their next-question-correct percent is very low. The alternative sequence of action reflects students' learning process. Intuitively, these students have positive attitude for study. They want to get some information from the hint based on which they try to solve the problem. But the results for the two alternative categories are very interesting. Though students in these two categories alternatively ask for hints and make attempts, the first action somewhat decided their learning altitude and final results. For students who make an attempt first, if they get the question wrong, they try to learn it by asking for hints. But for students who ask for a hint first, they seem to have less confidence in their knowledge. Although they also make some attempts, from the statistics of action sequence, they tend to ask for more hints than making attempts. The shortage of knowledge or the negative study attitude makes their performance as bad as the students asking exclusively for hints first.

2.2 SOA Binary Logistic Regression Model

In this section we build a logistic regression model based on sequence of action to better predict students' performance. In this model, we want to use students' current sequence of action to predict their performance on next question in same skill. The dependent variable is students' actual performance on a question, correct or incorrect, and the independent variables are categorical factor Skill_ID and continuous factor Next_Question_Correct_Percent from Table 1, which indicates the sequence of action of current question. For example, if sequence of action of current problem is "hhhhaha", we use 0.4615 its value. We equally split 66 students into six groups, 11 students in each, to run 6-fold cross validation. The SOA and KT model are trained on the data from every five groups and are tested on the sixth group.

Table2 shows experimental result of three metrics: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE) and Area Under ROC Curve (AUC). Lower values for MAE and RMSE indicate better model fit while higher values for AUC reflect a better fit. The values are calculated by averaging corresponding numbers obtained in each experiment of the 6-fold cross validation. The raw data and results for the six groups is available at this website:

(http://users.wpi.edu/~lzhu/SOA/DataSet_and_Results.rar).

Table 2. Prediction accuracy of KT, SOA and Ensemble

	MAE	RMSE	AUC
KT	0.3032	0.3921	0.6817
SOA	0.2900	0.3813	0.6841
t-test p value	0.0000	0.0000	0.5286

Although most numbers seem very close, SOA outperforms KT in all three metrics. To examine whether the difference were statistically reliable, we did a 2-tailed paired t-test based on the result from the cross validation. The last row in Table 2 shows that the differences are significant in both MAE and RMSE.

3. CONTRIBUTIONS

In this work, we presented a Sequence of Actions (SOA) model, in which students' action of asking for hints and making attempts are divided into five categories shown in Table 1. The result of a tabling method shows that students who make an attempt first did better on next question with the same skill than those who ask for a hint first. The result from logistic regression shows that paying attention to the sequence of action increases prediction accuracy of students' performance.

4. ACKNOWLEDGMENTS

This work is funding in part by IES and NSF grants to Professor Neil Heffernan. The first three authors have all been funded by the Computer Science Department at WPI.

5. REFERENCES

[1] Wang Q.Y., Pardos, Z. A., & Heffernan, N. T. (2011). *Tabling Method–A simple and practical complement to Knowledge Tracing*. KDD.

[2] Feng, M., Heffernan, N.T., &Koedinger, K.R. (2009). *Addressing the assessment challenge in an Online System that tutors as it assesses*. UMUAI: The Journal of Personalization Research19(3), 243-266.

[3] Corbett, A. T., & Anderson, J. R. (1995). *Knowledge tracing: modeling the acquisition of procedural knowledge*. User Modeling and User-Adapted Interaction, 4, 253–278.

[4] Shih, B., Koedinger, K.R., &Scheines, R. (2010). *Discovery of Learning Tactics using Hidden Markov Model Clustering*. Proceedings of the 3rd International Conference on EDM.

Towards the development of a classification service for predicting students' performance

Diego García-Saiz
Department of Mathematics, Statistics and Computation
University of Cantabria
Avda. Los Castros s/n, Santander, Spain
diego.garcia@unican.es

Marta Zorrilla
Department of Mathematics, Statistics and Computation
University of Cantabria
Avda. Los Castros s/n, Santander, Spain
marta.zorrilla@unican.es

ABSTRACT

Choosing a suitable classifier for a given data set is an important part of a data mining process. Since a large variety of classification algorithms are proposed in literature, non-experts, as teachers, do not know which method should be used in order to achieve a good pattern. Hence, a recommender service which guide on the process or automatize it is welcome. In this paper, we rely on meta-learning in order to predict the best algorithm for a data set given. More specifically, our work analyses what meta-features are more suitable for the problem of predicting student performance and also evaluates the viability of the recommender.

1. INTRODUCTION

One of the most important tasks in the process of knowledge discovery (KDD) is the selection of the algorithm that gets the best performance to solve a given problem. An approach based on meta-learning, able to automatically provide guidance on the best alternative from a set of meta-data, can be followed to achieve this goal.

We consider that this approach is suitable for educational context in which most teachers are not experts in data mining, but they do need to have objective information that allows them to enhance the teaching-learning process. Our ultimate goal is to automate the whole KDD process so that teachers should only be concerned to define the data set and a software service, supported by a recommender of algorithms, which generates the most accurate classification model based on the more relevant features of the data set.

In our work, we understand *meta-learning* as the automatic process of generating knowledge that relates the performance of machine learning algorithms to the characteristics of the data sets. We propose a number of features and discard others for their use in educational field. In our case study we generated 81 data sets from 2 virtual courses taught in the University of Cantabria and build over 700 classifiers using twelve different classification algorithms. Then we created three meta-data sets with the intrinsic characteristics extracted from each original data sets and define the algorithm with higher accuracy as class attribute. One of these data sets was used to generate our recommender.

There are different approaches about what features can be used as meta-data. In most cases measurable properties of data sets and algorithms are chosen. For instance, some authors [3] utilize general, statistical and information-theoretical measuresextracted from data sets whereas others as use landmarkers as in [4].

This paper is not intended to design a database to store data mining processes as there is already one available [1], but its main aim is to assess the feasibility of our proposal and propose a set of measurable features on educational data sets which can help us to choose automatically the classification algorithm with certain reliability.

We must also mention several research projects have targeted meta-learning in recent years, as e-LICO project [2].

2. EXPERIMENTATION

In our experiments, we used data from 2 virtual courses: a multimedia course taught during three academic years (2008-2010) hosted in Blackboard and a programming course taught in 2009 hosted in Moodle. All data sets gather the activity performed by learners in each course with their corresponding numeric mark.

In order to have enough data sets for our experimentation, we generated 81 data sets from them. First we created 3 data sets with data from multimedia course establishing the class attribute with values pass or fail, and another one as the union of these three. The same process was carried out with the programming course. Next, we generated 4 discretized data sets from the previous bi-class data sets using PKIDiscretize from Weka, and 4 data sets more but these partially discretized. Besides, we created two data sets with 4 classes (fail, pass, good, excellent) and one with 5 classes (drop-out, fail, pass, good, and excellent).

Next, we generated 60 data sets by adding to all original data sets a 10, 20, 30 and 40% of missing values. And finally, we created 4 data sets more by applying SMOTE algorithm on 2 of our original data sets with the following proportion of balancing class: 80-20%, 85-15%, 70-10% and 90-10%.

Models generation was performed by applying 12 classification algorithms on 63 data sets. The algorithms chosen were: NaiveBayes, BayesNet, NearestNeighbours, AdaBoost, OneR, Jrip, Ridor, NNge, J48, RandomForest, OneR and SimpleCart. We selected as features the number of attributes and instances in the data set, the number of categorical and numerical attributes, the type of data in the data set (numeric, nominal or mixed) and the number of classes. Regarding quality, we chosen completeness (percentage of null values) and finally, we used class entropy in order to establish if the class was balanced or not.

Next, we generated three data sets with the mentioned meta-features as attributes and the algorithm which achieved better performance as class value. The "md1" meta-data set contains an instance per data set, we only considered the best algorithm. If two or more algorithms achieved the same accuracy, all of them were included (84 instances). The "md2" meta-data set follows the same criteria as "md1" but in this case, we only used the models obtained by J48, JRIP, NaïveBayes and BayesNet. Finally, "md3" meta-data set contains as many instances as models achieved an accuracy whose statistical difference assessed by t-test was lower than 5%, with OneR as base algorithm.

In order to evaluate what meta-features are more useful to build the recommender, we applied a filtering algorithm offered by Weka, ClassifierSubSetEval. This algorithm returns how important the features are to perform a prediction task. It requires a base classifier as parameter, so it focuses on what attributes are more useful for a single classifier. Since we are focused on offering a recommender to non-experts in data mining, the base classifier used by ClassifierSubSetEval should be a prediction model easy to understand. We chosen two algorithms, J48 and NaïveBayes with the aim of testing two different approaches. The results when ClassifierSubSetEval was run are shown in Table 1, using as search algorithm LinearForwardSelection . The importance of the features is measured by means of an scale from 0 (useless) to 10 (very useful).

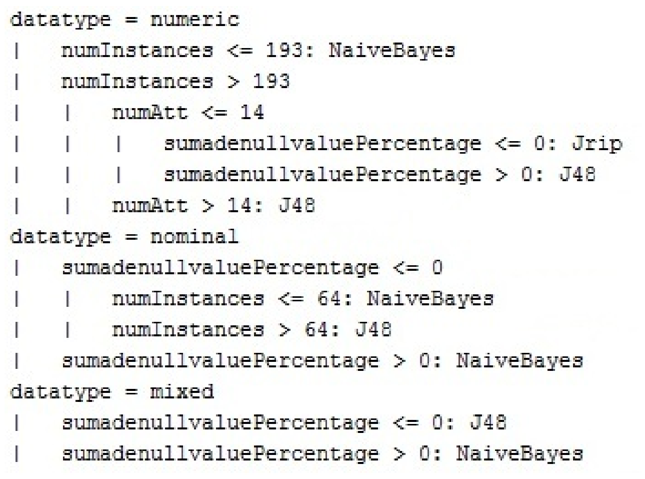
Analysing the results we can say that the degree of class imbalance, the number of instances and the completeness have a high significance. The other feature that seems to be important is the number of attributes. The rest of features are significant depending on the classification algorithm. For instance, the type of data is quite important for J48, but meaningless for NaïveBayes.

Table 1: Recommended features by ClassifierSubSetEval

	md1		md2		md3	
	NB	J48	NB	J48	NB	J48
#N Instances	9	5	8	10	8	8
#N Attributes	7	9	9	2	6	8
#N Numeric att.	1	3	0	4	5	5
#N Nominal att.	9	3	0	0	1	6
Completeness	10	10	8	10	7	9
#Type att.	7	3	6	4	5	4
#N Classes.	6	2	1	9	3	5
Is_balanced?	9	8	9	7	3	3

Next, we built models using the three meta-data sets generated for this experimentation. The more accurate model was achieved with "md2". This is due to the class attribute has 4 possible values in a data set with 80 instances, whereas, in "md1" and "md3", it has 12 different values with a slightly higher number of instances. Moreover, most models built from "md1" and "md3" were over-fitted.

Figure 1 depicts a model built with "md2" using J48. As expected, according to previous features analysis, it uses the type of data, the number of instances, the number of attributes and the completeness to build the model. From 81 data sets, 63 were used for building our recommender and the rest for testing. It achieved an accuracy of 68.75% which is a little lower than those obtained by classifiers built



```
datatype = numeric
|   numInstances <= 193: NaiveBayes
|   numInstances > 193
|   |   numAtt <= 14
|   |   |   sumadenullvaluePercentage <= 0: Jrip
|   |   |   sumadenullvaluePercentage > 0: J48
|   |   numAtt > 14: J48
datatype = nominal
|   sumadenullvaluePercentage <= 0
|   |   numInstances <= 64: NaiveBayes
|   |   numInstances > 64: J48
|   sumadenullvaluePercentage > 0: NaiveBayes
datatype = mixed
|   sumadenullvaluePercentage <= 0: J48
|   sumadenullvaluePercentage > 0: NaiveBayes
```

Figure 1: J48 Recommender

for this experimentation (range from 55% to 76%).

3. CONCLUSIONS

We have analysed which features are more suitable for describing educational data sets aimed at predicting student performance. We have also shown that construction of a recommender system following a meta-learning approach is feasible.

In a near future we will work with other kind of meta-characteristics such as the mentioned landmarkers and setting parameters of the algorithms. Of course, other quality measures of model, in addition to accuracy, will be considered.

4. REFERENCES

[1] H. Blockeel and J. Vanschoren. Experiment databases: Towards an improved experimental methodology in machine learning. In J. Kok, J. Koronacki, R. Lopez de Mantaras, S. Matwin, D. Mladenic, and A. Skowron, editors, *Knowledge Discovery in Databases: PKDD 2007*, volume 4702 of *Lecture Notes in Computer Science*, pages 6–17. Springer Berlin / Heidelberg, 2007.

[2] M. Hilario. e-lico annual report 2010. Technical report, Université de Geneve, 2010.

[3] D. Michie, D. J. Spiegelhalter, C. C. Taylor, and J. Campbell, editors. *Machine learning, neural and statistical classification.* Ellis Horwood, Upper Saddle River, NJ, USA, 1994.

[4] B. Pfahringer, H. Bensusan, and C. Giraud-carrier. Meta-learning by landmarking various learning algorithms. In *in Proceedings of the 17th International Conference on Machine Learning*, pages 743–750. Morgan Kaufmann, 2000.

Identifying and Visualizing the Similarities Between Course Content at a Learning Object, Module and Program Level

Kyle Goslin
Institute of Technology Blanchardstown
Blanchardstown Road North
Dublin 15, Ireland
kylegoslin@gmail.com

Markus Hofmann
Institute of Technology Blanchardstown
Blanchardstown Road North
Dublin 15, Ireland
markus.hofmann@itb.ie

ABSTRACT

As an educational institute grows an increase in the number of programs each with individual modules and learning objects can be seen. Learning environments provide a structured environment that can provide an additional level of insight into the relationship between content.

This paper outlines the identification of similarities at a Learning Object, Module and Program Level utilizing these inherent structures. Once generated, these results are then visualized in graph form providing an insight into the overlap between course material.

Keywords

Similarity Detection, Visualization, Data Structures, Moodle

1. INTRODUCTION

As institutions grow the replication of course material across departments also grows. This can be seen on a module level where subjects taught are quite similar but also on a cross department level whereby courses may not be directly linked but can have some unseen commonalities.

As a solution to this a tool was developed to extract the hierarchy and structures of the learning environment created by educators during their daily use. Similarity measures between documents are then calculated and can be used along with the gathered structural information to aid the process of narrowing and selecting applicable learning objects with similar content. These results are then visualized in graph from to aid the process of similarity detection.

2. BACKGROUND

Over the last number of years various different search tools [1] have been created that utilize the tagging of learning objects [2] through the use of metadata [3] and simple string matching. These approaches, however useful, do not take into consideration any of the prior knowledge that can be extracted from the environment to aid this search process. Each of these search queries is also limited to the relevancy and accuracy of the search terms entered by the user which often may not be as specific and relevant as required [4].

3. SYSTEM OVERVIEW

The *Tree Generator* creates a tree based structure of Moodle including meta data. A second tool titled the *Moodle Crawler* downloads each file from the Moodle instance locally and associates the Tree record ID to each file. Each file then converted into their HTML counter part and added to the local tree. Similarities between the data are then generated using the free and open source data mining tool RapidMiner [5]. A graph is then generated using a custom operator and viewed using Gephi [6].

Figure 1 below provides an overview of the generation process from start to finish.

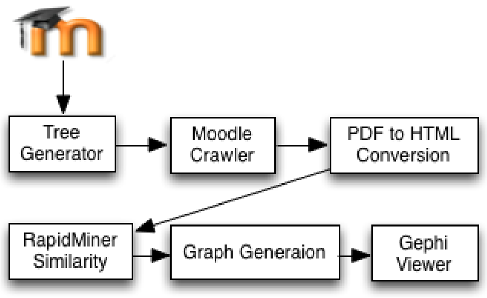


Figure 1: System Process Overview

3.1 Dataset

A live Moodle installation was used consisting of 30 modules with over 300 individual learning objects in 2 departments. These modules were in the fields of Information Technology and Business Administration. A number learning objects contained in these modules contain similar themes and could provide a strong baseline for similarity assessment.

4. VISUALIZATION

During the visualization process a number of different relationships between nodes were created by the *Tree Builder*.

Although used, these relationships were filtered out from the generated graphs to create a clearer visualization.

Graph 1: Single Program Similarity
Figure 2 shows the result of the graph generation process after a single program was selected. This single program contained three different modules, each with N number of different learning objects. In this graph, three different modules can be seen. Each modules learning objects were grouped close together to aid readability. Each different edge thickness outlines the higher the similarity to each different learning object.

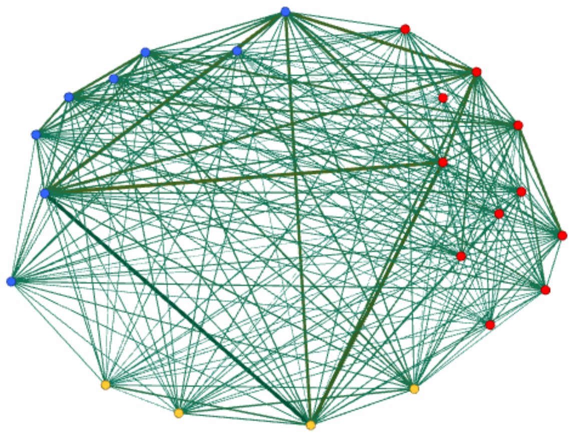


Figure 2: All learning objects inside each module (outlined by different colors)

Graph 2: Single Program Similarity - Filtered
Figure 3 identifies the similarity between programs. A filter was used to remove a number of different connections from the graphic. Each node represents individual learning objects in a program (identified by colour). Clear connections between content can be identified.

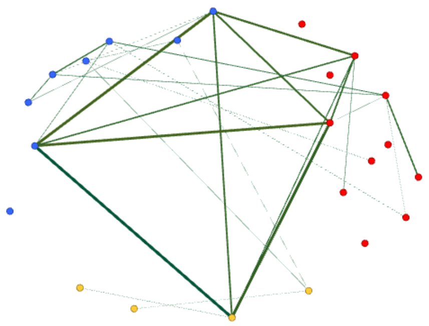


Figure 3: Learning objects in modules with emphasis on interesting edges

Graph 3: Full Moodle
To provide an overall view of the system a graph was created showing all of the modules inside the moodle instance. Each module is identified by different colors. Each of these modules consist of N number of different learning objects. Figure 4 shows a detailed graph was produced showing a number of different connections between modules.

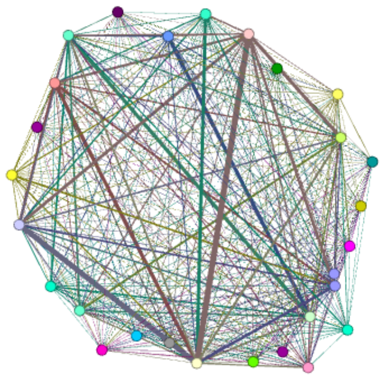


Figure 4: Similarities between all modules in a Moodle instance

5. CONCLUSIONS
This paper outlined the process of visualizing the similarities between content in a Moodle installation on a learning object, module and program level using custom tools to utilize the hierarchal structures of Moodle.

Once visualized, clear connections can be identified between learning objects and modules. The inherent tree based structures behind each node proved to help provide an additional level of context during the similarity generation process, allowing for a natural narrowing of the data set.

6. REFERENCES

[1] C. Curlango-Rosas, G.A. Ponce, G. Lopez-Morteo, and M. Mendiola. Leveraging google web search technology to find web-based learning objects. In *Web Congress, 2009. LA-WEB '09. Latin American*, pages 169 –176, nov. 2009.

[2] José-Luis Sierra and Alfredo Fernández-Valmayor. Tagging learning objects with evolving metadata schemas. In *Proceedings of the 2008 Eighth IEEE International Conference on Advanced Learning Technologies*, ICALT '08, pages 829–833, Washington, DC, USA, 2008. IEEE Computer Society.

[3] Ernesto Diaz-Aviles, Marco Fisichella, Ricardo Kawase, Wolfgang Nejdl, and Stewart. Unsupervised auto-tagging for learning object enrichment. In *Proceedings of the 6th European conference on Technology enhanced learning: towards ubiquitous learning*, EC-TEL'11, pages 83–96, Berlin, Heidelberg, 2011. Springer-Verlag.

[4] N.Y. Yen, T.K. Shih, L.R. Chao, and Qun Jin. Ranking metrics and search guidance for learning object repository. *Learning Technologies, IEEE Transactions on*, 3(3):250 –264, july-sept. 2010.

[5] Ingo Mierswa, Michael Wurst, Ralf Klinkenberg, Martin Scholz, and Timm Euler. Yale: Rapid prototyping for complex data mining tasks. In Lyle Ungar, Mark Craven, Dimitrios Gunopulos, and Tina Eliassi-Rad, editors, *KDD '06*, pages 935–940, New York, NY, USA, August 2006. ACM.

[6] Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. Gephi: An open source software for exploring and manipulating networks, 2009.

Using ITS Generated Data to Predict Standardized Test Scores

Kim Kelly
Worcester Polytechnic Institute
kkelly@wpi.edu

Ivon Arroyo
Worcester Polytechnic Institute
iarroyo@wpi.edu

Neil Heffernan
Worcester Polytechnic Institute
nth@wpi.edu

ABSTRACT

This study suggests that the data generated by intelligent tutoring systems can be used to accurately predict end-of-year standardized state test scores. A traditional model including only past performance on the test yielded an R^2 of 0.38 and an enhanced traditional model that added current class average improved predictions (R^2=0.50). These models served as baseline measures for comparing an ITS model. Logistic regression models that include features such as hint percentage, average number of attempts and percent correct overall improved the R^2 to 0.57. The predictive power of the data is as effective with only a few months of use. This lends support for the increased use of the systems in the classroom and for nightly homework.

Keywords

Intelligent tutoring system, homework, standardized test, prediction, regression, classification, decision tree

1. INTRODUCTION

With the introduction of the No Child Left Behind Act in 2001, assessing student performance became a significant focus of schools. With the high-stake nature of these tests, it is imperative to identify at-risk students accurately and as early in the year as possible to provide time for interventions. Intelligent tutoring systems (ITS) allow teachers to evaluate student performance while students are learning. Furthermore, the ITS provides data to teachers which can be used to predict standardized-state-test scores (Feng, et al. 2006, Feng, et al. 2008). Specifically, help request behavior is effective at predicting student proficiency (Beck et al. 2003).

While the above studies are promising, the content used to generate the data was very narrow, consisting of previously released state test questions. Therefore the material mapped directly to the test that was being predicted. The present study uses ASSISTments (www.assistments.org), a web-based intelligent tutoring system, which allows teachers to enter their own content in addition to using certified problem sets. This content can include in-class warm ups, challenge problems, and questions from the textbook. Some of the problem sets may include tutoring in the form of hints or scaffolding while others include correctness only feedback with varying numbers of attempts allowed. What impact does this diverse data have on the previously established usefulness of ITS in predicting end-of-year test scores? The present research attempts to determine if the data collected from student use of an ITS over an entire school year accurately predicts student performance on a standardized-state-test.

2. APPROACH

For the 2010-1011 school year, 129 students in a suburban middle school used ASSISTments as part of their 7th grade math class. The different types of assignments completed during the course of the year include classwork, homework and assessments. Student data from August through May was used to predict MCAS (Massachusetts Comprehensive Assessment System) scaled score and to classify performance. A smaller date range (August through October) was also considered to determine if the model is equally effective with less data, earlier in the school year.

2.1 Modeling

The **traditional model** most schools use to predict 7th grade MCAS scaled scores is a student's 6th grade MCAS score. An **enhanced traditional model** added student's average. These models serve as a baseline to compare the **ITS models** for the different date ranges. Based on the previous literature that successfully predicted state test scores from an ITS, many variables were constructed to be included in the model (number of questions answered, percent correct on first attempt, percentage of hints used, and average number of attempts per question).

For predicting a student's 7th grade MCAS scaled score, linear regression was used for both the traditional and enhanced traditional models. Whereas step-wise linear regression models were generated for both date ranges. The models were compared using R^2 and accuracy. To measure the accuracy of each model, the predicted score was used to classify each student (advanced, proficient, needs improvement, warning) and this classification was compared to the actual classification on the 7th grade MCAS.

For classifying purposes, decision trees were generated to predict specific performance level for each time frame as well. Cross validation was used to assess the accuracy of the models.

3. RESULTS

Students who were not enrolled in the course for the entire time period considered in this study were not included in the analysis (n=8). Finally, students whose 6th or 7th grade MCAS scores were not available were not included (n=4).

3.1 Prediction

Traditionally, prior performance on a standardized test is used to predict future performance on the same test. A linear regression using only 6th grade MCAS score to predict 7th grade MCAS scaled score serves as a comparison model for the more complex models. These scores are highly correlated ($r(115)=0.617$, $p<0.001$) and was 75% accurate in categorizing students.

The **enhanced traditional model** included 6th grade MCAS score ($\beta= 0.341$, $t(115) = 4.03$, $p < .001$) as well as Percent_Correct_First_Attempt ($\beta= 0.448$, $t(115) = 5.30$, $p <$

.001). This model was more successful (F(2,114)=57.10, p<0.001) and accounted for 51% of the variance. This model also yielded 75% accuracy in categorizing students.

A step-wise linear regression was used to generate a model that incorporates data collected from student use of ASSISTments. The regression model using the data from August through May yielded a model that included percent correct overall ($\beta = 0.479$, $t(115) = 6.80$, $p < .001$) and class ($\beta = 0.399$, $t(115) = 5.67$, $p < .001$). This model was 82% accurate. A step-wise regression generated an identical model using only the data from August through October and resulted in 81% accuracy. This confirms that the data collected during the first quarter is equally sufficient when predicting end-of-year test results. Furthermore, the inclusion of a measure only available through the use of the intelligent tutoring system supports its use in the classroom. The results of these models can be found in Table 1.

3.2 Classification

A J48 Decision tree with cross validation predicted MCAS classification with 68.4% accuracy for the full year. The attributes included in the tree were prior MCAS performance, total number of questions answered, and percentage of hints used. While the tree does well with predicting the classification of Advanced and Proficient, it was unable to identify the students who fell in the Needs Improvement category. This is a significant limitation of this model. However, it is important to note that with only 2 students falling in the Needs Improvement category, it will be very challenging to identify them.

A separate decision tree was constructed based on the data from August through October. This model predicted MCAS classification better with 76% accuracy. See Figure 1 for the tree. The attributes that were included were average number of attempts and percentage of hints used.

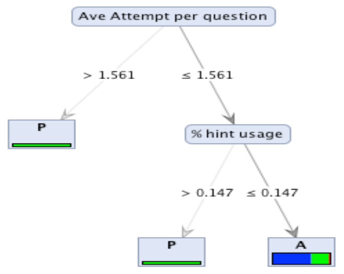


Figure 1. J48 Decision Tree for predicting MCAS classification using data from August through October.

4. Contribution and Discussion

Being able to predict how students will perform on end-of-year standardized tests allows teachers to identify at risk students and offer interventions. Traditionally, teachers had only the previous year's performance. While this is highly correlated with current performance, the current study shows that ITS provide additional data that allow teachers to better predict performance, and earlier in the year. Using this data to offer remediation and interventions should be considered by educators who use ITS regularly.

This study is unique in that the ITS was used throughout the year for nightly homework, often with correctness only feedback and diverse content that is not closely mapped to the final measurement of performance. The ability to still accurately predict performance provides evidence of yet another use of ITS within schools.

Table 1. Predictive and classification model performance.

Model	R^2	Accuracy	Kappa
Traditional	0.381	75%	0.44
Enhanced Traditional	0.505	75%	0.46
ITS (First Quarter)	0.566	81%	0.61
ITS (Full Year)	0.566	82%	0.63
Classification (First Quarter)	N/A	76%	0.39
Classification (Full Year)	N/A	68%	0.36

Both prediction and classification required data that could only be generated by the use of ITS, and not through traditional classroom measures. Specifically, percent correct overall was a useful predictor. This can only be generated by allowing students to learn while doing their homework and measuring their success beyond just their first response. Similarly, average number of attempts on each problem and hint percentage add to the model when trying to classify student performance. This lends more support for the use of ITS for nightly homework.

5. ACKNOWLEDGMENTS

Bill and Melinda Foundation via EDUCAUSE, IES grants R305C100024 and R305A120125, and NSF grants ITR 0325428, HCC 0834847, DRL 1235958.

6. REFERENCES

[1] Beck, J., Jia, P., Sison, J., Mostow, J. (2003). Predicting student help-request behavior in an intelligent tutor for reading. Proc. of the 9 th Int. Conf. on User Modeling, LNAI 2702

[2] Feng, M., Heffernan, N., Koedinger, K., (2006). Predicting state test scores better with intelligent tutoring systems: developing metrics to measure assistance required. Proceedings of the 8th International Conference on Intelligent Tutoring Systems. Springer-Verlag: Berlin.

[3] Feng, M., Beck, J., Heffernan, N., Koedinger, K. (2008). Can an intelligent tutoring system predict math proficiency as well as a standardized test. In Baker & Beck (Eds.). Proceedings of the First International Conference on Educational Data Mining.

Joint Topic Modeling and Factor Analysis of Textual Information and Graded Response Data

Andrew S. Lan, Christoph Studer, Andrew E. Waters, Richard G. Baraniuk
Rice University, USA
{mr.lan, studer, waters, richb}@sparfa.com

ABSTRACT
Modern machine learning methods are critical to the development of large-scale personalized learning systems that cater directly to the needs of individual learners. The recently developed SPARse Factor Analysis (SPARFA) framework jointly estimates learner's knowledge of the latent concepts underlying a domain and the relationships among a collection of questions and the latent concepts, solely from the graded responses to a collection of questions. To better interpret the estimated latent concepts, SPARFA relies on a post-processing step that utilizes user-defined tags (e.g., topics or keywords) available for each question. In this paper, we relax the need for user-defined tags by extending SPARFA to jointly process both graded learner responses and the text of each question and its associated answer(s) or other feedback. Our purely data-driven approach (i) enhances the interpretability of the estimated latent concepts without the need of explicitly generating a set of tags or performing a post-processing step, (ii) improves the prediction performance of SPARFA, and (iii) scales to large test/assessments where human annotation would prove burdensome. We demonstrate the efficacy of the proposed approach on two real educational datasets.

1. INTRODUCTION
Traditional education typically provides a "one-size-fits-all" learning experience, regardless of the different backgrounds, abilities, and interests of individual learners. Recent advances in machine learning enable the design of computer-based systems that analyze learning data and provide feedback to the individual learner. Such an approach has the potential to revolutionize today's education by offering a high-quality, personalized learning experience to learners.

Several efforts have been devoted into building statistical models and algorithms for learner data analysis. In [4], we proposed a personalized learning system (PLS) architecture based on the SPARse Factor Analysis (SPARFA) framework for learning and content analytics, which decomposes assessments into different knowledge components that we call *concepts*. SPARFA automatically extracts (i) a question–concept association graph, (ii) learner concept knowledge profiles, and (iii) the intrinsic difficulty of each question, solely from graded binary learner responses to a set of questions. This framework enables a PLS to provide personalized feedback to learners on their concept knowledge, while also estimating the question–concept relationships that reveal the structure of a course.

The original SPARFA framework [4] relies on a post-processing step to associate instructor-provided question tags to each estimated concept. Inspired by the recent success of modern text processing algorithms, such as latent Dirichlet allocation (LDA) [2], we posit that the text associated with each question can potentially reveal the meaning of the estimated latent concepts without the need of instructor-provided question tags. Such an data-driven approach is advantageous as it easily scales to domains with thousands of questions.

In this paper, we propose *SPARFA-Top*, which extends the original SPARFA framework [4] to jointly analyze graded learner responses to questions and the text of the questions, responses, or feedback. To this end, we augment SPARFA by a Poisson model for the word occurrences associated with each question. We develop a computationally efficient block-coordinate descent algorithm that, given only binary-valued graded response data and associated text, estimates (i) the question–concept associations, (ii) learner concept knowledge profiles, (iii) the intrinsic difficulty of each question, and (iv) a list of most important keywords associated with each estimated concept. We show that SPARFA-Top is able to automatically generate a human readable interpretation for each estimated concept in a purely data-driven fashion. This capability enables a PLS to automatically recommend remedial or enrichment material to learners that have low/high knowledge level on a given concept.

2. THE SPARFA-TOP MODEL
SPARFA [4] assumes that graded learner response data consist of N learners answering a subset of Q questions that involve $K \ll Q, N$ underlying (latent) concepts. Let the column vector $\mathbf{c}_j \in \mathbb{R}^K$, $j \in \{1, \ldots, N\}$, represent the latent *concept knowledge* of the j^{th} learner, let $\mathbf{w}_i \in \mathbb{R}^K$, $i \in \{1, \ldots, Q\}$, represent the *associations* of question i to each concept, and let the scalar $\mu_i \in \mathbb{R}$ represent the *intrinsic difficulty* of question i. The student–response relationship is modeled as

$$\begin{aligned} Z_{i,j} &= \mathbf{w}_i^T \mathbf{c}_j + \mu_i, \quad \forall i, j, \quad \text{and} \\ Y_{i,j} &\sim Ber(\Phi(\tau Z_{i,j})), \quad (i,j) \in \Omega_{\text{obs}}, \end{aligned} \tag{1}$$

where $Y_{i,j} \in \{0,1\}$ corresponds to the observed binary-valued graded response variable of the j^{th} learner to the i^{th} question, where 1 and 0 indicate correct and incorrect responses, respectively. $Ber(z)$ designates a Bernoulli distribution with success probability z, and $\Phi(x) = \frac{1}{1+e^{-x}}$ denotes the inverse logit link function. The set Ω_{obs} contains the indices of the observed entries (i.e., the observed data

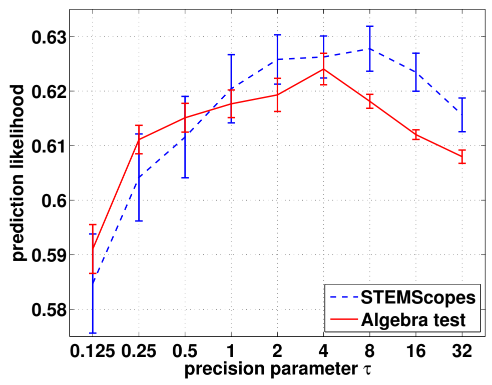


Figure 1: Average predicted likelihood using SPARFA-Top with different precision parameters τ.

may be incomplete). The precision parameter τ models the *reliability* of the observed binary graded response $Y_{i,j}$. In order to account for real-world educational scenarios, $\mathbf{w}_i$ is assumed to be sparse and non-negative [4].

We now introduce a novel approach to *jointly* consider graded learner response and associated textual information, to directly associate keywords with the estimated concepts. Assume that we observe the word–question occurrence matrix $\mathbf{B} \in \mathbb{N}^{Q\times V}$, where V corresponds to the size of the vocabulary, i.e., the number of *unique* words that have occurred among the Q questions. Each entry $B_{i,v}$ represents how many times the v^{th} word occurs in the associated text of the i^{th} question. Inspired by the topic model proposed in [6], the entries of the word-occurrence matrix $B_{i,v}$ in (2) are assumed to be *Poisson* distributed as follows:

$$A_{i,v} = \mathbf{w}_i^T \mathbf{t}_v \quad \text{and} \quad B_{i,v} \sim Pois(A_{i,v}), \quad \forall i, v, \qquad (2)$$

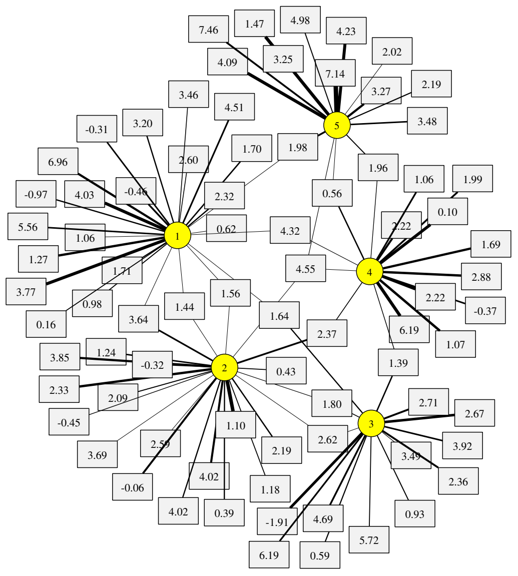
where $\mathbf{t}_v \in \mathbb{R}_+^K$ is a non-negative column vector that characterizes the expression of the v^{th} word in every concept. The latent factors $\mathbf{w}_i$, $\mathbf{c}_j$, $\mathbf{t}_v$ and μ_i are estimated through a block coordinate descent algorithm, which is detailed in [3].

3. EXPERIMENTS

We now demonstrate the efficacy of SPARFA-Top on two real-world educational datasets: an 8th grade Earth science course dataset provided by STEMscopes [5] and a high-school algebra test dataset administered on Amazon's Mechanical Turk [1], a crowdsourcing marketplace.

In Figure 1, we show the prediction likelihood defined by $p(Y_{i,j}|\mathbf{w}_i^T\mathbf{c}_j + \mu_i, \tau), (i,j) \in \bar{\Omega}_{\text{obs}}$ for SPARFA-Top on 20% holdout entries in $\mathbf{Y}$ and for varying precision values τ. We see that textual information can slightly improve the prediction performance of SPARFA-Top over SPARFA (which corresponds to $\tau \to \infty$), for both datasets. The reason for (albeit slightly) improving the prediction performance is the fact that textual information actually reveals additional structure underlying a given test/assessment.

Figure 2 shows the question–concept association graph along with the recovered intrinsic difficulties, as well as the top three words characterizing each concept, for the STEMscopes dataset. Compared to SPARFA (cf. [4, Fig. 2]), we observe that SPARFA-Top is able to relate all questions to concepts, including those questions that were found to be



Concept 1	Concept 2	Concept 3	Concept 4	Concept 5
Energy	Water	Plants	Water	Water
Water	Percentage	Buffalo	Soil	Heat
Earth	Sand	Eat	Sample	Objects

Figure 2: Question–concept association graph and most important keywords recovered by SPARFA-Top for the STEMscopes dataset; boxes represent questions, circles represent concepts, and thick lines represent strong question–concept associations.

unrelated to any concept. Furthermore, the table in Figure 2 demonstrates that SPARFA-Top is capable of automatically generating an interpretable meaning of each concept.

4. CONCLUSIONS

We have introduced the SPARFA-Top framework, which extends SPARFA by jointly analyzing both the binary-valued graded learner responses to a set of questions and the text associated with each question via a Poisson topic model. Our purely data-driven approach avoids the manual assignment of tags to each question and significantly improves the interpretability of the estimated concepts by automatically associating keywords extracted from question text to each estimated concept. For additional details, please refer to the full version of this paper on arXiv [3].

5. REFERENCES

[1] Amazon Mechanical Turk. http://www.mturk.com/mturk/welcome, Sep. 2012.

[2] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *JMLR*, 3:993–1022, Jan. 2003.

[3] A. S. Lan, C. Studer, A. E. Waters, and R. G. Baraniuk. Joint topic mmodeling and factor analysis of textual information and graded response data. *arXiv preprint: arxiv.org/abs/1305.1956*.

[4] A. S. Lan, A. E. Waters, C. Studer, and R. G. Baraniuk. Sparse factor analysis for learning and content analytics. Oct. 2012, submitted.

[5] STEMscopes Science Education. http://stemscopes.com, Sep. 2012.

[6] J. Zhu and E. P. Xing. Sparse topical coding. In *Proc. 27th Conf. on Uncertainty in Artificial Intelligence*, Mar. 2011.

Component Model in Discourse Analysis

Haiying Li
University of Memphis
202 Psychology Building
University of Memphis
Memphis, TN, 38152
Tel. 1-901-678-2364
hli5@memphis.edu

Arthur C. Graesser
University of Memphis
202 Psychology Building
University of Memphis
Memphis, TN, 38152
Tel. 1-901-678-4857
graesser@memphis.edu

Zhiqiang Cai
University of Memphis
202 Psychology Building
University of Memphis
Memphis, TN, 38152
Tel. 1-901-678-2364
zcai@memphis.edu

ABSTRACT

Automated text analysis tools such as Coh-Metrix and Linguistic Inquiry and Word Count (LIWC) provides overwhelming indices for text analysis, so fewer underlying dimensions are required. This paper developed an underlying component model for text analysis. The component model was developed from large English and Chinese corpora in terms of results from Coh-Metrix, and English and Chinese Linguistic Inquiry and Word Count (LIWC).

Keywords

Component model, Coh-Metrix, LIWC, principal component analysis

1. INTRODUCTION

With the development of the computational linguistics, automated text analysis tools like Coh-Metrix and Linguistic Inquiry and Word Count (LIWC) have been developed to analyze enormous amounts of data efficiently.

Coh-Metrix provides 53 language and discourse measures at multilevels related to conceptual knowledge, cohesion, lexical difficulty, syntactic complexity, and simple incidence scores (http://cohmetrix.memphis.edu) [1]. Meanwhile, a principle components analysis performed on 37,520 texts of TASA corpus extracts five factors (Coh-Metrix-Text Easability Assessor, TEA, http://tea.cohmetrix.com), including Narrativity (word familiarity and oral language), Referential cohesion (content word overlap), Deep cohesion (causal, intentional, and temporal connectives), Syntactic simplicity (familiar syntactic structures), and Word concreteness (concrete words) [1].

Even though the Coh-Metrix provides the normed five dimensions, no articles describe the details of this model. This paper not only gives a thorough description of this model, but also uses this method to build up the normed dimensions with the text analysis tools of English and Chinese LIWC.

LIWC is a text analysis software program with a text processing module and an internal default dictionary [2]. LIWC classifies words into 64 linguistic and psychological categories. The 2007 English LIWC dictionary contains 4,500 words and word stems.

The Chinese LIWC dictionary was developed by National Taiwan University of Science and Technology based on the LIWC 2007 English dictionary, but some word categories unique to the Chinese language were added to the Chinese LIWC dictionary [3]. The Chinese LIWC dictionary included 6,800 words across 71 categories. The Memphis group converted the traditional Chinese characters in LIWC dictionary to the simplified Chinese characters, which was used in our study.

With the overwhelming features for text analysis, researchers prefer fewer underlying dimensions. The most prevalent method to reduce the dimensionality is the principal component analysis (PCA) in text analysis [4, 5]. However, PCA assumes the ratio of cases to variables, so the corpus with smaller amount of cases is inappropriate to perform PCA [6]. Therefore, the standardized and normed component scores from the large reference corpus are needed.

This paper aims to develop a component model of text analysis with the automated tools of Coh-Metrix and LIWC; thus, the component scores of any coming data set computed with this model will be standardized and comparable.

2. METHOD

Two reference corpora were used in this study. The English corpus used TASA (Touchstone Applied Science Associates, Inc.), randomly-collected excerpts of 37,520 samples, 10,829,757 words with nine genres, including language arts, science, and social studies/history, business, health, home economics and industrial arts.

The Chinese reference corpus was collected according to similar genres in TASA such as classic fiction, modern fiction, history, science. Texts in the Chinese corpus included complete 4,679 documents with 25,184,754 words rather than segmented.

Six factors extracted from LIWC in these two independent corpora showed significantly high correlation on dimensions of cognitive complexity, narrativity, emotions and embodiment [7]. Therefore, these two corpora are able to reflect some common linguistic and psychological features.

The procedure of the component model is described below. First, TASA was analyzed by Coh-Metrix, English LIWC; Chinese corpus was analyzed by Chinese LIWC. Thus, three data sets were generated. Second, PCA was performed to reduce a range of indices from Coh-Metrix (53) and LIWC (English 64; Chinese 71) to fewer potential constructs. The fixed number of dimensions was decided by the eigenvalue greater than 2. Finally, the mean, standard deviation and coefficient for each category in each dimension were extracted to develop component model.

3. RESULTS AND DISCUSSION

The factorability of the items for the appropriateness of the performance of PCA used such criteria as the ratio of cases to variables, correlations, Kaiser-Meyer-Olkin measure, and Bartlett's sphericity.

First, the ratio of cases to variables at least 521:1 was satisfied. Then the majority of correlations among indices were above .50. Secondly, the overall Kaiser-Meyer-Olkin (KMO) measure of sampling adequacy of Coh-Metrix sets was higher than .50 and the Bartlett test of sphericity was statistically significant across three data sets. All indices were included in the analyses. The varimx rotation was used in the analysis.

The initial eigen values greater than 2 indicated the appropriate fixed number of components. One reason why we used eigen values greater than 2 rather than 1 was that too many components were extracted with eigen values greater than 1. In TASA Coh-Metrix, 6 components were extracted explaining 58% of the total variance. In TASA LIWC, 6 components were extracted and explained 40% of the total variance. In Chinese LIWC, 7 components were extracted and explained 53% of the total variance.

The components were labeled based on the linguistic or psychological features of the highly loaded categories in the component. For the English Coh-Metrix data set, the components were labeled from the first to the fifth in order as Narrativity; Referential Cohesion; Syntactic Simplicity, Word Concreteness, and Deep Cohesion. The last component only had 3 variables, so we removed that component from the model. For the English LIWC data set, the components were labeled from the first to the sixth in order as Narrativity; Processes, Procedures, Planning; Social Relations; Negative Emotion; Embodiment; Collection. For the Chinese LIWC data set, the components were labeled in the order from the first to the seventh as Processes, Procedure, Planning; Narrativity; Space and Time; Embodiment; Positive Emotion, Negative Emotion; and Personal Concerns.

The component composite score for the coming data set will be computed through an automated tool developed according to the formula of Component Model. Component Model will be obtained by the following formula,

$$y = \sum_{1}^{n}\left(\frac{x-\mu}{s}\gamma\right)$$

among which *y* is a component score for a coming corpus (CC); *x* is the value of each category on a document of CC; *μ* is the mean of each category from reference corpus (RC) which includes TASA Coh-Metrix, TASA LIWC or Chinese LIWC; *s* is the standard deviation of each category from RC; γ is coefficient of each category from RC. *1* to *n* means the number of categories in each component. ∑ means the sum of all the scores of the categories on each component.

For example, a teacher would like to look at the composite component score of Negative Emotion from the students' writings in English with LIWC. The teacher only has 15 subjects, so this data set is inappropriate to perform PCA. Therefore, the English LIWC Component Model should be used. First, the teacher should analyze the writing with English LIWC to obtain the score of all the indices (64). Then the mean and standard deviation of indices in all the categories, the corresponding coefficients of Negative Emotion component in the Component Model should be obtained from the reference corpus.

For instance, the "verb" score for one subject is 1.5. According to the model, the mean of the "verb" is 1.37, standard deviation 1.29, and the coefficient -0.06. Thus, the value of the "verb" in the component score is [(1.5-1.37)/1.29](-0.06) = 0.01 for this subject. We need compute the value of all the other categories in this way, then sum them, and finally obtain the value of the Negative Emotion composite score for this subject.

Thus, each component composite score from any coming corpus will be computed and standardized based on this component model from these three component models.

4. CONCLUSION

This study developed three component models for text analysis with Coh-Metrix component model, English LIWC component model and the Chinese LIWC component model. The component model can be used to generate the composite component scores when the data set has a small sample size and PCA is inappropriately performed. The results are comparable across different data sets.

The limitation of this study is that we didn't evaluate the model with human judgment. In the future, the evaluation of the model will be carried out.

5. ACKNOWLEDGMENTS

This work was supported by the National Science Foundation (BCS 0904909) for the Minerva project: Languages across Culture.

6. REFERENCES

[1] McNamara, D.S., Graesser, A.C., McCarthy, P., and Cai, Z. in press. *Automated evaluation of text and discourse with Coh-Metrix*. Cambridge University Press, Cambridge.

[2] Pennebaker, J. W., Booth, R. J., and Francis, M. E. 2007. *LIWC2007: Linguistic Inquiry and Word Count.* Austin, Texas: LIWC.net

[3] Huang, J., Chung, C. K., Hui, N., Lin, Y., Xie, Y., Lam, Q., Cheng, W., Bond, M., and Pennebaker, J. W. 2012. 中文版語文探索與字詞計算字典之建立 [The development of the Chinese Linguistic Inquiry and Word Count dictionary]. *Chinese Journal of Psychology, 54(2)*, 185-201.

[4] Biber, D. 1988. *Variation across speech and writing*. Cambridge, England: Cambridge University Press.

[5] Lee, D. Y. W. 2004. *Modeling variation in spoken and written English*. Routledge, London/New York.

[6] Hair, J. F., Black, W. C., Babin, B. J., and Anderson, R. E. 2009. *Multivariate data analysis*. Prentice Hall, New Jersey.

[7] Li, H., Cai, Z., Graesser, A.C., and Duan, Y. 2012. A comparative study on English and Chinese word uses with LIWC. In *Proceedings of the Twenty-Fifth International Florida Artificial Intelligence Research Society Conference*. (California, US, May 23 – 25, 2012), 238-243.

Modeling Student Retention in an Environment with Delayed Testing

Shoujing Li
Worcester Polytechnic Institute
100 Institute RD; Worcester MA
sli@wpi.edu

Xiaolu Xiong
Worcester Polytechnic Institute
100 Institute RD; Worcester MA
xxiong@wpi.edu

Joseph E. Beck
Worcester Polytechnic Institute
100 Institute RD; Worcester MA
josephbeck@wpi.edu

ABSTRACT

Student modeling has been widely used in the prediction of student correctness behavior on the immediate next action. Some researchers have been working on student modeling to predict delayed performance, that is, *retention*. Prior work has found that the factors influencing retention differ from those that influence short-term performance. However, this prior research did not use data which were specially targeted to measure retention. In this study, we describe our experiments of using dedicated retention performance data to test the students' ability to retain, and experiment with a new feature called *mastery speed*, indicates how many problems the students need to attain initial mastery. We found that this new feature is the most useful of our features. It's not only a helpful predictor for 7-day retention tests, but also a long-term factor that influences students' later retention tests even after 105 days. We also found that, although statistically reliable, most features are not *useful predictors*, such as the number of students' previous correct and incorrect responses which are not as helpful in predicting students' retention performance as in PFA.

Keywords

Educational data mining, Knowledge retention, Robust learning, Feature selection, Intelligent tutoring system.

1. INTRODUCTION

Automatic Reassessment and Relearning System (ARRS) is an extension of the mastery learning problem sets in the ASSISTments system (www.assistments.org), a non-profit web-based tutoring system for 4th through 10th grade mathematics. Mastery Learning is a pedagogical strategy which, in most ITS, indicates that a student is presented with problems to solve until he masters the skill. The exact definition of "mastery" varies from tutor to tutor: some tutors consider a student to have mastered the skill if his estimated knowledge is very high, for example over 0.95 (e.g., [3]), while ASSISTments uses a heuristic of three correct responses in a row. The idea of ARRS is if a student masters a problem set, such mastery is not necessarily an indication of long-term retention. Therefore, ARRS will present the student with a reassessment test on the same skill at expanding intervals: first 7 days after the initial mastery is due, then 14 days after the prior test, than 28 days later, and finally 56 days later. Thus, the retention tests are spread over an interval of at least 105 (7+14+28+56) days. In this study, we defined retention performance as the reassessment test performance one week after a student was assigned a skill (i.e., the first reassessment test). Note, that if a student fails the reassessment test, ASSISTments will give him an opportunity to relearn the skill. Once a student relearns (demonstrates mastery) a skill, he will receive another reassessment test at the same delay at which he previously responded incorrectly. In other words, if the student failed the second reassessment test, he would have to relearn the skill and achieve 3 correct answers in a row, before receiving another reassessment test 14 days later.

In our previous study, we identified *mastery speed* as a useful construct in prediction of retention performance. Mastery speed refers to the number of attempted problems during the process of achieving mastery. Mastery speed represents a combination of how well the student knew this skill initially, and how quickly he can learn the skill.

2. MODELS AND RESULTS

2.1 Data set

For this study, we used data from the ARRS system, specifically students' 7-day test performance and other features about their previous learning on that particular skill. We had 48,873 questions answered by 4054 students, from 91 different skills. Then we calculated the following features which were used in our regression models:

- *mastery_speed*: the number of problems needed to master a certain skill. We binned this feature into 6 categories ('<3 attempts', '3-4 attempts', '5-8 attempts', '>8 attempts', 'not mastered', 'skipped initial mastery'). Students could master a skill in less than 3 attempts if their teachers overrode ASSISTments mastery criterion.
- *n_correct* (*n_incorrect*): the number of students' prior correct (incorrect) responses on that skill before the retention test.
- *n_day_seen*: the number of distinct days that the students have practiced this skill.
- *g_mean_performance*: the exponential moving average of students' performance before the reassessment test. We used the same formula as in Wang and Beck's previous work [2]: *g_mean_performance (opp) = g_mean_performance (opp-1) * 0.7 + correctness (opp) * 0.3* using opp to represent the opportunity count and a decay of 0.7.
- *g_mean_time*: the exponential moving average of students' response time on that skill before the reassessment test [2]. The formula is: *g_mean_time (opp) = g_mean_time (opp-1) * 0.7 + response_time (opp) * 0.3*.
- *problem_easiness*: percentage correct for this problem.

2.2 Separate Model with each Feature

In our binary logistic regression models, we used correctness as the dependent variable. We first tested a base model with just three features: *user_id*, *skill_id*, and *problem_easiness*, which showed as reliable predictors in a model we created with all features in our feature set. The base model provided an R^2 of 0.373. The next step we took was to test each feature one at a time added to the base model. Table 1 shows the Beta coefficient, p-values and R^2 gain for each regression model.

Each row in the table represents one regression model, with the feature listed and other three features in the basic model. The last column, R^2 gain, shows the increase in R^2 from adding that feature to the base model. Given even the modest (by EDM standards) data set we have for this study, circa 50,000 rows, even trivially small effects can show up as statistically "significant." Therefore, we compute how much improvement the feature actually provides us with. From the table, it's clear that *mastery_speed* is the most powerful predictor for students' retention performance. And also the students' previous performance on that skill (*g_mean_performance*) has a clear influence on prediction. The other variables have a trivial impact on performance. Note that even the best two features have a small impact on retention.

Compared with prior work [2], we found that *n_day_seen* did not replicate as being a useful feature. Strangely, a student's raw number of correct and incorrect response has little impact on retention. But *g_mean_performance* which measures students' previous performance on correctness has a clear influence on students' retention, which indicates that simply counting the raw number of correct or incorrect responses does not seem that helpful. Using exponential moving average which weights recent attempts more heavily as we did to compute *g_mean_performance*, is a helpful way to use students' previous correctness information.

Table 1. Parameters table for separate Models

Feature	R^2	B	p-value	R^2 gain
mastery_speed	0.379	---	0.000	0.006
n_correct	0.374	0.010	0.000	0.001
n_incorrect	0.373	-0.007	0.004	0.000
n_day_seen	0.373	0.026	0.002	0.000
g_mean_performance	0.378	1.130	0.000	0.005
g_mean_time	0.373	0.000	0.649	0.000

2.3 Impact of Mastery Speed

From the previous models we presented, we found that mastery speed has a clear influence on students' 7-day reassessment tests. However, what about the 14 day test, 28 day test, and even the 56 day tests? We collected all student performances on all four reassessment tests. As shown in Figure 1, we calculated the percentage of correct answers on each retention test, disaggregated by initial mastery speed.

Students get better as they move to the later retention tests. This is expected since they must get the previous tests correct in order to move on, and some weaker students are forced to repeat and so are systematically oversampled on the left side of the graph. On the 7-day retention test, students who mastered a skill quickly with 3 or 4 attempts (blue line) have a 24% higher chance of responding correctly than those students who required more than 8 attempts to master a skill (green line). Such a difference is perhaps not surprising. More interesting is the persistence of this differential performance: the 56 day level tests, the group who mastered quickly are still performing about 15% better than the students who mastered slowly. This difference persists in spite of weaker students being screened out on earlier retention tests. This result tells us that the initial mastery speed is of importance in terms of students' retention performance even after 105 days.

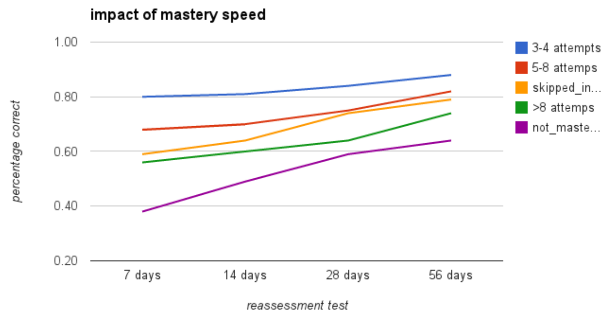


Figure 1. Impact of mastery speed on retention tests

3. CONCLUSIONS AND FUTURE WORK

This paper represents our attempt to model student retention performance in the context of a computer tutor. The two most interesting results were mastery speed being the best predictor, and the effects of performance on initial mastery persisting across such a lengthy interval. We did not anticipate this effect, and were therefore surprised by it.

There are several interesting open questions that might be further explored in the future. First, we have noticed anecdotally and through preliminary analysis that students sometimes get confused among similar skills during problem solving, an example of proactive interference [1]. Computer tutors would seem to be a strong research vehicle for better understanding of such effects in an authentic learning context, and over longer time than typical psychology lab studies. Another question is that we have found that slow mastery speed results in poor performance on delayed tests. An open question is whether a stronger mastery criterion, such as 4 or 5 correct in a row, would be helpful.

ACKNOWLEDGMENTS

We want to acknowledge the funding on NSF grant DRL-1109483 as well as funding of ASSISTments. See here (http://www.webcitation.org/67MTL3EIs) for the funding sources for ASSISTments.

REFERENCES

[1] Anderson, J.R., Rules of the Mind. Lawrence Erlbaum (1993).

[2] Wang, Y., & Beck, J. E. (2012). Using Student Modeling to Estimate Student Knowledge Retention. In Proceedings of the 5th International Conference on Educational Data Mining, 176-179.

[3] Corbett, A. T., & Anderson, J. R. (1994). Knowledge tracing: Modeling the acquisition of procedural knowledge. User modeling and user-adapted interaction, 4(4), 253-27

Predicting Group Programming Project Performance using SVN Activity Traces

Sen Liu
USC Information Sciences Institute, 4676 Admiralty Way
Marina del Rey CA 90292
+1 213 880 8363
senliu@usc.edu

Jihie Kim
USC Information Sciences Institute, 4676 Admiralty Way
Marina del Rey CA 90292
+1 310 448 8769
jihie@isi.edu

Sofus A. Macskassy
USC Information Sciences Institute, 4676 Admiralty Way
Marina del Rey CA 90292
+1 310 448 8243
sofmac@isi.edu

Erin Shaw
USC Information Sciences Institute, 4676 Admiralty Way
Marina del Rey CA 90292
+1 310 448 9196
shaw@isi.edu

ABSTRACT
This paper presents a model for integrating student activity traces in a collaborative programming project using SVN, and relates different attributes of the SVN activities to student and team performance. We show how student participation patterns can be related to the grades of their group programming projects. Graph theory, entropy analysis and statistical techniques are applied to process and analyze data.

Keywords
Collaborative project, group project, SVN, data mining, entropy analysis, graph theory

1. INTRODUCTION

The goal of this case study is to make progress towards understanding the impact of collaboration on individual and group performance in programming courses that use a collaborative code management system, such as SVN (Subversion), which supports team-based programming projects by providing a complete history of individual programming activities. Past studies of group work have analyzed how the characteristics of team members affect group outcomes [1] and whether certain members or leaders influence performance [2]. Related work by the authors has shown that team pacing is highly correlated to group project performance [3]. Building on these results, we explore the following questions:

- Does individual coursework performance affect the group project performance?
- Does the most interactive (or influential) student affect the group project performance?
- Does even work pacing affect the group work performance?

Results indicate that when integrating components from different members, teamwork skills and usage of teamwork tools may improve the group performance; however, for implementing difficult programs, individual members' programming skills become more important. The performance of leaders or central students can affect the group performance greatly, and work pacing and management of the work throughout the project period can be an important fact for a successful team programming.

2. STUDY CONTEXT

To better prepare students for professional employment, two undergraduate computer science teachers at the University of Southern California combined a first and second year course so that students could work on an authentic project. This case study of that experiment spans a seven-week period of collaboration among students in the two classes.

2.1 Group Project Description

Students from the two courses formed 19 groups, each of which had 3 to 11 members from cs200 (freshmen) and 4 to 6 members from cs201 (sophomore). Each group designed and built a manufacturing assembly cell. The freshmen implemented the front-end and sophomores implemented the back end code. The students cooperated in designing the API between the two. The project had four subtasks: To design the project; to implement the components of the program (V0); to integrate the components (V1); and to implement non-normative case handling (V2). For teamwork planning and documenting, students made use of co-authoring tools. To manage code development, they used Apache Subversion (SVN). This work focuses on the SVN activities.

2.2 Data Description

SVN data from two semesters, 2011 spring and 2011 fall, was used for the analysis. Table 1 gives an overview of the data including students per team, number of files and number of file modifications made by each team.

Table 1. Summary of SVN data used for analysis.

	Group	N of Students	N of Files	N of File Mods
2011 FALL	M1	16 (5,11)	4007	5333
	M2	14 (5,9)	4007	6142
	T1	13 (4,9)	474	1433
	T2	13(4,9)	1740	3173
	W1	13 (5,8)	1603	2856
	W2	13 (5,8)	1412	3288
	W3	14 (6,8)	1994	3845
	W4	14 (6,8)	2082	3357
	W5	13 (5,8)	2156	3873
2011 SPRING	M1	11 (6,5)	2919	5885
	M2	10 (5,5)	3332	5276
	M3	11 (5,6)	1737	3243
	M4	9 (6,3)	2992	4279
	T1	10 (5,5)	1770	3370
	T2	12 (5,7)	1301	2871
	T3	10 (4,6)	1096	1842
	W1	12 (7,5)	5711	7287
	W2	9 (5,4)	1186	2184
	W3	11 (6,5)	2444	4137

Group project grades, other student performance (student exam and coursework grades), and SVN activity was used to analyze collaboration. The project grades of a group were computed by averaging the grades of its group members. CS200 had three assignments and two exams while CS201 had no assignments and two exams. SVN activity was measured by three variables that represent the degree of participation and collaboration in file co-editing: 1) The number of files she/he modified or added each day, 2) The number of lines of code she/he modified or added each day, and 3) The number of interactions she/he made with each of the other group members in her/his group (if two students modify the same file, there is one interaction between them. If two students modify more than one file, then each "co-modified" file counts as an interaction.

3. DATA ANALYSIS AND RESULTS

This section uses graph theory and entropy analysis to explore participation patterns and their relation to project performance.

3.1 Relationship between influential student performance and group performance

Next we looked at the relationship between influential students and group performance to determine whether the performance of some members has an impact on group performance. Social Network Analysis (SNA) [4] has for decades been analyzing social networks to identify and categorize important people and has defined a variety of "centrality" metrics to identify different types of importance. We used three of these metrics for analysis: degree centrality, closeness centrality and betweenness centrality.

Degree centrality: The degree centrality metric counts the number of relations a person has (figure 1a). The more relations, the more important that person is because she/he talks to more people. A student who co-modifies code with many others might positively impact the project grade and is assigned a high degree.

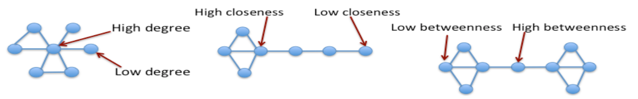


Figure 1. High/low (a) degree, (b) closeness & (c) betweenness

Closeness centrality: This metric identifies people based on how close they are, on average, to everybody else (figure 1b). To compute closeness, we define the *shortest distance, d(v,t)* to be the fewest number of edges needed to traverse from node *v* to node *t*. A student who is closest to everyone might have a lot of influence in the group.

Betweenness centrality: The last metric we consider in this study is the betweenness centrality (figure 1c). People with high betweenness are also known as "bridges" or "brokers" in that they sit between groups that otherwise do not have a lot of interaction. Such a student might be the one to help integrate the front- and back-ends and might positively impact the success of the project.

To compute this metric, we must first compute to what extent a students is a bridge. This is done by computing a shortest path for each pair of vertices. The *betweenness centrality* of node *v* is the number of shortest paths that go through a particular node. We generated one graph per group. Each vertex is a student and an edge indicates one or more interactions between them. Almost every pair of group members had a few interactions, so we only drew an edge between students with more than 10 interactions. After generating the graph, we computed the three metrics described above for each student. If stronger students are more central then we expect their projects to do better.

Two group graphs are shown in figure 2. The larger a vertex is, the larger the corresponding degree centrality. Nodes in red are cs200 students. Nodes in green (with an extra circle), yellow (square) and blue (diamond) are cs201 students. The yellow node (square) is the cs201 student with the best exam1 score. The blue node (diamond) is the cs201 student with the best exam2 score. The left group was one of the better performers (project grades were 92.31 for cs200 and 94.75 for cs201). We have color-coded the nodes based on cs200/cs201 breakdown as well as the two best cs201 students (best exam scores). We find that cs201 students are more central than cs200 students and see closer interaction between the cs201 students. Finally, note that the two best cs201 performers are also the two most central nodes.

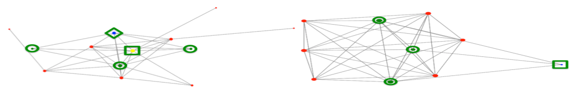


Figure 2. High (left) and low (right) performing groups.

In comparison, consider the graph on the right, which represents a group that performed less well (project scores were 88.63 for cs200 and 87.6 for cs201). We see a dramatic difference in the graph. First, although 3 of 4 cs201 students are quite central, the best cs201 student is not at all engaged (blue node to the far right). All cs200 students are engaged quite well. The blue node to the far right is the cs201 student with the best scores for both exams. These two group graphs seem to indicate that group structure, and in particular, the location of the better students, might significantly impact project grades.

3.2 Relationship between SVN activity and group performance

Finally, we look at the relationship between SVN coding activity and group project grades. We hypothesized that groups that work consistently will have a better grade than groups that do most of their work right before a deadline. To test this hypothesis, we turned to the information theoretic function of *entropy*. Entropy measures the amount of uncertainty in a system, or in our case, how much the activity of a group is spread throughout the project timeline Groups that are consistently active throughout their project will have high entropy, whereas groups that have a spike in activity towards the deadline will have low entropy. The entropy of a group's activity was based on the number of modifications. For each submission, the correlation and p-values between entropy and project grade (cs200+cs201) were computed. The correlations (p-values) between entropy and project grades were v0: 0.24 (0.33), v1: 0.59 (0.007), and v2: -0.37 (0.12). For v1, there is a significant positive correlation between entropy (working continuously) and group project grade; however, for v0 and v2, the p-values are large, which means that entropy and group performance are likely to be independent.

4. ACKNOWLEDGEMENT

The authors thank USC CS faculty Drs. David Wilczynski and Michael Crowley for their assistance. The research was supported by a grant from the National Science Foundation (#0941950).

5. REFERENCES

[1] Michaelsen, L.K., Sweet, M. (2008), The Essential Elements of Team-Based Learning, New Directions for Teaching and Learning, n116 p7-27 Win 2008.

[2] Strijbos, J. W. (2004). The effect of roles on computer supported collaborative learning, Open Universiteit Nederland, Heerlen, The Netherlands (Chapters 3 and 4).

[3] Ganapathy, C., Shaw, E. & Kim, J. (2011) Assessing Collaborative Undergraduate Student Wikis and SVN with Technology-based Instrumentation: Relating Participation Patterns to Learning, Proc. of the American Society of Engineering Education Conference, 2011.

[4] Wasserman, S. & Faust., K. (1994). Social Network Analysis. Cambridge: Cambridge University Press, 1994.

Toward Predicting Test Score Gains With Online Behavior Data of Teachers

Keith E. Maull
University of Colorado
Institute of Cognitive Science
Boulder, Colorado USA
keith.maull@colorado.edu

Tamara Sumner
University of Colorado
Institute of Cognitive Science
Boulder, Colorado USA
tamara.sumner@colorado.edu

ABSTRACT
As technology continues to disrupt education at nearly all levels from K–12 to college and beyond, the challenges of understanding the impact technology has on teaching continue to mount. One critical area that yet remains open, is examining teachers' usage of technology by specifically collecting detailed data of their technology use, developing techniques to analyze that data and then finding meaningful connections that may show the value of that technology. In this research, we will present a model for predicting test score gains using data points drawn from typical educational data sources such as teacher experience, student demographics and classroom dynamics, as well as from the online usage behaviors of teachers. Building upon prior work in developing a usage typology of teachers using an online curriculum planning system, the Curriculum Customization Service (CCS), to assist in the development of their instruction and planning for an Earth systems curriculum, we apply the results of this typology to add new information to a model for predicting test score gains on a district-level Earth systems subject area exam. Using both multinomial logistic regression and Naïve Bayes algorithms on the proposed model, we show that even with a simplification of the highly complex tapestry of variables that go into teacher and student performance, teacher usage of the CCS proved valuable to the predictive capability in average and above average test score gains cases.

Keywords
online user behavior, teaching, pedagogy, learner gain prediction, instructional planning support

1. BACKGROUND
Within the past 20 years, the use of technology in the classroom has grown at an unimaginable pace. From K–12, to college and lifelong learning, students and teachers alike are now using a vast array of tools in their educational endeavors. Teachers, especially, are using tools in many interesting ways with the hope that these tools improve their teaching productivity, better engage their learners, and ultimately provide optimizations that make their jobs less difficult so that they can maximize their value and skill in teaching. Tools that educators once relied on to enter grades and organize lesson plans, have transformed into the now-diverse online ecosystem of intra-, extra- and Inter-net based platforms that allow them to do a multitude of activities like collaborate with like-minded educators located anywhere in the world, find digital resources relevant to their curricular objectives, find out how state and national standards are tied to specific resources prepared for use in their classroom, examine the progress their students are making through those lessons and even manage all these things in a single portal. These technologies are affecting everyone in education – administrators, educators, learners, parents, etc. – and as the state of the art pushes policy and pedagogy forward, in its wake a mounting number of challenges must be sorted out, including whether or not these technology tools are facilitating educational productivity or hindering it.

It is widely recognized that teachers matter a great deal in the learning process of students, and many studies suggest that teacher skill is one of the key predictive forces in learner success. Even with large efforts such as The Gate's Foundation $45 million dollar, multi-year study to understand the factors that might accurately predict teacher effectiveness, it still remains unclear from these and many other studies, what impact technology and specifically online tools designed to impact pedagogy, are having on the toolkits, skillsets and patterns of productivity employed by effective teachers, where effectiveness in this context is measured by learner gains. This research aims to explore the mechanisms and models of understanding how teacher utilization of online tools might be linked with learner gains. By studying the usage of the Curriculum Customization Service (CCS), we will try to bridge the gaps between the online behaviors of teachers and learner gains, while at the same time utilizing common educational data, such as teacher experience, class demographics and class dynamics variables.

2. RESEARCH CONTEXT
The research presented here examines the online usage behaviors of teachers within the context of the learning gains observed over the course of a single year of data. Earth systems teachers within a large urban public school district within the U.S. were trained and given the Curriculum Customization Service (CCS) to use for their planning

and instructional tasks during the 2009-2010 school year. The CCS has been described elsewhere in detail [2], but summarily, the CCS is an online instructional and curriculum planning tool designed to provide teachers access to an array of materials to support them throughout the school year. It contains publisher materials (e.g. digital versions of publisher books, student handouts), including instructional supports for classroom activities, as well as digital library resources that have been vetted and aligned to the curricular goals and objectives of the district. The content and structure of the CCS closely match the goals of objectives of the district-designed curriculum, and for this research the curricular focus of the CCS materials was the grade 9 Earth systems curriculum.

As previously reported in [1], the usage patterns of the CCS users have been automatically explored using unsupervised clustering techniques yielding a user typology representing different kinds of aggregate use of the CCS over the span of a single year of use. It was found that five categories of usage emerged from the data, briefly described in Table 1. These usage categories form the basis for the online component of this research, and these automatically generated categories do not in any way represent fixed, or even definitive categories in all instances and users of the CCS (or any other system). The usage types discovered by these methods, however, were studied further through qualitative analysis of teacher surveys, interview and in-class observations, and have been given additional support and validation through other research [2].

Typology Label	User Characteristics
Limited Use	Lower overall use of the system.
Interactive Resources specialist	Heavier relative use of the Interactive Resources components of the system.
Power user	Heavy robust use of the CCS.
Moderate generalist	Overall moderate use of the system.
Community-seeker	Heavier relative use of the community and sharing features of the CCS.

Table 1: User typology for CCS users in this research.

The impetus of this research is therefore to explore whether the CCS can be shown to have a predictive association with *classroom learner gains* through the user typology give above. In particular, this research examines how the typology designations along with three crucial inputs (1) the demographic composition of the learners within each teacher's classroom, (2) the dynamics of each classroom in this research, and (3) the skill level of the teachers, might be used to build a model to predict learner gains.

3. METHOD

By using the pre- and post-test scores of the district-wide Earth science Benchmark exam administered to students at the beginning and end of the school year, this research aims to build and apply a model (see section 4) that examines the linkages between CCS usage and the other variables collected in this research. Two years (2008–09 and 2009–10) of standardized test and Benchmark exam score data, teacher experience and class data (class size and demographic makeup) were examined to determine if there were any significant differences in the population characteristics and student test score performance. For the first year data of data (2008–09) the CCS was not used, yet in the successive year (2009–10) the CCS was used. Though there were significant differences between the Benchmark exam score gains ($\alpha < .001$; $df = 2$; $\chi^2 = 1039$; $p = 2.2e^{-16}$), where the mean letter grade gain in 2008–09 Benchmark exam was 0.29 and the mean letter gain in 2009–10 was 1.04, the other variables such as demographic makeup, class size and teacher experience, show no significant differences.

4. DATA & MODEL

Classroom and teacher data were segmented and binned for 27 teachers for which there was complete, comparable data sets as well as CCS usage data in the 2009–10 school year. These data represented 81 total class sections of students for that year. To explore the role the CCS may have played in the Benchmark exam gains of the 2009–10 school year, a model was created to predict learner gains with demographic, class size and teacher skill variables, in addition to the CCS usage typology category previously discovered in section 2. This model for predicting gain $G_{s,t}$ is given by $G_{s,t} \sim E+D+N+U$, where E is the teacher experience category, D is the demographic category, N the class size category and U the CCS usage category. The output variable $G_{s,t}$ was predicted along three outcomes : below average, average and above average gains. Data sets and outcomes were further grouped into the gains for all sections and gains for individual sections.

5. INVESTIGATION & RESULTS

Two classifiers were compared, Naïve Bayes and multinomial logistic regression, in several different configurations to study the predictive capability of model with and without CCS usage. The best model sensitivity (true positive rate) achieved by all models examined was 0.67 for the average and above average gains, corresponding to $G_(s,t) \sim U + N$ and $G_(s,t) \sim U + D$ – CCS usage category + class size and CCS usage + demographic category, respectively. The model performed poorly at predicting below average gains for the individual section gain grouping.

6. ACKNOWLEDGEMENTS

This material is based upon work supported in part by the NSDL program in the National Science Foundation under Awards #0734875 and #0840744 and by the University of Colorado at Boulder.

7. REFERENCES

[1] K.E. Maull, M.G. Saldivar, and T. Sumner. Understanding digital library adoption: a use diffusion approach. In *Proceeding of the 11th annual international ACM/IEEE joint conference on Digital libraries*, pages 259–268. ACM, 2011.

[2] M. G. Saldivar. *Teacher Adoption of a Web-Based Instructional Planning System.* PhD thesis, School of Education, University of Colorado, Boulder, CO, 2012.

Domain-Independent Proximity Measures in Intelligent Tutoring Systems

Bassam Mokbel
CITEC Center of Excellence
Bielefeld, Germany
bmokbel@techfak.uni-bielefeld.de

Sebastian Gross
TU Clausthal
Clausthal-Zellerfeld, Germany
sebastian.gross@tu-clausthal.de

Benjamin Paassen
CITEC Center of Excellence
Bielefeld, Germany
bpaassen@techfak.uni-bielefeld.de

Niels Pinkwart
TU Clausthal
Clausthal-Zellerfeld, Germany
niels.pinkwart@tu-clausthal.de

Barbara Hammer
CITEC Center of Excellence
Bielefeld, Germany
bhammer@techfak.uni-bielefeld.de

ABSTRACT

Intelligent tutoring systems (ITSs) typically analyze student solutions to provide feedback to students for a given learning task. Machine learning (ML) tools can help to reduce the necessary effort of tailoring ITSs to a specific task or domain. For example, training a classification model can facilitate feedback provision by revealing discriminative characteristics in the solutions. In many ML methods, the notion of proximity in the investigated data plays an important role, e.g. to evaluate classification boundaries. For this purpose, solutions need to be represented in an appropriate form, so their (dis-)similarity can be calculated. We discuss options for domain- and task-independent proximity measures in the context of ITSs, which are based on the ample premise that solutions can be represented as formal graphs. We propose to identify and match meaningful contextual components in the solutions, and present first evaluation results for artificial as well as real student solutions.

1. INTRODUCTION

Intelligent tutoring usually relies on knowledge about the domain being taught and adaptation of pedagogical strategies regarding learners' individual needs. Therefore, ITSs typically use formalized domain knowledge to provide intelligent one-on-one computer-based support to students. Often, even the specific learning task must be modeled explicitly, which requires significant effort by human experts. Hence, among several directions of research, one major idea regards ITSs which are adaptive, based on examples and using ML or data mining techniques, rather than an explicit modeling of the background information. This direction also opens a way towards the application of ITSs in domains where a formalization of the underlying knowledge is hardly possible, such as ill-defined domains in which there may exist a wide variety of strategies for solving a given task [4]. Example-based learning has shown to be an effective tutoring approach in supporting learning, see [1], which can also be applied without formalizing domain knowledge. In [2], the authors propose ways how feedback provision can be realized in example-based learning environments.

Acknowledgments: This work was supported by the German Research Foundation (DFG) under the grant "FIT - Learning Feedback in Intelligent Tutoring Systems." (PI 767/6 and HA 2719/6).

Assuming that effective feedback-strategies can be established based on appropriate examples, let us restrict to a scenario where a set $\widetilde{X}$ of examples $\tilde{\mathbf{x}}^j$ is explicitly given, and a student solution $\hat{\mathbf{x}}^i$ from the set $\widehat{X}$ needs to be associated to the most suited example. We further assume that examples are themselves solutions (or are represented in the same form) and we can process them in the same manner. Let $d(\mathbf{x}^i, \mathbf{x}^j)$ be a meaningful proximity measure which indicates the dissimilarity of any two solutions $\mathbf{x}^i, \mathbf{x}^j \in X = \widehat{X} \cup \widetilde{X}$ by a positive value. Then, a student solution $\hat{\mathbf{x}}^i \in \widehat{X}$ can simply be associated to the most similar (and thus most suited) example by choosing: $\arg\min_j \; d(\hat{\mathbf{x}}^i, \tilde{\mathbf{x}}^j), \; j \in \{1, \ldots, |\widetilde{X}|\}$. In the following, we will present general approaches to calculate this dissimilarity, if solutions are represented as formal graphs with annotations. The overall calculation is not tailored to a specific learning task or domain, if general data representations are used. To explain the details of the approach, and show first experimental results, we will refer to our example application scenario: an ITS to support programming courses for the Java language.

2. THE PROXIMITY OF SOLUTIONS

The basic requirement is that solutions can be represented as graphs, with different kinds of meta information annotated on the nodes and edges. Each node represents a (syntactic or semantic) element of the solution, and edges establish relationships between them. Solutions $\mathbf{x}^i \in X$ are thus graphs $G_i = (V_i, E_i)$. Considering a very simple annotation, we require that all vertices are attributed to a certain *node type*, a symbol from the finite alphabet $\Sigma = \{l_1, \ldots, l_T\}$. In our application scenario, we consider *syntax trees* of Java programs, where the nodes represent syntactic elements and the corresponding node types indicate their functionality, e.g. the declaration of a variable, a logical expression, a variable assignment, etc. Additionally, a parser adds edges denoting relationships between the syntactic elements, like the call to a function, the usage of a variable, etc.

Using classical data mining approaches, there are several ways to define a proximity measure for these annotated graphs. For example, to represent a solution by a *feature vector*, one can extract frequencies of syntax elements within a solution vs. all solutions to gain a representation analogous to popular *tf-idf weights* [5]. By $d_{\texttt{tfidf}}(\mathbf{x}^i, \mathbf{x}^j)$ we refer to the

Euclidean distance between the tf-idf weight vectors of two Java syntax graphs. This kind of feature encoding captures statistics about the symbols, however their relations are not considered. Measures for *symbolic sequences* respect the ordering of symbols, e.g. alignment measures [3]. For this purpose, we can encode the syntax trees as sequences $\mathbf{s}^i \in \Sigma^*$ by visiting vertices in a depth-first-search order, concatenating their node types. This ordering corresponds to the original sequence of statements in the Java source code. Let $d_{\texttt{align}}(\mathbf{x}^i, \mathbf{x}^j)$ be the dissimilarity score of a Smith-Waterman *local alignment* of the respective sequences $\mathbf{s}^i$ and $\mathbf{s}^j$, with fixed costs for edit operations, see [3].

Both, tf-idf weights and alignment do not explicitly consider contextual relationships within the syntax. To take these structural characteristics into account, we propose to identify densely connected subgraphs and calculate a piecewise proximity between parts of the solutions. Our basic assumption is that solutions consist of semantic building blocks, in which the elements are in close relation to each other, and are less related to other elements. For example, in a computer program the variables and expressions within a for-loop are likely to be connected to each other, but would be less connected to elements outside the loop. In the following, we call these dense subgraphs *fragments*. To identify such fragments, we use the graph clustering algorithm *Spectral Clustering* (SC) [6]. SC separates the graph into a fixed number of subgraphs, and as a result, we get an assignment of every node to one out of m identified fragments $\{F_1^i, \ldots, F_m^i\}$ of the graph G_i for solution $\mathbf{x}^i$, where $\bigcup_{s=1}^m F_s^i = G_i$ and $F_s^i \cap F_t^i = \emptyset \;\; \forall s,t \in \{1,\ldots,m\},\; s \neq t$.

The goal of fragmenting the graph is to compare distinctive parts of the solutions independently. To evaluate the dissimilarity of a single pair of fragments (F_s^i, F_t^j) with $i \neq j$ and $s,t \in \{1,\ldots,m\}$, one can rely on established proximity measures from the literature, as exemplified by $d_{\texttt{tfidf}}$ or $d_{\texttt{align}}$. This requires only that each fragment can be represented individually to apply the respective measure, e.g. as a string, a numeric vector, etc. We call this the *signature* of the fragment. Let $d(F_s^i, F_t^j)$ be the dissimilarity of the fragment pair. To compare the two underlying solutions $\mathbf{x}^i$ and $\mathbf{x}^j$ as a whole, m suitable pairs of fragments (F_s^i, F_t^j), $(s,t) \in M \subset \{1,\ldots,m\}^2$ have to be established for comparison. Since we want those pairs to yield the best overall match, i.e. the minimum sum of dissimilarities, we arrive at an optimal matching problem. For now, we use a simple greedy heuristic to gain an approximate matching M of all fragments. We then compute the overall dissimilarity as the mean $\delta(F^i, F^j) = \frac{1}{m}\sum_{(s,t)\in M} d(F_s^i, F_t^j)$.

3. EVALUATION AND CONCLUSION

We use three datasets consisting of Java programs, where a semantically meaningful class separation is given in each set. To quantify the discriminative quality of the different proximity measures, we report the results of a simple *k-nearest-neighbor* (k-NN) classifier w.r.t. this class structure, see Tab. 1. The *Artificial* dataset consists of 48 programs created by a human expert and serves as a basic testbed. The 48 programs all solve the simple task to decide whether all words in a given input sentence are palindromes. The programs are deliberately designed to form 8 groups of 6 solutions each. Each group represents a distinct approach to solve the task, resulting from a combination of 3 simple (binary) design choices: using Java utility functions or not, using String objects or arrays of characters, splitting the sentence into words or iterating over the whole input sequence. Within each group, the 6 programs are only slightly different, with altered syntactic details like variable names and the sequence of operations. The *'Tasks'* dataset consists of 438 real student solutions, collected during 3 different programming exams for business students. Each solution was provided by an individual student, and the data is class-labeled according to the 3 different tasks assigned in the respective exam: [I] implementing Newton's method to find zeros in 2nd order polynomials (144 solutions); [II] calculating income tax for a given income profile (155); and [III] checking if a given sentence contains a palindrome, and if the sentence is a pangram (139). The *'TextCheck'* dataset consists of 68 student solutions which solve the above-mentioned task [III]. Here, class labels were provided by tutoring experts who were asked to determine meaningful groups in the solution set. The experts distinguished them according to 3 design choices very similar to the ones used in the *Artificial* set (which was subsequently created). This resulted in 8 classes corresponding to distinct strategies to solve the task. In general, solutions are very heterogeneous and classes are highly imbalanced. Therefore, this dataset represents a state-of-the-art challenge for a real ITS.

After preprocessing as described, we applied four variants of proximity measures, evaluating the accuracy of a 3-NN classifier, see Tab. 1. The results show that with all measures the solutions from the *Artificial* and *Tasks* dataset are classified rather reliably, which indicates that the measures are semantically meaningful. Accuracies for the *TextCheck* data are generally low, as expected from the challenging scenario. However, the measures based on fragmentation, $\delta_{\texttt{tfidf}}$ and $\delta_{\texttt{align}}$, showed a performance increase as compared to their simpler counterparts $d_{\texttt{tfidf}}$ and $d_{\texttt{align}}$. A rigorous evaluation of the approach is the subject of ongoing work.

Dataset	#fragments	$d_{\texttt{tfidf}}$	$\delta_{\texttt{tfidf}}$	$d_{\texttt{align}}$	$\delta_{\texttt{align}}$
Artificial	$m = 4$	0.94	0.92	0.94	0.94
Tasks	$m = 6$	0.98	0.83	0.98	0.98
TextCheck	$m = 6$	0.34	0.32	0.41	0.54

Table 1: Classification accuracies of a 3-NN classifier with different proximity measures on the experimental datasets. The number of fragments m in the measures $\delta_{\texttt{tfidf}}$ and $\delta_{\texttt{align}}$ was chosen with regard to the average size of graphs in the respective dataset.

4. REFERENCES

[1] T. Gog and N. Rummel. Example-based learning: Integrating cognitive and social-cognitive research perspectives. *Edu. Psych. Rev.*, 22:155–174, 2010.

[2] S. Gross, B. Mokbel, B. Hammer, and N. Pinkwart. Feedback provision strategies in intelligent tutoring systems based on clustered solution spaces. In *DeLFI 2012*, pages 27–38. Köllen, 2012.

[3] D. Gusfield. *Algorithms on Strings, Trees, and Sequences.* Cambridge University Press, 1997.

[4] C. Lynch, K. D. Ashley, N. Pinkwart, and V. Aleven. Concepts, structures, and goals: Redefining ill-definedness. *Int. J. of A.I. in Edu.*, 19(3):253 – 266, 2010.

[5] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Inf. Process. Manage.*, 24(5):513–523, Aug. 1988.

[6] U. von Luxburg. A tutorial on spectral clustering. *Stat. Comput.*, 17(4):395–416, 2007.

Exploring Exploration: Inquiries into Exploration Behavior in Complex Problem Solving Assessment

Jonas C. Müller
jonas.mueller@uni.lu

André Kretzschmar
andre.kretzschmar@uni.lu

Samuel Greiff
samuel.greiff@uni.lu

University of Luxembourg
6, rue Richard Coudenhove Kalergi
1359 Luxembourg-Kirchberg

ABSTRACT
Complex Problem Solving (CPS) is a prominent representative of transversal, domain-general skills, empirically connected to a broad range of outcomes and recently included in large-scale assessments such as PISA 2012. Advancements in the assessment of CPS are now calling for a) broader assessment vehicles allowing the whole breadth of the concept to unfold and b) additional efforts with regard to the exploitation of log-file data available. Our Paper explores the consequences of heterogeneous tasks with regard to the applicability of an established measure of strategic behavior (VOTAT) featured currently in assessment instruments. We present a modified conception of this strategy suitable for a broader range of tasks and test its utility on an empirical basis. Additional value is investigated along the line of theory driven educational data mining of process data.

Keywords
Complex Problem Solving, Theory Driven Data Mining, Exploration Behavior.

1. INTRODUCTION
Targeting human behavior in problem situations characterized by dynamic and interactive features [1], widespread application of Complex Problem Solving (CPS) assessment only began after the introduction of formal frameworks and the restriction to so-called minimal complex systems [3]. Recent inclusions of CPS as a representative of domain-general skills in large-scale studies such as the Programme for International Student Assessment (PISA) in 2012 can be seen as a direct result of these advancements.

1.1 CPS assessment with finite-state automata
Initially allowing for the reliable assessment of CPS, thereby building towards the foundation of the success of the concept, the restriction to tasks based on the formal framework of Linear Structural Equations (LSE) is at the same time restrictive with regard to the kind of problems that can be modeled: Only *quantitative* relations between variables can be simulated in assessment. This restriction can be hindering in several ways as it is limiting the necessary behavior for successful handling of the problem. Therefore, current work in the domain [3, 4] is targeting the inclusion of tasks based on a second formal framework: Finite State Automata (FSA), thereby (re-)introducing a whole range of problem features to task construction and assessment [2].

1.2 Strategies in CPS assessment
The availability of process data documenting participant's behavior while working on a CPS task presents an ideal point of departure for educational data mining along the line of Sao Pedro et al. [5]. The problem lies with defining meaningful patterns, that can subsequently be used in text replay tagging [5]: Basic analysis based on behavioral measures like the number of interactions or time working on a task has been rather disappointing from the view of explaining variations in CPS performance. Fortunately enough, a strategy called vary-one-thing-at-a-time (VOTAT, also called control-of-variables-strategy, CVS) can be considered optimal for exploring LSE-based tasks [3]. Process analysis built on this strategy has been widely employed in CPS assessment.

1.3 Adapting VOTAT to FSA-based tasks
The introduction of FSA-based tasks, however, is making the analysis and scoring of VOTAT-behavior insufficient. In some cases, a participant solely applying VOTAT will not even reach essential states for the task's understanding. To account for this change in optimal strategic behavior, we adapted the search for adequate behavior to the possibilities of FSA-based tasks.

Tasks based on LSE show underlying relations that stay the same during the whole course of working on them: The effects of increasing variable A will always result in a reduction of the value in variable Y, making VOTAT the optimal exploration strategy. In FSA-based tasks on the other hand, the effect of increasing an input variable might not only depend on that variable, but also on the input values of other variables, specific arrangements of inputs (e.g., one high, one low), or prior output variable values.

As a result, adequate explorative behavior has to take the state into account the task is currently in: A resulting behavioral pattern, we call nested-VOTAT, is referring to different 'areas' of the system in which input variations are showing qualitatively comparable effects. Nested-VOTAT is shown, when isolated input variations (i.e., a VOTAT-like behavior) are used in these areas: Systematically exploring the effects of input variation in isolation and for all input variables within an area of similar task functioning. This scoring can subsequently be used in procedures like text replay tagging [5].

1.4 Research question
We are interested in the applicability and empirical usefulness of this extension of VOTAT for FSA-based tasks, especially when empirically related to classical tasks based on LSE. The approach is mixing exploratory elements of data analysis with theoretically driven assumptions.

2. METHOD

Participants and procedure: Participants provided data in the context of a broader assessment at a German school in November 2012. There were 565 students who completed the relevant assessment of CPS, 54% of participants were female, mean age was 14.95 years (*SD* = 1.30).

Measures: *FSA*: Five LSA-based tasks were included in the assessment. The test is called MicroFIN [4], 'Micro' standing for the minimal complex systems approach and 'FIN' for the framework of finite state automata. Separate phases of MicroFIN are targeting knowledge acquisition and knowledge application. Empirically, we expect a three-dimensional measurement model for MicroFIN, separating the application of nested-VOTAT, knowledge acquisition, and knowledge application.

Scoring of nested-VOTAT: Credit for nested-VOTAT is given per task for the isolated variation of all input variables in one of the qualitatively different areas of the task. The number of these areas varies with the number of states the problem shows different relations to input variations for.

LSE: To get a comparable measure of CPS based on LSE, eight MicroDYN tasks were included in testing. MicroDYN [3] includes a scoring of VOTAT in the exploration phase, which is typically strongly connected to knowledge acquisition. A two-dimensional measurement model is expected, conflating strategy application (i.e., VOTAT) and knowledge acquisition [3].

3. RESULTS

Manifest Correlations: Manifest correlations between nested-VOTAT indicators and knowledge acquisition and knowledge application in MicroFIN were of rather low size per task. This result is in line with previous findings with regard to single task indicators of MicroFIN and MicroDYN.

Dimensionality and internal consistency: Exploratory factor analysis for nested-VOTAT indicated a single latent factor. Internal consistency was poor with Cronbach's alpha $\alpha = .53$.

Latent modeling: *Measurement models:* Measurement models for MicroFIN indicated the separability of nested-VOTAT application and knowledge acquisition, as well as a third dimension for knowledge application. The three dimensional model fitted well, conflating any of the two dimensions resulted in significantly worse model fit (χ^2 (167) = 230.246, $p < .001$, RMSEA = 0.026, CFI = 0.981, TLI = 0.978; latent factors correlated $r = .71$ to .81, all $p < .001$). Measurement models for MicroDYN indicated the expected two-dimensional model fitting well (χ^2 (251) = 588.234, $p < 0.001$, RMSEA = 0.049, CFI = 0.994, TLI = 0.993, latent correlation of factors $r = .79$, $p < .001$). Separating the use of VOTAT from knowledge acquisition in MicroDYN resulted in estimation problems due to very highly correlated factors.

Structural models: Relations between latent indicators for MicroFIN and MicroDYN can be found in Table 1.

Latent correlations between MicroFIN and MicroDYN ($r = .56$ to .68) were slightly lower than the internal correlations between facets within each instrument ($r = .71$ to .81). No major difference with regard to the facets could be found: Knowledge acquisition facets were not stronger related to each other, than to the knowledge application facet of the other instrument. The facet indicating the application of nested-VOTAT in MicroFIN did show a stronger connection to knowledge acquisition, than knowledge application in MicroFIN ($p < .001$).

Table 1: Latent correlations of MicroFIN and MicroDYN

		MicroFIN			MicroDYN
Test	Facet	(1)	(2)	(3)	(4)
MicroFIN	(2)	0.81			
	(3)	0.80	0.71		
MicroDYN	(4)	0.66	0.64	0.61	
	(5)	0.68	0.56	0.62	0.78

Note: (1) Knowledge acquisition in MicroFIN, (2) Knowledge application in MicroFIN, (3) Use of nested-VOTAT in MicroFIN, (4) Knowledge acquisition in MicroDYN (incl. use of VOTAT), (5) Knowledge application in MicroDYN. All correlations are significant on a .01 level.

4. DISCUSSION

The present study represents a first advance into utilizing the potential of process data in FSA-based tasks. It shows the feasibility of including measures of exploration behavior into the assessment of CPS even when based on heterogeneous tasks.

The notion of nested-VOTAT has been shown to be applicable to a range of tasks based on FSA with a varying degree of success: The low internal consistency of $\alpha = .53$ could mean different optimal strategies being necessary in some of the tasks, as it could be related to measurement problems due to the high heterogeneity of tasks. A broader variation of item features and alternative strategy scorings is needed to clarify this aspect.

The latent correlations point to the potential of including process-related measures into FSA-based CPS assessment: Contrary to the case of LSE-based tests, the higher heterogeneity of FSA seems to result in a more probabilistic relation between strategies in exploration and knowledge acquisition.

Future inquiries into the determinants of CPS performance should be built on broad assessment vehicles and careful examination of participant behavior. Based on the findings elaborated here, approaches of educational data mining like text replay tagging can be utilized even when targeting a heterogeneous set of tasks.

5. REFERENCES

[1] Buchner, A. 1995. Basic topics and approaches to the study of complex problem solving. *Complex problem solving: The European perspective*. P.A. Frensch and J. Funke, Eds. 27–63.

[2] Buchner, A. and Funke, J. 1993. Finite-state automata: Dynamic task environments in problem-solving research. *The Quarterly Journal of Experimental Psychology Section A*. 46, 1 (1993), 83–118.

[3] Greiff, S., Wüstenberg, S. and Funke, J. 2012. Dynamic Problem Solving: A new assessment perspective. *Applied Psychological Measurement*. 36, 3 (2012).

[4] Müller, J.C., Kretzschmar, A., Wüstenberg, S. and Greiff, S. 2013. Extending the Assessment of Complex Problem Solving to Finite State Automata: The Merits of Heterogeneity. Manuscript submitted for publication.

[5] Sao Pedro, M.A., Baker, R.S.J.d., 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 (2013), 1–39.

The Complex Dynamics of Aggregate Learning Curves

Tristan Nixon, Stephen Fancsali, and Steven Ritter
Carnegie Learning, Inc.
437 Grant Street, Suite 918
Pittsburgh, PA 15219, USA
(888) 851.7094 {x123, x219, x122}
{tnixon, sfancsali, sritter}@carnegielearning.com

ABSTRACT

Mastery learning in intelligent tutoring systems produces a differential attrition of students over time, based on their levels of knowledge and ability. This results in a systematic bias when student data are aggregated to produce learning curves. We outline a formal framework, based on Bayesian Knowledge Tracing, to evaluate the impact of differential student attrition in mastery learning systems, and use simulations to investigate the impact of this effect in both homogeneous and mixed populations of learners.

Keywords

Mastery learning, attrition bias, learning curves, aggregate learning, heterogeneous learner populations, knowledge tracing

1. MASTERY ATTRITION BIAS

Attrition bias occurs when some aspect of an experimental design has a significant and systematic effect on whether subjects complete all measures [6]. Although students working with an intelligent tutoring system (ITS) are not *ipso facto* in any experimental conditions, the mastery learning assessment built into many such systems creates an attrition bias. ITSs that implement mastery learning assess a student's performance as she works through instructional material, and continually re-evaluate whether she has received sufficient practice on targeted skills or knowledge components (KCs). This is a commonly used method to allocate student time, but by selectively removing students who master material quickly from the sample, it differentially biases the resulting data in ways that may conceal the learning of individuals [4]. ITSs that re-visit previously mastered KCs may exhibit this same effect only within blocks of contiguous practice.

In the Bayesian Knowledge Tracing (BKT)[2] model of student learning, the performance of an individual student can be described by the equation:

$$P_c(t) = P_k(t)(1-\theta_s) + (1-P_k(t))\theta_g$$

where $P_c(t)$ is the probability that the student will give a correct response at time *t*, given the probability of student knowledge, $P_k(t)$, and the performance parameters θ_s, θ_g (slip and guess). Consider a homogenous population of learners, all with the same parameters. We describe the average correctness of responses as:

$$\bar{C}(t) = \frac{K(t) - S(t) + G(t)}{K(t) + U(t)}$$

where $K(t)$ and $U(t)$ are the numbers of students in the known and unknown states at time *t*, respectively. $S(t)$ and $G(t)$ are binomial random variables giving the numbers of slips and guesses:

$$S(t) \sim B(K(t), \theta_s), \qquad G(t) \sim B(U(t), \theta_s)$$

It can be shown that the expected behavior of the aggregate learning curve:

$$E[\bar{C}(t)] = E[\bar{K}(t)](1-\theta_s) + (1 - E[\bar{K}(t)])\theta_g$$

is controlled by the ratio of students in the known state:

$$\bar{K}(t) = \frac{K(t)}{K(t) + U(t)}$$

The aggregate learning curve may be described as a weighted average between the expected performance in the known and unknown states, weighted by the ratio of students in each. The known and unknown populations will change according to the following stochastic recurrence relations:

$$K(t) = K(t-1) + L(t) - M_k(t)$$
$$U(t) = U(t-1) - L(t) - M_u(t)$$

where $L(t)$ is the number of students who learn the skill at time *t*, and so transition from the unknown into the known state. It is also binomially distributed: $L(t) \sim B(U(t-1), \theta_l)$, where θ_l is the BKT learning (aka. transition) parameter. $M_k(t)$ and $M_u(t)$ give the numbers of students from the known and unknown states, respectively, that are judged to have mastered the material by the system, and so removed from the population. The initial share of students in the known and unknown states is controlled by θ_i, the initial knowledge parameter from BKT:

$$K(1) = I \sim B(N, \theta_i).$$

From this we can see that the learning curve begins from a theoretical initial value of:

$$E[\bar{C}(1)] = \theta_i(1-\theta_s) + (1-\theta_i)\theta_g$$

In the no-mastery attrition situation, where $M_k(t)$ and $M_u(t)$ are always 0, $\bar{K}(t)$ will tend towards 1. Therefore the learning curve will converge to a theoretical maximum:

$$\lim_{t\to\infty} E[\bar{C}(t)] = (1-\theta_s)$$

We see this behavior in the left-hand plot of Figure 1.

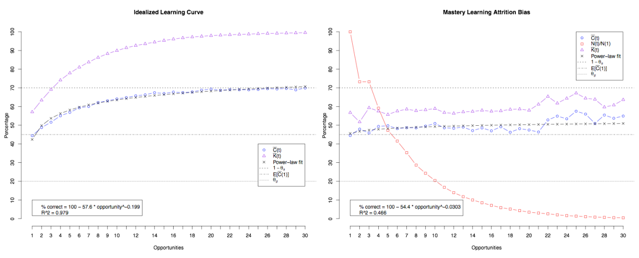


Figure 1: simulated learning curves with (right) and without (left) mastery learning

However, when mastery learning is involved, we must consider a balance of factors. We can expand the $\overline{K}(t)$ ratio in terms of its recurrence relations:

$$\overline{K}(t) = \frac{K(t-1) + L(t) - M_k(t)}{K(t-1) + U(t-1) - M_k(t) - M_u(t)}$$

Assuming $M_u(t)$ is negligible, the changes in this ratio depend on the relative magnitudes of $L(t)$ and $M_k(t)$. Except for when the unknown population has diminished to zero, the denominator of the ratio is larger than the numerator, so subtracting $M_k(t)$ from both will lead to a reduction in $\overline{K}(t)$. Since a falling $\overline{K}(t)$ ratio increases the weight that the unknown states play in the aggregate curve, mastery learning leads to lower aggregate performance, *ceteris paribus*.

Although learning and mastery have opposite effects on the instantaneous change in the $\overline{K}(t)$ ratio, they are not constant or independent over time. Learning has a negative-feedback relationship to itself: it reduces the size of the unknown student population, so the expected value of $L(t)$ will diminish over time. Mastery also has a negative-feedback relationship with the known population, but learning tends to counter-act that effect. Thus, learning has a positive-reinforcement relationship on mastery. In sum, there are many reasons why mastery learning leads to aggregate learning curves that do not take the shape we expect in their idealized form.

2. HETEROGENEOUS POPULATIONS

So far we have been considering idealized situations in which all students are instances of a BKT model with a common set of parameters. Naturally, we wish to investigate what can happen to aggregate learning curves when we have a heterogeneous population of different learners. There are very many different possible ways a heterogeneous population might be composed, and there could be very many perverse aggregate learning curves created by specially constructed mixed populations. We illustrate just a couple of examples that show interesting aggregate behavior.

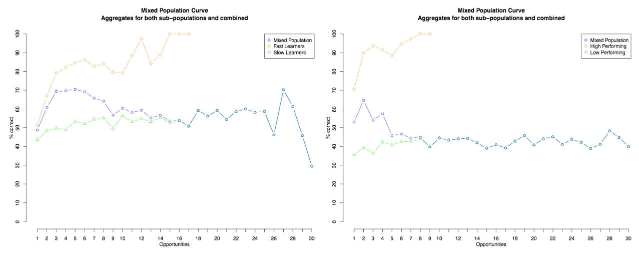


Figure 2: simulated heterogeneous populations.

Figure 2 demonstrates a couple of examples representing the range of aggregate behavior possible in mixed populations. In the left-hand plot, we show a population with similar initial knowledge, but composed of both fast-learning and slow-learning students. In the right-hand plot, we have a mixed population of higher-performing and lower-performing students. In both cases, the initial opportunities are a balanced mix of both populations. However, as the better students are preferentially removed by the mastery-learning system, they represent a diminishing fraction of the total population, and eventually the aggregate curve converges to that of the lower-performing sub-population. In the one case, the aggregate curve demonstrates a rising and falling pattern, whereas in the other case, the curve appears to demonstrate "negative learning". In a mixed population, the frequency of correct responses is the weighted average across the (j) sub-populations:

$$\bar{C}(t) = \frac{1}{N(t)} \sum_{j=1}^{J} N_{[j]}(t) \left[K_{[j]}(t) - S_{[j]}(t) + G_{[j]}(t) \right]$$

We could easily extend this notation of sub-populations to distinct per-student learning profiles. In this situation, $J = N(t)$ and $N_{[j]}(t)$ is either 1 or 0, depending on whether the j^{th} student has "mastered-out" or not.

3. CONCLUSIONS

Aggregate learning curves are used to evaluate and improve instructional systems[3]. However, there are significant distortions to aggregate measures of student learning created by the differential attrition bias inherent to mastery learning systems. Aggregate performance on each step shown by learning curves need not be representative of the learning of individuals or groups of students [4]. Aggregate measures of such attrition-biased data will tend to under-represent the amount of learning occurring. Explicitly modeling the effect of this attrition bias may be a fruitful direction for future research.

A mixed population with different learning characteristics can introduce additional distortions when mastery learning is involved. There has been much work already on identifying the learning characteristics of individuals and sub-populations[1][5]. Further developments in this direction would help build richer and more accurate models of learning robust to the attrition bias in mixed-population data.

4. REFERENCES

[1] Baker, R. Corbett, A. T., Aleven, V. 2008. More Accurate Student Modelling through Contextual Estimation of Slip and Guess Probabilities in Bayesian Knowledge Tracing. *Proceedings of the 9th International Conference on Intelligent Tutoring Systems.* ITS 2008. 406-415

[2] Corbett, A. T. and Anderson, J. R. 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction.* 4, 4, 253-278

[3] Martin, B., Mitrovic, A., Koedinger, K. R., Mathan, S., 2011. Evaluating and improving adaptive educational systems with learning curves. *User Modeling and User-Adapted Interaction*. 21, 3, 249-283

[4] Murray, C., Ritter, S., Nixon, T., Schwiebert, R., Hausmann, R., Towle, B., Fancsali, S., Vuong, A. 2013. Revealing the Learning in Learning Curves. *Proceedings of the 16th International Conference on Artificial Intelligence in Education* (Memphis, TN, July 9-13 2013). AIED 2013.

[5] Pardos, Z. A., Heffernan, N. T. 2010. Modeling individualization in a Bayesian networks implementation of knowledge tracing. *Proceedings of the 18th International Conference on User Modeling, Adaptation and Personalization.* 255-266

[6] Shadish, W., Cook, T., Campbell, D. 2002. *Experimental and Quasi-Experimental Designs for Generalized Causal Inference.* Houghton Mifflin Company, Boston, MA.

Extracting Time-evolving Latent Skills from Examination Time Series

Shinichi OEDA
Department of Information and Computer Engineering,
Kisarazu National College of Technology
2–11–1 Kiyomidai, Kisarazu-shi, Chiba, 292–0041, Japan
oeda@j.kisarazu.ac.jp

Kenji YAMANISHI
Department of Mathematical Informatics,
The University of Tokyo
7–3–1 Hongo, Bunkyo-ku, Tokyo, 113–8656, Japan
yamanishi@mist.i.u-tokyo.ac.jp

ABSTRACT

Examination results are used to judge whether an examinee possesses the desired latent skills. In order to grasp the skills, it is important to find which skills a question item contains. The relationship between items and skills may be represented by what we call a Q-matrix. Recent studies have been attempting to extract a Q-matrix with non-negative matrix factorization (NMF) from a set of examinees' test scores. However, they did not consider the time-evloving nature of latent skills. In order to comprehend the learning effects in the educational process, it is significant to study how the distribution of examinees' latent skills changes over time. In this paper, we propose novel methods for extracting both a Q-matrix and time-evolving latent skills from examination time series, simultaneously.

1. INTRODUCTION

The relationship between items and skills may be represented by what we call a *Q-matrix* [5]. Its original idea came from the rule space method (RSM) developed by Tatsuoka et al. [5]. The Q-matrix allows us to determine which skills are necessary to solve each item. However, the process of determining the skills involved in a given item is a boring and heavy task. Recently, there exist several studies on how to extract a Q-matrix from a set of examinees' test scores [1, 2]. These studies applied the *non-negative matrix factorization* (NMF) to the problem of establishing the skills from examinee performance data. They were applied to only static examination results provided at a certain time. However, in order to comprehend the learning effects in the educational process, it is significantly important to study how the distribution of examinees' latent skills changes over time. There have been studied on cognitive modeling from student performance over time [3, 4]. From another aspect, we propose novel methods for extracting time-evolving latent skills. It enables us to extract both a Q-matrix and time-evolving latent skills from examination time series, simultaneously.

2. EXTRACTION OF Q-MATRIX WITH NMF

In the Q-matrix extraction with NMF [1, 2], $\mathbf{R}$ represents an observed examination outcome data for m question items and n examinees. We define $\mathbf{Q}$ as a Q-matrix with m items and k skills, while we define $\mathbf{S}$ as an S-matrix with k skills and the n examinees. We assume that $\mathbf{R}$ is factorized into two matrices $\mathbf{Q}$ and $\mathbf{S}$. If Q-matrix is a conjunctive model, an examination result may be obtained according to the equation: $\neg\mathbf{R} = \mathbf{Q} \circ (\neg\mathbf{S})$ [1]. The operator $\neg$ denotes a Boolean negation, which is defined as a function that maps a value of 0 to 1 and any other value to 0. This equation will yield 0 in $\mathbf{R}$ whenever an examinee is missing one or more skills for a given item, and will yield 1 whenever all the necessary skills are mastered by an examinee.

3. Q-MATRIX EXTRACTION FROM EXAMINATION TIME SERIES

We propose an *online NMF* to extract a stable Q-matrix from examination time series in an online fashion. The key idea of the online NMF is that the initial values of matrix inherits those of the decomposed matrix at the previous data. The online NMF runs sequentially every time an examination result is input. When applying the conventional NMF-based method into such a sequential scenario, it must calculate a Q-matrix of which the initial values are set to random ones ignoring the lastest ones. Meanwhile, the online NMF produces a Q-matix letting the initial values be those obtained at the last time. This enables us to learn a Q-matrix in an online fashion.

In order to make the obtained Q-matrix more stable, we further propose an *online NMF with regularization.* In it we add the regularization term to the squared error function in the objective function to be minimized so that the Q-matrix does not change so much. We introduce here a cost function as follows:

$$\min_{\mathbf{Q}_t,\mathbf{S}_t} \{\|\neg\mathbf{R}_t - \mathbf{Q}_t\neg\mathbf{S}_t\|_F^2 + \lambda(t)(\|\mathbf{Q}_{t-1} - \mathbf{Q}_t\|_F^2)\}, \tag{1}$$

where $\lambda(t)$ is a monotonous increasing function of time. It is defined as $\lambda(t) = \alpha t/T$, where t is a time in $(1, \ldots, T)$, and α is a constant parameter of the increasing rate. At each time t, we find $\mathbf{Q}_t$ and $\neg\mathbf{S}_t$ according to (1), so that the sum of the factorization error and regularization term is minimized. We can do this through an iterative procedure in which each iteration involves two successive steps corresponding to suc-

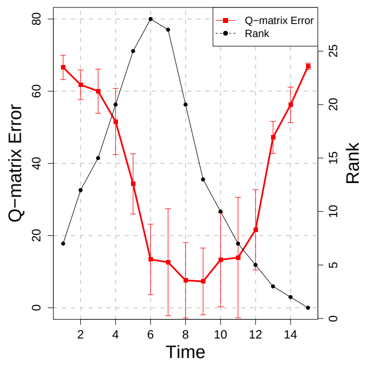


Figure 1: The conventional NMF

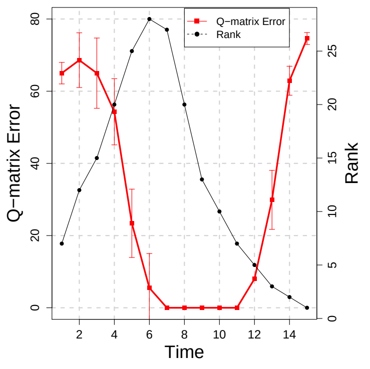


Figure 2: The online NMF

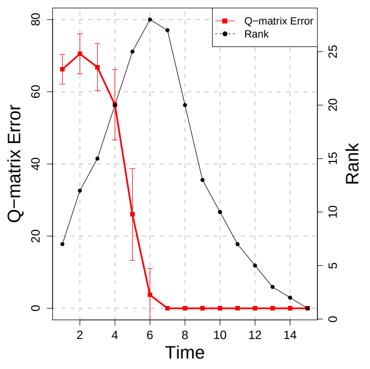


Figure 3: The online NMF with Regularization

cessive optimizations with respect to $\mathbf{Q}_t$ and $\neg\mathbf{S}_t$. First, we set $\mathbf{Q}_t$ by inheriting $\mathbf{Q}_{t-1}$, and choose $\neg\mathbf{S}_t$ by giving random non-negative values. Then, in the first phase, we minimize the cost function (1) with respect to $\neg\mathbf{S}_t$, keeping $\mathbf{Q}_t$ fixed. In the second phase, we minimize the same cost function with respect to $\mathbf{Q}_t$, keeping $\neg\mathbf{S}_t$ fixed. This two-stage optimization is then repeated until convergence.

4. EXPERIMENTAL RESULTS

In order to verify the effectiveness of our methods, we made a synthetic examination time series. We generated a time-varying S-matrix and a fixed Q-matrix to obtain $\neg\mathbf{R}_t$ according to the equation $\neg\mathbf{R}_t = \mathbf{Q} \circ (\neg\mathbf{S}_t)$. A conjunctive Q-matrix consisted of 31 items and 6 skills. We designed a time series of $\neg\mathbf{S}_t$ as a process of acquiring skills, on the basis of the item response theory (IRT) [6].

As a measure of the performance for Q-matrix extraction, we introduce a *Q-matrix error* e_t between $\mathbf{Q}$ and an extracted matrix $\widehat{\mathbf{Q}}_t$ as follows:

$$e_t = \|\widehat{\mathbf{Q}}_t - \mathbf{Q}\|_F^2. \qquad (2)$$

Figures 1 and 2 show the experimental results obtained using the coventional NMF and the online NMF. Note that the factorized solutions obtained using the NMF may not be unique due to the randomness of the initial matrices. Hence we calculated the mean of Q-matrix errors from 10-fold simulations and indicated error bars indicating the standard deviations. The Q-matrix errors both in Figures 1 and 2 were large at the initial and the final stages, while the ranks of matrices were small at the same stages. We calculated the rank of each matrix $\neg\mathbf{R}_t$ by means of QR decomposition. As a result, there was a correlation between the Q-matrix error and the rank of matrix. Note that in the conventional NMF, the Q-matrix error did not become zero at any stage, and the standard deviations were uniformly large. In the online NMF, the Q-matrix errors became zero at the middle stage of $t = 7, \cdots, 11$. However, the Q-matrix error gradually increased after $t = 12$. Figure 3 shows the result of the online NMF with regularization. It overcame the problem as above. That is, the Q-matrix error became zero after $t = 12$.

5. CONCLUSIONS

In this paper, we have introduced the online NMF with regularization for the purpose of extracting a Q-matrix and a time-evolving S-matrix from time series of examination results. We have designed it in order to extract a stable Q-matrix in an online fashion. We have employed a synthetic data set to demonstrate that it performs more accurately and in a more stable way than the conventional NMF in the extraction of the Q-matrix.

6. ACKNOWLEDGMENTS

We would like to thank Ryota TOMIOKA and Hisashi KASHIMA for their valuable comments and discussions. This research has been conducted at the University of Tokyo. This work was partially supported by MEXT KAKENHI 23240019, Aihara Project, the FIRST program from JSPS, initiated by CSTP, and NTT Corporation.

7. REFERENCES

[1] B. Beheshti, M. C. Desmarais, and R. Naceur. Methods to find the number of latent skills. *Proceedings 5th International Conference on of Educational Data Mining, EDM2012*, pages 81–86, 2012.

[2] M. C. Desmarais, B. Beheshti, and R. Naceur. Item to skills mapping: Deriving a conjunctive q-matrix from data. *Proceedings of the 11th International Conference Intelligent Tutoring Systems, ITS2012*, pages 454–463, 2012.

[3] J. P. Gonzalez-Brenes and J. Mostow. Dynamic cognitive tracing: Towards unified discovery of student and cognitive models. *Proceedings 5th International Conference on of Educational Data Mining, EDM2012*, pages 49–56, 2012.

[4] C. Studer, B. Junker, and H. Chan. Incorporating learning into the cognitive assessment framework. In *SREE Fall 2012 Conference on Effective Partnerships: Linking Practice and Research*, 2012.

[5] K. Tatsuoka. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement*, 20:345–354, 1983.

[6] W. J. van der Linden and R. K. Hambleton. Handbook of modern item response theory. *Springer*, 1996.

Uncovering Class-Wide Patterns in Responses to True/False Questions

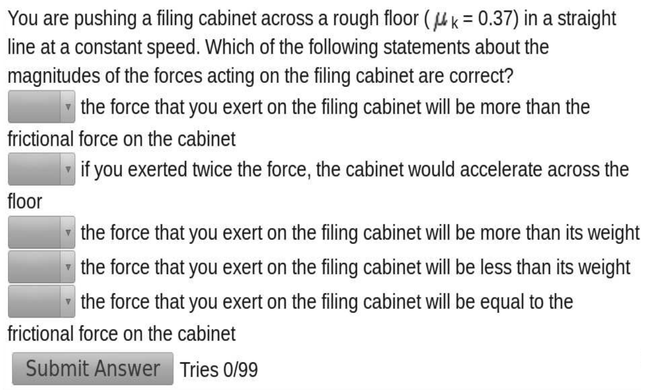
Andrew Pawl
Department of Engineering Physics
University of Wisconsin-Platteville
Platteville, WI 53818
pawla@uwplatt.edu

ABSTRACT

A popular type of problem in online homework involves a set of several true/false statements and requires that students submit their answers to all the statements at once. Such problems can force a student to submit many responses to the same true/false statement. It is possible to examine student submission patterns to problems of this type with the goal of determining which of the individual true/false statements exhibit a large proportion of response switches and which statements exhibit largely consistent responses. This paper describes algorithms that allow an instructor to uncover those statements that exhibit class-wide randomness and also those that exhibit a class-wide preference for an incorrect response. The utility of the approach is suggested by the fact that examining statements which emerge as outliers according to these metrics uncovers several statements that probe known student misconceptions.

1. INTRODUCTION

A popular type of problem in the LON-CAPA online homework network [1] consists of a situation or set of situations followed by five related true/false statements [2] (an example is shown in Fig. 1). The student is required to submit answers to all the true/false statements at once and receives only correct/incorrect feedback. A student who submits an incorrect answer will not know which of the statements has been answered incorrectly or even how many of the statements are incorrect, and so may submit as many as $2^5 = 32$ responses before arriving at the correct answer.


You are pushing a filing cabinet across a rough floor (μ_k = 0.37) in a straight line at a constant speed. Which of the following statements about the magnitudes of the forces acting on the filing cabinet are correct?

the force that you exert on the filing cabinet will be more than the frictional force on the cabinet

if you exerted twice the force, the cabinet would accelerate across the floor

the force that you exert on the filing cabinet will be more than its weight

the force that you exert on the filing cabinet will be less than its weight

the force that you exert on the filing cabinet will be equal to the frictional force on the cabinet

Figure 1: An example of the type of problem discussed in this paper.

One goal of this work is to develop a means to detect statements to which the class consistently responds incorrectly. Such response patterns correlate with *strong misconceptions*. The definition of "strong misconception" in the context of this work is an intuitive belief that is in conflict with the concepts taught in the course. An example is the first statement in the problem shown in Fig. 1. Research has shown that students in introductory physics courses have a strong tendency to believe that there must be a net force in the direction of motion, even when this belief conflicts with Newton's First Law [3, 4]. Thus, one might expect the class to consistently answer "True" to this statement, even when forced to answer multiple times.

Another complementary goal is to investigate whether significant class-wide randomness in the answers to a given statement can be an indicator of *incomplete understanding*. Again, the problem of Fig. 1 provides a useful illustration. If a significant portion of the class is indeed convinced that a net force is necessary to produce constant velocity, this could produce a conflict in the minds of the students about the consequences of applying more force. Will the extra force produce a steady acceleration in accordance with Newton's Second Law, or will there be a transient acceleration dropping to zero when the appropriate velocity is reached? Because of these conflicting ideas, one might expect students to exhibit a tendency to change their answer to the second statement shown in Fig. 1.

2. ASSESSING CONSISTENCY

A class with an average near 100% correct submissions to a certain statement is consistently giving the correct response. However, because a true/false statement has only one incorrect response, an average near 0% correct submissions *also* implies consistent responses (the class is continuing to respond with the one incorrect answer). If the class is answering randomly, the average will approach 50% correct submissions as the number of tries becomes large. Thus, a submission-weighted consistency score C_{sw} can be defined:

$$C_{sw} = \frac{N_{s,correct}}{N_{s,tot}} - 0.5 \tag{1}$$

where $N_{s,correct}$ is the number of correct submissions to the statement and $N_{s,tot}$ is the total number of submissions to the statement. With this definition, $C_{sw} = +0.5$ is complete correctness and $C_{sw} = -0.5$ is complete incorrectness.

A respondent-weighted measure of the consistency of a class on a particular statement C_{rw} can be defined by analyzing the first submission of each respondent. As an equation:

$$C_{rw} = \frac{N_{s,i,correct} - N_{s,i,incorrect}}{N_{r,tot}} \tag{2}$$

where $N_{s,i,correct}$ is the number of correct initial submissions, $N_{s,i,incorrect}$ is the number of incorrect initial submissions and $N_{r,tot}$ is the total number respondents.

The overall consistency score C_{tot} is then defined:

$$C_{tot} = C_{sw} \times C_{rw} \times (\text{sign}(C_{sw}) + \text{sign}(C_{rw})). \tag{3}$$

3. ASSESSING RANDOMNESS

The second goal is to uncover statements that produce frequent switching of the response. The total number of possible switches for a class making $N_{s,tot}$ submissions to a statement is $N_{s,tot} - N_{r,tot}$ where $N_{r,tot}$ is the number of respondents. A switch is "realized" if the current submission is different from the prior one. A submission-weighted randomness score R_{sw} can be defined as the fraction of possible switches that are realized:

$$R_{sw} = \frac{N_{s,switch}}{N_{s,tot} - N_{r,tot}} \tag{4}$$

where $N_{s,switch}$ is the number of submissions that represent a switch of the answer from the immediate predecessor.

A respondent-weighted measure of randomness R_{rw} can be defined by determining what fraction of the students who responded to the statement ever switched their response from correct to incorrect. As an equation:

$$R_{rw} = \frac{N_{r,correct \rightarrow incorrect}}{N_{r,tot}} \tag{5}$$

where $N_{r,correct \rightarrow incorrect}$ is the number of respondents who ever switched their response from correct to incorrect.

The overall randomness score is defined:

$$R_{tot} = R_{sw} \times R_{rw}. \tag{6}$$

4. EVIDENCE FOR VALIDITY

Fig. 2 is a scatter plot of the overall consistency score versus the overall randomness score for 250 true/false statements from problems of the type described. The plot allows a re-examination of the example of Fig. 1. The expectation outlined in the Introduction is that the first statement shown in the problem of Fig. 1 should qualify as a strong misconception and that the second statement should exhibit significant

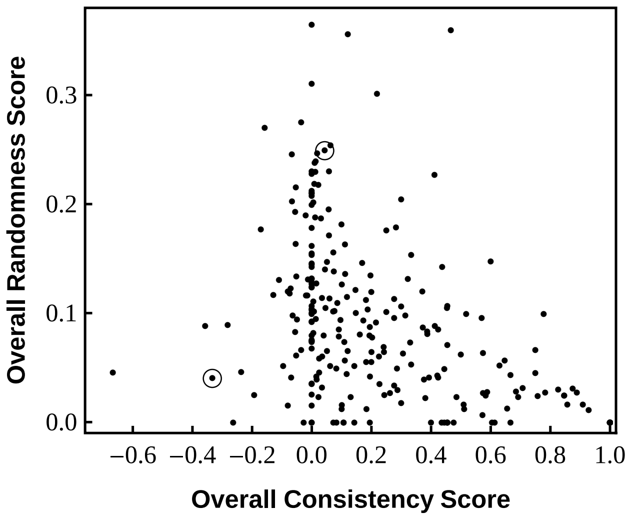


Figure 2: R_{tot} (defined in Eq. 6) vs. C_{tot} (defined in Eq. 3) for 250 true/false statements.

randomness. The two points outlined with circles in Fig. 2 correspond to these statements. As expected, the first statement has a strongly negative consistency score (-0.33) while the second statement has a strong randomness score (0.25).

5. CONCLUSIONS

This paper has presented data-mining algorithms for assessing the consistency and the randomness of student responses to individual true/false statements. These algorithms are directly applicable to problems involving several linked true/false statements, which have been implemented in online homework. Investigation of examples indicates that the consistency score can uncover class-wide misconceptions and the randomness score can be a useful indicator of incomplete understanding among the class. Both scores can also serve to uncover errors in problem construction. The promise of the approach is that a simple question format that is suitable for use in online homework or as part of online courses can uncover the specific concepts that give a significant portion of the class problems.

6. ACKNOWLEDGMENTS

This paper relies on work done when the author was a post-doc with D.E. Pritchard's RELATE group at the Massachusetts Institute of Technology. R.E. Teodorescu constructed the homework sets and provided comments on this work.

7. REFERENCES

[1] http://lon-capa.org

[2] Kashy, E., Gaff, S.J., Pawley, N.H., Stretch, W.L., Wolfe, S.L., Morrissey, D.J., and Tsai, Y. 1995. Conceptual questions in computer-assisted assignments. *Am. J. Phys.* 63 (Nov. 1995), 1000-1004.

[3] Clement, J. 1982. Students' preconceptions in introductory mechanics. *Am. J. Phys.* 50 (Jan. 1982), 66-71.

[4] Hestenes, D., Wells, M., and Swackhamer, G. 1992. Force Concept Inventory. *Phys. Teach.* 30 (Mar. 1992) 141-158.

Causal Modeling to Understand the Relationship between Student Attitudes, Affect and Outcomes

Dovan Rai
Department of Computer Science
Worcester Polytechnic Institute
dovan@wpi.edu

Joseph E. Beck
Department of Computer Science
Worcester Polytechnic Institute
josephbeck@wpi.edu

Ivon Arroyo
Department of Computer Science
University of Massachusetts Amherst
ivon@cs.umass.edu

ABSTRACT

We are using causal modeling to analyze relationships between pedagogical intervention, students' attitudes, affective states, perceptions and outcomes, based on the data from a math tutor, Wayang Outpost. The causal model generated gives interpretable multi level interrelationships within the data variables identifying direct and indirect effects among them. We observed that among the four affective variables, confidence and frustration are more tightly linked with their performance and ability whereas interest and excitement are more related to their attitude and appreciation of math and tutor.

1. INTRODUCTION

In recent years, researchers have found that attending to students' motivational and affective characteristics are as crucial as cognitive aspects for effective learning. Since students' cognitive, affective and behavioral aspects are interconnected, analyzing their interrelationships give us clearer understanding of the learning process.

Causal models [1, 2] are graphical models that make the additional assumption that the links between nodes represent causal influence. By causal, we mean that a link A→B means that if we intervene and change the value of A, then B will change. Based on the conditional independencies within the data, causal modeling makes causal inferences among the variables.

We used TETRAD, a free causal modeling software package [2] to generate causal graphs. We also made use of an extension to Tetrad, developed by Doug Selent, a graduate student at WPI, that enabled us to restrict links on the basis of the magnitude of the relation between the pair of nodes. It measures the magnitude of the relation using R^2, the amount of the variability accounted for by the relationship between the nodes.

The data comes from students working with Wayang Outpost, an adaptive math tutoring system that helps students learn to solve standardized-test questions, in particular state-based exams taken at the end of high school in the USA.

We are using data from 94 middle school students, in the 7th and 8th grade (approximately 12 to 14 years old), who were part of mathematics classes in a rural-area public middle school in Massachusetts, USA. As part of the activity, students took a survey on the first day, which assessed their baseline achievement level, as well as affective and motivational factors related to mathematics problem solving. The students took identical survey at the end of the study. Student responses were collected in 5-point Likert scale.

Based on the survey data, we created different variables, which we will be explaining in the following paragraphs.

Attitudes and appreciation of Math: Students were asked various questions related to their attitude towards Math before they used the tutor.

Math Self Concept (Sample item: *How good would you be at learning something new in math?)*

Math Liking (Sample item: *How much do you like doing math?)*

Math Value (Sample item: *In general, how useful is what you learn in math?)*

Affect: Students were asked questions on four affective variables while using the tutor: Confident, Frustrated, Interested, Excited (Sample Item: How confident do you feel when solving math problems?)

Pedagogical intervention: MathFluencyTraining (training on basic math facts, e.g. multiplication tables, and retrieval speed) Students were randomly assigned to either receive or not receive math fluency training.

Perception of tutoring system: PerceptionWayang (Students' perception of the tutor)

Perception_LC (Students' perception of learning companion, an animated learning peer who offers encouragement to the student)

Pretest Score and learningGain: Students took MCAS (state standardized test) test before using the tutoring system. We are using this test score as pretest score.

We calculate the difference in test scores between the MCAS tests students took before and after using the tutor and designate that value as learningGain.

Gain and outcomes

Based on student's pre and post survey responses, gain outcomes were calculated. We are using the base and gain values for the constructs and not considering post values (which are redundant once we took the first the based and gain values).

2. CAUSAL MODELING

We inputted the data in TETRAD and performed a model search using the PC search algorithm and generated a graph as shown in figure 1. PC search algorithm assumes that there are no hidden common causes between observed variables in the input (i.e., variables from the data set, or observed variables in the input graph) and that the graphical structure sought has no cycles [2]. The causal links are color-coded (corresponding to the R^2 value: red >0.5, orange >0.1, yellow >0.05) and 'plus' sign denotes positive relationship and 'minus' sign denotes negative relationships. We also input domain knowledge based on temporal precedence. For example: gender is put on higher knowledge tier than math self concept so that there could be a causal link directed from gender to math self-concept but not the other way round.

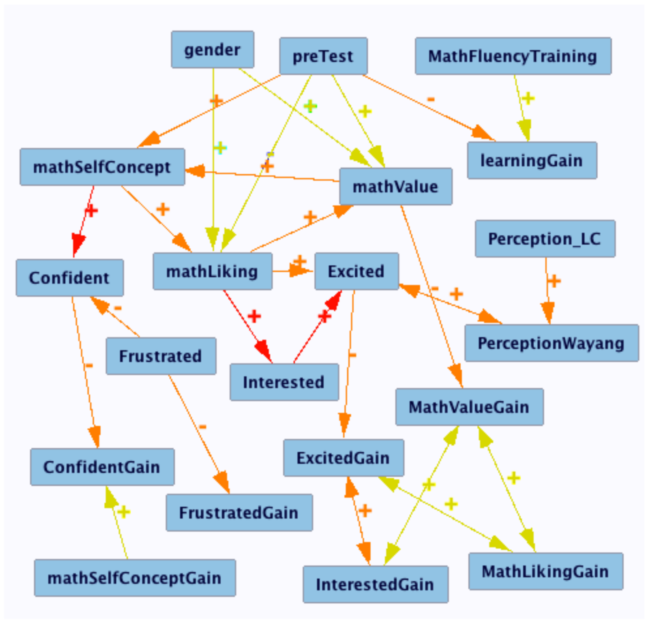


Figure 1 Causal model of attitude, affect and outcomes

We observe very strong relationships between student attitude towards math and her affective states within the tutor. Students who have higher self-concept in math report being more *Confident*. Students who like math report being more *Interested* and *Excited* while using the tutor. While *Interested, Excited and Confident* are more tightly coupled with attitude variables, *Frustrated* is relatively separate and connected to that web via the variable *Confident*.

In the correlation matrix, all three math attitude variables are related to *Interested, Excited,* and *Confident* and only *mathSelfConcept* and *MathLiking* are negatively correlated to *Frustrated* (*mathValue* has no relation to *Frustrated*). Three math attitude variables are also highly correlated among themselves. Causal model has teased apart this dense correlation web into sparser directed structure.

From causal model, we see that gender directly affects *mathLiking* and *mathValue*. Affect variables *Excited* and *Interested* are indirectly affected by gender mediating through *mathLiking*. The causal structure (gender→mathLiking→ Interested) asserts that female students like math more, which makes them more interested while using Wayang. There is no direct link gender→Interested which implies that the female students who like math as much as male students do not necessarily have any higher Interest level. But, this could be a case of multicollinearity since Correlation (mathLiking, Interest)=0.85** [4].

Math fluency training has a significant impact in improving learning gain of students as shown by the link Math Fluency (Training → learningGain.) This indicates that a group of students who got math fluency training achieved higher improvement in math tests after using the software. A paper has been published about this [3]; however, it is interesting to see the strength of this causality compared to other factors.

Pretest is only indirectly related to affect variables through attitude variables as mediators. This suggests that higher knowledge has to be internalized as higher math self-concept and math liking to be reflected as higher enjoyment of the tutor.

In terms of affective variables, we can see the following two clusters:

Performance oriented (incoming math ability) student descriptors:

preTest, math self-concept→confidence and frustration

Students who have higher prior knowledge and better self concept in math reported higher confidence and lower frustration.

Liking and Appreciation:

Math value, math liking, perception of LC, perception of Wayang → interest and excitement

Also, students who reported gain in confidence also had higher gain in self-concept in math and those who gained in interest and excitement also ended up with higher liking for math and had greater value for math. We also observe that gender belongs to this cluster (related to mathValue, mathLiking, interest and excitement). Basically, among the four affective variables, confidence and frustration are more tightly linked with their performance and ability whereas interest and excitement are more related to their attitude and appreciation of math and tutor.

3. CONCLUSIONS

Since correlation underdetermines causality, there are multiple equivalent causal models that can be generated from the same set of data. Moreover, the stronger causal assumptions employed by this approach add more analytical power but also introduce higher chances of inaccuracy. Researchers have to, therefore, be careful about interpreting the causal relations of the model before making any causal claims. Most of the times, causal models are only able to summarize associations rather than uncover new causal mechanisms. It basically depends on whether we are able to observe all possible common causes in our data set. We do not recommend causal modeling to use as a tool to make strong causal conclusions, as we would do with randomized controlled studies. A thorough understanding of the domain is required before we make causal interpretations of the model. Causal models can only be as good as the variables that we are able to include in the modeling process. But given the data variables, causal modeling is the most superior exploratory tool available compared to the existing statistical approaches such as correlation and regression.

4. REFERENCES

[1] Pearl J. 2009. *Causality*. 2nd Edition. Cambridge University Press.

[2] Glymour, C. and Scheines, R. 2004. *Causal modeling with the TETRAD program*. Synthese. 37-64

[3] Arroyo, I., Royer, J. M., and Woolf, B.P. 2011. *Using an Intelligent Tutor and Math Fluency Training to Improve Math Performance*. IJAIED, Best of ITS 2010. Vol 21 (2), pp. 135-152. IOS Press.

[4] Rai, D. and Beck J. E. 2011. *Exploring user data from a game-like math tutor: a case study in causal modeling*. In Proceedings of the 4th International Conference on Educational Data Mining (EDM). 307-313

Determining Review Coverage by Extracting Topic Sentences Using A Graph-based Clustering Approach

Lakshmi Ramachandran
North Carolina State University
lramach@ncsu.edu

Balaraman Ravindran
Indian Institute of Technology, Madras
ravi@cse.iitm.ac.in

Edward F. Gehringer
North Carolina State University
efg@ncsu.edu

ABSTRACT
Reviews of technical articles or documents must be thorough in discussing their content. At times a review may be based on just one section in a document, say the *Introduction*. Review coverage is the extent to which a review covers the "important topics" in a document. In this paper we present an approach to evaluate the coverage of a submission by a review. We use a novel agglomerative clustering technique to group the submission's sentences into topic clusters. We identify topic sentences from these clusters, and calculate review coverage in terms of the overlaps between the review and the submission's topic sentences. We evaluate our coverage identification approach on peer-review data from Expertiza, a collaborative, web-based learning application. Our approach produces a high correlation of 0.51 with human-provided coverage values.

Keywords
review quality, review coverage, topic identification, agglomerative clustering, lexico-semantic matching

1. INTRODUCTION
The past few years have witnessed a growth in Massive Open Online Courses (MOOCs) such as Coursera and Udacity, as a platform for web-based collaborative learning. MOOCs require a scalable means of assessment, and for material that cannot be assessed by multiple-choice tests, peer-review fills the bill. Feedback in the form of text-based reviews help authors identify mistakes in their work, and learn possible ways of improving them. Since reviews play a crucial role in helping authors, it is important to ensure that they are *complete*, and their content is *useful* to authors. At times reviews may cover just one section in the author's submission (the text under review), say the *Introduction*, and provide no feedback on any of the other sections in the document.

Kuhne et al. [1] found that authors are content with reviewers who have made an effort to read and understand the author's work. Reviews that cover the important sections of the author's work are likely to be more useful, since they are more complete than reviews discussing a single section. A complete review also reflects positively on a reviewer's understanding of the author's work.

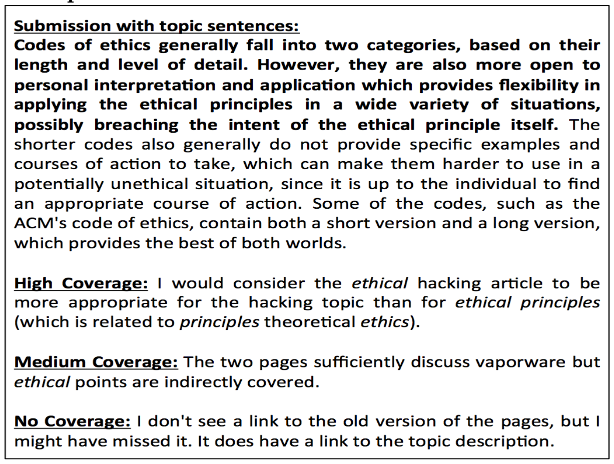
Existing review assessment approaches use shallow text features such as word count to analyze their usefulness. Xiong et al. use a bag-of-words exact match approach to identify instances of problems (in the author's work) caught by peer-reviews [2]. Cho uses machine classification techniques such as naïve Bayes, SVM (support vector machines) and decision trees to classify feedback [3]. At present none of the automatic review analysis approaches look for the degree of coverage of a submission by a review. One of the chief contributions of this paper is the focus on the important but often ignored problem of identifying review coverage.

2. APPROACH
We employ an agglomerative clustering technique to group submission sentences into clusters or topics, and then identify the most representative sentences from across the different clusters. Cluster-based approaches have been widely applied to text and other knowledge mining applications. Steinbach et al. use bisecting k-means to cluster documents [7]. Qazvinian et al. [5] use a cluster-based approach to determine the sentences central to a document. They use a hierarchical agglomeration algorithm to cluster sentences. The ClusterRank algorithm, proposed by Garg et al. [6], applies a clustering technique to identify sentences belonging to the same topic.

Sentences discussing the same topic, but containing different terms may not be effectively grouped by a clustering approach relying purely on the frequency of words. Steinbach et al. found that agglomerative clustering with a word-frequency based matching made mistakes by grouping nearest documents belonging to different classes into the same cluster [7].

We employ a lexico-semantic matching technique, which captures context information. We use a word order graph to represent text, since it captures syntax or order of tokens in a text. Word order graphs are suited for identifying lexical and voice changes, which are common among paraphrased text. Similarity should capture the degree of relatedness between texts. Hence we use a WordNet-based metric [8]. Topic-representative sentences are selected from the most significant clusters. A review is compared with topic sentences to identify coverage.

Figure 1: Submission with its topic sentences, and three reviews with high, medium and no coverage of the topic sentences



Submission with topic sentences:
Codes of ethics generally fall into two categories, based on their length and level of detail. However, they are also more open to personal interpretation and application which provides flexibility in applying the ethical principles in a wide variety of situations, possibly breaching the intent of the ethical principle itself. The shorter codes also generally do not provide specific examples and courses of action to take, which can make them harder to use in a potentially unethical situation, since it is up to the individual to find an appropriate course of action. Some of the codes, such as the ACM's code of ethics, contain both a short version and a long version, which provides the best of both worlds.

High Coverage: I would consider the *ethical* hacking article to be more appropriate for the hacking topic than for *ethical principles* (which is related to *principles* theoretical *ethics*).

Medium Coverage: The two pages sufficiently discuss vaporware but *ethical* points are indirectly covered.

No Coverage: I don't see a link to the old version of the pages, but I might have missed it. It does have a link to the topic description.

Table 1: Correlation between system-generated and human-provided coverage values.

Approach	correlation	Avg. # words
Our system	**0.51**	108
MEAD summarizer	0.46	100

In order to illustrate our approach we use real-world submission and review data from assignments completed using Expertiza [9]. Expertiza is a collaborative, web-based learning application that helps students submit assignments and peer review each other's work. Figure 1 contains a sample submission with its topic-representative sentences highlighted in bold, and three sample reviews with high, medium and no coverage of the submission's topic sentences. The first review *covers* the submission because it mentions *ethical principles* and *ethics*. However, the review with *medium* coverage mentions just *ethics*, and the review with *no* coverage does not contain any relevant information.

3. EXPERIMENT

In this section we study the usefulness of our approach in determining a review's coverage. We compare our approach with MEAD, a centroid-based summarization approach [4]. Radev et al.'s approach uses the most common words in a document to identify the best sentences to be included in a summary. MEAD is an extractive summarization approach, and since in our approach too we extract the most representative sentences from a submission, we find MEAD to be an ideal system to compare our approach with.

We use peer-review data from computer science classes over a couple of semesters to evaluate our approach. A dataset containing 577 reviews and submissions was selected from Expertiza [9]. Review data is annotated on a scale of 0-5, where 0 indicates no coverage and 5 indicates maximum coverage.

We identify the Spearman correlation between system-generated and human-provided coverage values (Table 1). Our approach produces a correlation of 0.51, while MEAD's coverage produces a correlation of 0.46. A positive correlation of 0.51 indicates that the system has a good degree of agreement with human-provided coverage values.

Topic sentences generated by our approach have almost the same number of words as those generated by MEAD. However, our approach produces higher correlations with human ratings than the output from the MEAD summarizer. Thus, our approach is able to effectively identify topic-representative sentences from a document, and estimate a review's coverage of these topic sentences.

4. CONCLUSION

Assessment of reviews is an important problem in the fields of education, science and human resources, and so it is worthy of serious attention. Our aim is to utilize natural language processing techniques to determine review coverage. Since reviews are central to the process of assessment it is important to ensure that they cover the main points of a submission. In this paper we have explained our approach to solving the problem of automatically determining a review's coverage of a submission. Ours is a pioneering effort in applying clustering and topic identification techniques to calculate review coverage. We have also shown that our review coverage approach produces a good correlation of 0.51 with human-provided coverage values.

5. REFERENCES

[1] C. Kuhne, K. Bohm, and J. Z. Yue, "Reviewing the reviewers: A study of author perception on peer reviews in computer science," in *Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2010 6th International Conference on.* IEEE, 2010, pp. 1–8.

[2] W. Xiong, D. J. Litman, and C. D. Schunn, "Assessing reviewer's performance based on mining problem localization in peer-review data." in *EDM*, 2010, pp. 211–220.

[3] K. Cho, "Machine classification of peer comments in physics," in *EDM*, 2008, pp. 192–196.

[4] D. R. Radev, H. Jing, M. Styś, and D. Tam, "Centroid-based summarization of multiple documents," *Inf. Process. Manage.*, vol. 40, no. 6, pp. 919–938, Nov. 2004.

[5] V. Qazvinian and D. R. Radev, "Scientific paper summarization using citation summary networks," in *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, ser. COLING, 2008, pp. 689–696.

[6] N. Garg, B. Favre, K. Riedhammer, and D. Hakkani-Tür, "Clusterrank: a graph based method for meeting summarization," in *INTERSPEECH*, 2009, pp. 1499–1502.

[7] M. Steinbach, G. Karypis, V. Kumar *et al.*, "A comparison of document clustering techniques," in *KDD workshop on text mining*, vol. 400, 2000, pp. 525–526.

[8] C. Fellbaum, "Wordnet: An electronic lexical database." *MIT Press*, p. 423, 1998.

[9] E. F. Gehringer, "Expertiza: Managing feedback in collaborative learning." in *Monitoring and Assessment in Online Collaborative Environments: Emergent Computational Technologies for E-Learning Support*, 2010, pp. 75–96.

Affective State Detection in Educational Systems through Mining Multimodal Data Sources

Sergio Salmeron-Majadas
aDeNu Research Group. UNED
Calle Juan del Rosal, 16. Madrid
28040. Spain
+34 91 398 93 88
ssalmeron@bec.uned.es

Olga C. Santos
aDeNu Research Group. UNED
Calle Juan del Rosal, 16. Madrid
28040. Spain
+34 91 398 93 88
ocsantos@dia.uned.es

Jesus G. Boticario
aDeNu Research Group. UNED
Calle Juan del Rosal, 16. Madrid
28040. Spain
+34 91 398 71 97
jgb@dia.uned.es

ABSTRACT

Affective computing in e-learning is playing a vital role, as emotions can strongly impact on learner's results. Detecting affective states and managing them can lead to a learning performance improvement. For this, different sensors can be used to monitor user's interactions and detect emotional changes. Due to the huge varied data volume a multimodal approach based on data mining has been proposed.

Keywords

Affective Computing, Educational Data Mining, Emotions, Adaptive Systems, User Modeling.

1. RESEARCH APPROACH

Given the strong role emotions play on the learning process, by combining in a wise way the use of emotional information and user interactions in an e-learning platform, an impact on user's performance and cognition can be expected [3]. The ongoing research works in the literature aim to progress on managing different information, sources such as physiological sensors or face tracking systems [1]. To this, data mining is applied to provide personalized feedback to learners, which aims at supporting them in achieving better results on their tasks [5].

The approach followed in the MAMIPEC project [4] is focused on addressing the problem of emotion detection from a multimodal viewpoint by using different data sources. Our goal here is to combine those data sources to model the learner's current affective state and improve thus the possible results obtained from a single data source [1]. The learner model, which is based on standards, is thus enriched with new features that are used to provide personalized feedback during the learning process. In particular, the research of this ongoing Ph.D work focuses on identifying and modeling affective states to support adaptive features in educational systems. The top-level hypothesis behind the research is that the application of data mining techniques to different emotional data sources can improve the modeling of the learners' affective state in terms of standards and thus, better provide a personalized support in open educational service oriented architectures (i.e. those that take advantage of standards-based models).

2. INITIAL EXPERIMENTS FOR DATA GATHERING

A large-scale experiment was carried out to get data to feed the data mining system. More than 90 participants came to our lab and were asked to solve a series of mathematical tasks (the mathematical domain was chosen as Maths can awaken different intense emotions in the learner [2]) in a dotLRN platform while being monitored. Physiological, facial and interaction data were gathered during the experiment. Physiological signals recorded were: i) the participant's heart rate, ii) the participant's breath frequency, iii) the participant's galvanic skin response (electrical conductance of participant's skin) and temperature. For detecting changes in the signals recorded, baselines for the involved sensors were computed. Besides physiological sensor devices, a Microsoft Kinect device was used to extract participant's facial characteristic points in order to get their expressions. Additionally, a key-logger and a mouse-tracker were developed in order to track all the interactions performed by the user with the platform during de experiment. A webcam registered the face of the participant during all the session as well as the desktop was also recorded. Also, some questionnaires were offered to the participants, such as Big Five Inventory (BFI) to know the main five structural dimensions of individual's personality, General Self-Efficacy Scale (GSE) to get the self-beliefs of the participants to cope with a variety of difficult demands in life and the Positive and Negative Affect Schedule (PANAS).

To evaluate the emotions experienced by participants during the tasks, participants were asked to fulfill a SAM (Self-Assessment Manikin) scale after each exercise. Moreover, after each mathematical task, participants were also asked to write a self report (as plain text) regarding their feelings when doing the task.

3. DATA MINING

With this collected data, the work on this Ph.D focuses on detecting the emotions felt by the users by processing the different data sources obtained during the experience.

3.1 Data Preprocessing

For the keystroke data, some measures were processed such as mean time between strokes or between stroke and release. For the sensor data, first of all, the data were split into tasks and the noisy values were removed. Once done this, the mean of the physiological baseline recorded before the experiment was calculated so all the other values are compared to this mean. The mean of all these differences is stored for all the tasks solved by each participant. For detecting affective facial information the Kinect is expected to provide facial patterns or gestures and link them to emotional states. To do this a psycho-educational expert is currently viewing the webcam captured videos in order to detect relevant facial expressions during the experiment. The mouse-tracking data have not been processed either. Although some measures can be easily calculated (mean speed, distance moved, etc) from the interactions recorded, there is a need to research how to identify meaningful individual mouse movements (i.e. when the mouse draws differentiated paths).

3.2 Data Mining Methods

Currently, two different approaches are being explored: one for predicting user's emotion from the text typed (in the emotional report) and another where all the processed information was used as input to predict the SAM valence values given by learners.

3.2.1 Text Mining

Three of the tasks consisted in typing participants' emotions (like a typed think-aloud), these tasks were used to gather information from the keyboard interactions, but also to get emotional knowledge from the text. A simple approach was adopted, consisting of processing the text typed by the user searching for positive and negative terms included in the emotional valence labeled MPQA Opinion Corpus affective database to provide an emotional score to each text. A psycho-emotional expert also assigned an emotional valence score to each text after its reading to label the data for its use with supervised learning methods.

When comparing the text mining scores with the experts' score based on the participants' reports, if both are binned into 2 categories (Positive-Negative), a 73.42% of success prediction rate was achieved.

3.2.2 Affective States Detection

Another data mining approach aims to predict the user's emotional valence while dealing with a given task based on the processed records obtained during that task: questionnaires scores, text mining scores, keyboard interactions and physiological data.

To use supervised learning methods, two different labeling systems were used as results to be predicted: the SAM scores given by the participants and experts' emotional tagging based on users' emotional report afterwards.

For the data mining process, prediction trees and naïve bayes predictors were used. The biggest effort on this stage was made by grouping and binning different data sources in different ways to get the best results. Two, three and four-category binning was used when binning valence values and text mining scores (using equal percentile values so all the bins have the same probability and equal spaced intervals over the domain range). After every different combination, naïve bayes and prediction tree (C4.5) nodes were executed with leave-one-out sampling.

The best result were obtained using a naïve bayes algorithm with a 79,72% success rate on predicting the valence given by the experts ignoring the neutral tagged values.

4. OPEN ISSUES

The work reported so far has helped to identify several kinds of open issues to deal with i) the infrastructure for the data gathering and synchronization, ii) the emotions detection of neutral values, iii) the data mining approach itself, and iv) reducing the intrusiveness. Results achieved so far suggest that the way sensor data have been preprocessed might need to be redesigned, focusing not only on using data mining over all the data gathered preprocessed, but also on using data mining to preprocess single signals data (eg. Keystrokes, mouse movements). Finally, a study on which data sources are more valuable for emotions detection should be done, focusing on getting emotional information in a non-intrusive and cheap way.

5. ONGOING WORKS

Several open issues have been identified analyzing the data gathered. In particular, there is a need to redefine certain aspects from the data gathering in order to provide more meaningful information from where to obtain better mining results in future experiments. The approach proposed at the end of this stage must offer an affective state detection strong enough to provide a robust base to the model generated in the next layers.

Proposed work aims to provide an accurate standards-based user model useful to supply personalized assistance taking the learner's affective state into account. The first layer of this work is still being addressed, studying the state of the art in order to meet multimodal approaches on emotion detection and also being able to detect different emotions by using data mining.

6. ACKNOWLEDGEMENT

Authors would like to thank experiments' participants, MAMIPEC project (TIN2011-29221-C03-01) colleagues and the Spanish Government for its funding.

7. REFERENCES

[1] D'Mello, S. and Kory, J. 2012. Consistent but modest: a meta-analysis on unimodal and multimodal affect detection accuracies from 30 studies. Proceedings of the 14th ACM international conference on Multimodal interaction (New York, NY, USA, 2012), 31–38.

[2] Goetz, T., Frenzel, A.C., Hall, N.C. and Pekrun, R. 2008. Antecedents of academic emotions: Testing the internal/external frame of reference model for academic enjoyment. Contemporary Educational Psychology. 33,1 (2008), 9–33.

[3] Moore, S.C. and Oaksford, M. 2002. Some long-term effects of emotion on cognition. British Journal of Psychology. 93, 3 (2002), 383–395.

[4] Santos, O.C., Salmeron-Majadas, S., Boticario, J.G. 2013. Emotions detection from math exercises by combining several data sources. Proceedings of the 16th International Conference on Artificial Intelligence in Education (AIED 2013), in press

[5] Shen, L., Wang, M. and Shen, R. 2009. Affective e-learning: Using "emotional" data to improve learning in pervasive learning environment. Educational Technology & Society. 12, 2 (2009), 176–189.

Exploring the relationship between course structure and etext usage in blended and open online courses

Daniel T. Seaton
Office of Digital Learning
Mass. Institute of Technology
Cambridge, MA 02139
dseaton@mit.edu

Yoav Bergner
Physics and RLE
Mass. Institute of Technology
Cambridge, MA 02139
bergner@mit.edu

David E. Pritchard
Physics and RLE
Mass. Institute of Technology
Cambridge, MA 02139
dpritch@mit.edu

ABSTRACT

We use a two-parameter family of bounded distribution functions (Kumaraswamy) to fit electronic textbook (etext) usage in 20 blended and online courses from Michigan State University, MIT, and edX. We observe clusters of courses in the parameter space that correlate with course structural features such as frequency of exams.

Keywords

Course structure, etexts, MOOCs, usage mining

1. INTRODUCTION

When etexts are integrated into Learning Management Systems (LMS), one can extract the number, frequency and duration of page views from the tracking logs. These data have fed back into etext interface design [2] and personalized approaches aimed at understanding student reading habits and comprehension [3], but an incomplete picture remains in terms of guiding instructors on how to integrate etexts into courses. This is particularly salient given studies which point to low use of traditional textbooks and poor correlation of use with performance [4].

We consider the fraction of etext pages accessed by students in courses of varying structure. Aspects of course structure include the primary/supplementary role of the etext, its integration with graded assessment, and the frequency of exams. Our data come from blended and distance learning courses from Michigan State University (MSU) and from open online courses from the RELATE group at the Massachusetts Institute of Technology (MIT) and edX. The MSU populations are typical for introductory science at a large state university. In contrast, the student populations in open online courses are highly variable in age and preparation [1].

Courses in this study use either LON-CAPA, with e-texts as modularized html pages, or edX, which uses digital versions of traditional textbooks within simple navigation. We fit the distribution of unique page views in each course using a two-parameter family of distribution functions with support on the interval [0,1]. The complimentary cumulative distribution function (CCDF) of the Kumaraswamy distribution is given by $F(x; a, b) = (1 - x^a)^b$. The (a, b) parameters which determine the shape of each distribution may not be familiar. We highlight four relevant regions: bimodal $(a, b < 1)$, low usage $(a < 1, b > 1)$, high usage $(a > 1, b < 1)$, and unimodal $(a, b > 1)$. Note the probability distributions (PDF, not CCDF) associated with $a = b = 0.5$ (bimodal) and $a = b = 1.5$ (unimodal) have the same "average" use.

2. BLENDED COURSES

We first consider nine blended courses from MSU (same instructor), all Fall semester Mechanics using LON-CAPA for weekly homework and readings from the MultiMedia Physics[1] etext. We group the courses by structure differences as follows: MSU supplemental ($N = 898, 911, 808$), in which the etext was available alongside a physical textbook; MSU primary ($N = 159, 190$), in which the etext was the only learning text; and MSU reformed ($N = 211, 209, 197, 254$), which used a primary etext along with reading quizzes and a switch to frequent exams (every two weeks).

Fig. 1(a) shows both the CCDFs (inlay) and the clustering of a, b values in the fit-parameter space, where we also indicate contours of constant fractional usage. MSU supplemental courses are in the low-usage region, average view fraction (avf) < 0.1, while MSU primary courses are in the unimodal region (avf ≈ 0.55). The change of role from supplementary to primary assigned textbook appears to have a significant impact. Proximity to the $(a = b = 1)$ saddle point indicates that the MSU primary view distributions are almost flat. MSU reformed courses are in the high-usage region (avf ≈ 0.8). Unfortunately we cannot separate the effect of frequent exams from reading questions, as both changes were implemented together.

3. ONLINE COURSES

Our second analysis involves three types of purely online courses: MSU distance ($N = 155, 231, 165, 187, 163$), online intro physics courses; RELATE reformed ($N = 58, 100$), open online intro physics courses from the RELATE group at MIT; MITx ($N = 7158, 4829, 2686, 1935$), four MOOCs from MITx. Student populations for open online courses represent only certificate earners. MSU courses used the

[1]http://www.pa.msu.edu/bauer/mmp/

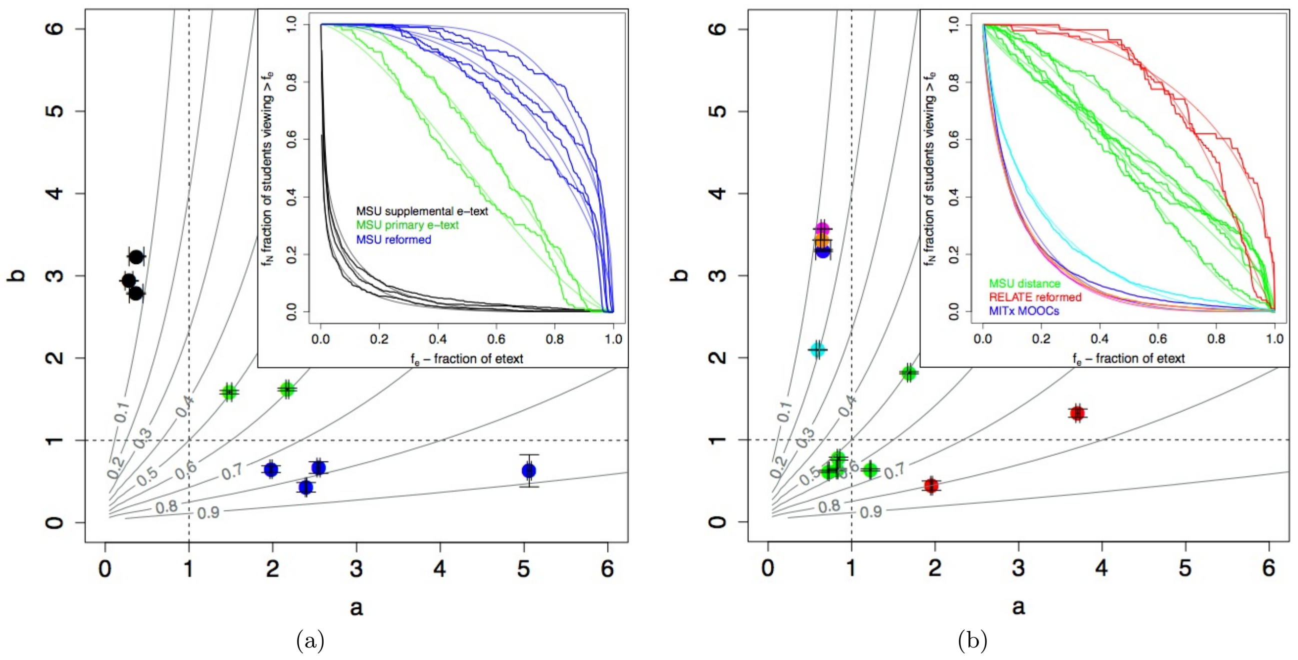


Figure 1: Etext usage via Kumaraswamy a, b-parameters for (a) blended courses and (b) online courses. Contour lines represent the average fraction of the etext viewed. CCDFs (bold) with curve fits (smooth) are displayed as an inlay.

same etext and LMS as the previously discussed blended courses. RELATE reformed courses used LON-CAPA with a Mechanics text developed by the RELATE group [5]. MITx MOOCs were disseminated through edX.

The online course data in Fig. 1(b) display similar, if slightly more nuanced, clustering of usage by course structure. MSU distance courses cluster near the saddle point ($a = b = 1$), but end up within three of the four usage regions. Since the average view fraction remains similar to the blended courses ($0.5 - 0.6$), the points on either side of the saddle point lend themselves to the following interpretation: while students in MSU primary (blended) courses typically accessed 55% of the etext (unimodal), students in distance courses viewed either more or less than this (bimodal).

The two RELATE courses in Fig. 1(b) both have similar average view fraction (≈ 0.78), but one (summer) appears in the unimodal region and the other (spring) in the high-usage region. The spring instance required students to complete all 14 weekly units; in summer, the last three units were optional. Only a small fraction of students completed the optional assignments, explaining the shift. The three MITx MOOCs in a tight cluster are computer science offerings (avf $\approx$ 0.1). A fourth, Solid-State Chemistry, lies just outside (avf $\approx$ 0.2). The MOOCs fall in the same region of the parameter space as the MSU supplemental (blended) courses, which is consistent with their similar course structure: all provide their etext as a supplement text.

4. DISCUSSION AND CONCLUSIONS

Course structure appears to have a dominant and predictable effect on etext usage distributions. Our framework, based on clustering in the two-parameter space of Kumaraswamy distributions, is easy to apply and generalizes to any finite resource type tracked within an LMS (e.g. lecture videos, homework). Instructors may exploit the dramatic correlation of course structure and etext usage to encourage particular student usage of their online materials.

Acknowledgments

Work partially supported by NSF grant DUE-1044294. We thank G. Kortemeyer, J.M. Van Thong, and Piotr Mitros, as well as edX, TLL, and ODL at MIT.

5. REFERENCES

[1] S. Kolowich. Who Takes MOOCs?. *Inside Higher Ed*, 2012.

[2] J. Pearson, G. Buchanan, and H. Thimbleby. The reading desk: Applying physical interactions to digital documents. In *Proceedings of the 2011 annual conference on Human factors in computing systems.* ACM, 2011.

[3] T. Peckham and G. McCalla. Mining Student Behavior Patterns in Reading Comprehension Tasks. In *5th International Conference on Educational Data Mining*, 2012.

[4] T. Stelzer, G. Gladding, J. P. Mestre, and D. T. Brookes. Comparing the efficacy of multimedia modules with traditional textbooks for learning introductory physics content. *American Journal of Physics*, 77(2):184, 2009.

[5] R. E. Teodorescu, A. Pawl, S. Rayyan, A. Barrantes, and D. E. Pritchard. Toward an Integrated Online Learning Environment. In *Proceedings of the Physics Education Research Conference*, pages 321–324, 2010.

Data preprocessing using a priori knowledge

Jean Simon
Université de La Réunion
IUFM de La Réunion Allée des Aigues Marines
97400 Saint Denis de La Réunion France
00262262904325
jean.simon@univ-reunion.fr

Abstract

In this paper we propose one possible way to preprocess data according to Activity theory. Such an approach is particularly interesting in Educational Data Mining.

Keywords

Activity theory, data preprocessing, preservice teacher

1 INTRODUCTION

This paper relies on the following approach:
1. Activity theory [1] is frequently used to study human activity specially CSCW and CSCL because this theory is particularly suitable to understand what the people do when they cooperate or collaborate.
2. One major problem in data-mining consists in the data preprocessing. Better those data are preprocessed, more information their treatment makes it possible to obtain.
3. The idea developed here is thus to rely on Activity theory to preprocess the data before their treatment. This preprocessing should make it possible to get more interesting results to study what occurs on a CSCW platform.

In a first time, we present the methodology we have adopted and, in a second, the application of this methodology to the analysis of the traces left during five years by the preservice teachers of the Reunion Island teacher training school.

2 METHODOLOGY

Activity theory (AT). As we can see on Figure 1, in the activity, the subject pursues a goal that results in an outcome. For this, he uses tools and acts within a community. His relation to this community is defined by rules. To achieve the goal, it may be necessary to establish a division of labor within the community. For example, in the context of hunting, there will be hunters and beaters.

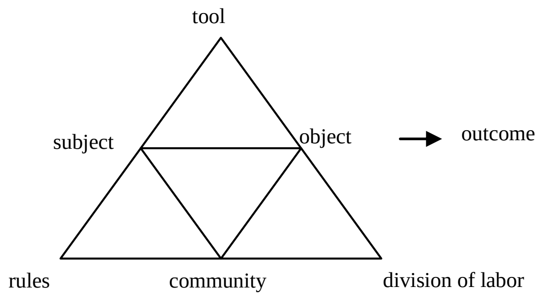


Figure 1. Triangles of AT according to Engeström [1].

Using Activity Theory to understand what happens on a platform is very common in the field of CSCL. For Halverson [2], it is powerful for, at least, three reasons: 1. this theory names well its theoretical constructs which are useful to manipulate data. 2. In this theory, the individual is at the center of everything and this is fundamental when we study learning. 3. Activity system diagrams highlight the processes and show both descriptive and rhetorical power.

Data mining. In education, the use of groupware and learning management system grows increasingly. Most of these systems record the traces of the users' activity. These traces constitute huge masses of data and permit to study the real activity of the subjects. Very broadly, the objective is to analyze what works and what does not in a given device to improve it. For Romero & al [3] there are four essential steps in data-mining: Collect data, Data preprocessing, Apply data mining, Interpret, evaluate and deploy the results. Data preprocessing is an important point for data mining because data tend to be incomplete, noisy and *inconsistent.*

Using AT to preprocess the data. What we propose is to use AT to preprocess the data and obtain a higher-level, representation. If we take the diagrams of Figure 1 we identify immediately three types readily available data:

- the tool: it will be the traces of actions on the platform and objects on which these actions operate (e.g., document deposit, document reading),
- the subject : it will be the users registered on the platform,
- the group : every CSCL platform keeps traces of the groups created on it.

Moreover, the traces of the links between these three types of data are also easily accessible.

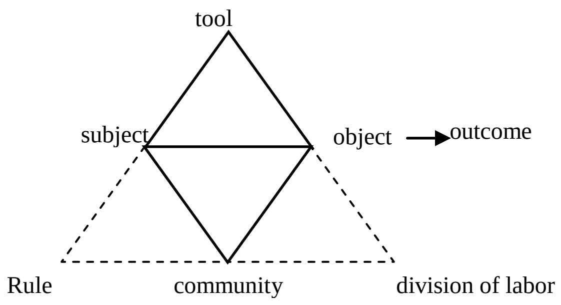


Figure 2. AT for data mining. Solid lines indicate data which are found among traces left on a CSCL platform and dotted lines information that it will be necessary to infer.

On any platform, it is recorded who are members of a particular group (dyad: subject-group) and who did what (dyad: subject-tool). From these two dyads, we can establish the third one (dyad: group-tool). If the node "objective" is rarely the subject of a specific trace, it is sometimes possible to find it through the name of the group or the name of the main folder that the group shares. The node "division of the labor" is rarely explicit and, concerning the last node, it is exceptional to have traces of the "rules within the group".

It is easy to see how technically we can preprocess the data. In the case mentioned above, instead of studying unique data, such as, for example, the "actions" alone carried out on the platform, we study couples “subject-action ", who is doing what on the platform. We can continue the process. Thus it is possible and desirable to work with 3-tuples "subject-group-action" and when you can identify the objective, 4-tuples “group- objective-subject- action” which correspond to the solid lines of figure 2. Once these 4-tuples are built, it is possible to put the focus on one of the nodes, the subject, the tool, the goal the group, without losing data interdependencies.

3 APPLICATION

The Reunion Island teacher training school trains the primary school teachers (PE2s : professeurs des écoles 2ème année). Since 2005, PE2s use a CSCW platform which allowed them to pool and share the work of preparation of the class. With their trainers, the platform has served various purposes during 5 years: to deposit documents (“collective memory”), to improve lesson plans, to help online and at distance trainees when they are in charge of a class, to validate the C2i2e which confirms that the trainee is able to use ICT in education…

As platform, we chose BSCW essentially because users may structure as they wish spaces they have created on it.

In BSCW, information is organized hierarchically in folders and sub-folders and is presented in the form of various documents (texts, pictures, URL ...) which are created, read, annotated, modified, restructured... So it is possible, to connect all the traces to the higher folder shared by a group and to reconstitute the 3-tuples (group, subject, action). Moreover, as a name is associated with each of these folders, name often indicating the purpose, it is also possible to build 4-tuples (group, goal, subject, action). We can then study the different groups or the different objectives or the different subjects or the different actions.

Table 1. comparison of the groups with or without trainers

Groups without or without trainer over 5 years	all	*PE2s*	*PE2s + trainer*	TICE
Total number of groups	960	668	292	68
Average number of PE2s for one group	15	13	20	13
Average number of documents for one group	15	6	34	41
Average number of PE2s’documents for one group	12	6	25	39
Average number of PE2 producers for one group	3	2	5	11
Average number of documents per producer for one group	4	3	5	4

Analysis according to the groups. We wanted to see what the actions were, depending on whether the group was composed only of PE2s or whether a teacher shared the group with them (PE2s+trainers). We made this distinction because we wanted to know if trainees would use the platform without being forced by the trainers. Here, we take into account only one action “document deposit”.

The 1167 PE2s have constituted more than 960 groups. 668 were groups of PE2s only and 292 groups included at least one trainer. One PE2, of course, could be in several groups. We can see that PE2s freely use the platform : the number of groups shared only by them is significantly greater than the number of groups shared with trainers (668 vs. 292). However, we find that the activity is much higher in groups shared with trainers than in groups shared only by PE2s in productions. There are fewer documents in the groups PE2s than in the groups PE2s+trainers (6 vs. 34). We can suppose that there is an effect “teachers” that incites students to work more.

Analysis according to the objective. With the titles of the folders, it was easy to isolate the groups named “TICE” (ICT for education). The folders associated with these groups were used to validate the C2i2e. All those groups have a trainer as member. In table 1, we have therefore compared those “TICE” groups with all the groups with trainers. As we can see, according to the objective, the actions on the platform reveal that activity is not the same. The “TICE” groups are smaller (13 members vs. 20), but in those groups, almost every PE2 works : 11/13 deposit. As there are more PE2s producers (11 vs. 5), there are also more documents in the “TICE” groups even if each PE2 producer deposits fewer documents (4 vs. 5).

4 CONCLUSION

The wealth of data mining relies on the fact that this is a bottom-up approach. The researcher starts from the data and expects that the machine will propose a categorization of these data of which he will try to find the underlying rules or, even better, that the machine itself will give rules of the categorization. In this approach, it is assumed that, somehow, the researcher has no a priori knowledge about the data. This approach is very interesting because it can lead to direct research in an unexpectedly way. However, it often happens that the proposed categorization is not exploitable for the researcher [4]. It is therefore desirable to reduce the hypothesis space that the machine is likely to return. One possible way to do this is to preprocess the data. In the context of CSCL, what we propose is to use Activity theory as a priori knowledge to do it. As we can see by this way we obtain results easily exploitable.

5 REFERENCES

[1] Engeström Y. 1987. *Learning by expanding: An Activity-Theoretical Approach to Developmental Research.* Orienta-Konsultit Oy.

[2] Halverson, C. A. 2002. Activity Theory and Distributed Cognition: Or what does CSCW need to do with theories ? *Computer Supported Cooperative Work (CSCW)*, 11(1-2), 243-267.

[3] Romero, C., Ventura, S., García, E. 2007 Data Mining in Course Management Systems: MOODLE Case Study and Tutorial. *Computers and Education,* 51. pp. 368-384.

[4] Talavera, L., & Gaudioso, E. 2004. Mining student data to characterize similar behavior groups in unstructured collaboration spaces. In Workshop on artificial intelligence in CSCL. *16th European conference on artificial intelligence* (pp. 17–23)

Discovering the Relationship between Student Effort and Ability for Predicting the Performance of Technology-Assisted Learning in a Mathematics After-School Program

Jun Xie[1], Xudong Huang[1], Henry Hua[1], Jin Wang[1], Quan Tang[1], Scotty D. Craig[2], Arthur C. Graesser[1], King-Ip Lin[1], Xiangen Hu[1]
[1]University of Memphis
Memphis, TN, 38152, USA
{jxie2, xhuang3, hyhua, wjin, quantang, grasser, davidlin, xhu } @memphis.edu
[2]Arizona State University
Polytechnic, Mesa, AZ, 85212, USA
scotty.craig@asu.edu

ABSTRACT

This study explored the relationship between students' math ability and effort in predicting 6th grade students' performance in the Assessment and LEarning in Knowledge Spaces (ALEKS) system. The students were clustered into four groups by K-means: high ability high effort, high ability low effort, low ability high effort and low ability low effort. A one-way ANOVA indicated that student's math posttest within the high ability, high effort group was significantly higher than other groups. An interaction was therefore observed between ability and effort. Further analysis revealed that math ability and effort had a multiplication impact on students' math posttest. That is, expending effort improves student's math posttest but how much progress in mathematics is achieved depends on the student's math ability. Higher students' math ability multiplies with effort in determining performance.

Keywords: After-school program, ALEKS, math ability, effort, math performance.

1. INTRODUCTION

Advanced learning environments [1, 4] are designed to create the most effective learning gains for students, but students vary, and not all students benefit equally from these systems. The issue then becomes discovering individual factors that maximize the usefulness of the program [9]. Studies have found that individual factors such as student effort and ability impact learning behaviors and outcomes in the computer- based learning environment [3, 8, 9].

Given the existing findings linking effort and ability with learning, this study analyzed two important characteristics on which students vary: effort and ability. We investigated the extent to which effort and math ability differentially influenced learning gains in a Web-based intelligent tutoring system (ITS) called ALEKS (the Assessment and LEarning in Knowledge Spaces) [5].

2. METHODS

Students attended the ALEKS based afterschool program two days a week for two hours a day over 25 weeks. But the learning period each day was only 1- hour, which was divided into three segments by two 20-minute breaks. Therefore the full dosage of a student was 50 hours. The 5th grade scores of the Tennessee Comprehensive Assessment Program (TCAP) were employed to assess students' pre-program mathematics knowledge ability. The 6th grade scores of TCAP served as the posttest scores.

The data sample included 268 student volunteers from 5 middle schools in a west Tennessee (US) school district. They spent time in ALEKS between 10 hours and 40 hours.

3. ANALYSIS

The independent variables were math ability and effort. Math ability was measured by TCAP 5th grade scores. For data mining purposes, we measured student effort by the ratio of the total number of mastered topics divided by the total number of attempted topics. Although it was not a pure measure of effort, this ratio was a reliable method for contrasting student learning with persistence. The first reason was that by using artificial intelligent to map each student's knowledge, ALEKS offered the topics he/she was ready to learn right now [2]. For students, the distances were equal between the difficulty of each student's topics and his/her math ability. Once they seriously tried, the students would master them. Second, ALEKS required students to write down each step when solving every problem, thence students had no chance to guess the right answers to the problems. Third, this effort measure would not be contaminated by students absent –minded learning behaviors such as spent time clicking on computer screen randomly. Therefore, this ratio reflected students' true effort in ALEKS. The dependent variables are math posttest which is TCAP 6th grade scores. All the variables were normalized.

3.1 Clustering Math Ability and Effort

The clustering goal was to divide the students into low ability high effort (LH); high ability low effort (HL); high ability high effort (HH); low ability low effort (LL). We preferred a data-driven approach that examined the data to determine the right groups. To this end, this study applied the K-means clustering algorithm with Euclidean distance based on the two attributes. The Waikato Environment for Knowledge Analysis (WEKA) was conducted to cluster the data. Table 1 showed the mean of each attribute and the number of students in each group.

Table 1. Clustering results on math ability and effort

Attributes	LH (N=121)	HL (N=73)	HH (N=21)	LL (N=53)
Math ability	-.15	.40	1.21	-.78
Effort	.03	-.07	1.13	-.78

The K-means method did not provide a clear index to indicate whether it distinguishes students notably based on math ability and effort. In order to solve this problem, a one-way ANOVA was conducted to further test the distinction between each group on ability and effort. Results indicated that math ability was significantly different between groups. Effort in each group was also significantly different, except between low ability high effort group and high ability low effort group. Therefore, the groups were distinctive in general.

3.2 Relationship between Math Ability and Effort for Predicting Posttest

One-Way ANOVA was conducted to test the difference of math posttest among four groups. The analysis indicated a significant difference of math posttest based on groups $F(3, 264) = 28.215$, $p = .000$; $\eta^2 = .243$. The Multiple Comparisons analysis found that the math posttest scores in four groups were significantly different between each other. In the four groups, the highest and lowest posttest scores were respectively within high ability high effort group and low ability low effort group. The posttest scores within low ability high effort group were lower than the high ability low effort group. It appeared that there was an interaction between ability and effort.

Univariate analysis was used to test the interaction between ability and effort. It found a significant interaction, $F(1, 264) = 5.682$, $p = .018$; $\eta^2 = .021$. The results demonstrated that students' math posttest with high effort were significantly higher than those expending low effort. However, students with high ability achieved significantly higher math performance than those with low ability. In other words, math ability and effort appeared to have a multiplicative impact on math posttest scores. Figure 1 illustrated the interaction of math ability and effort.

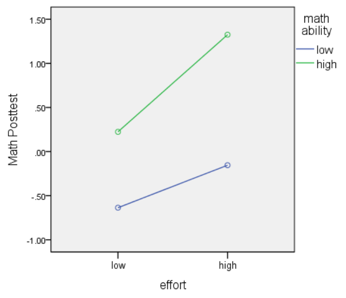


Figure 1. The interaction between math ability and effort

4. DISCUSSIONS

This study applied K-means to cluster students based on the levels of math ability and effort in ALEKS. The notable result was that in ALEKS learning system, math ability and effort had a multiplying interaction on students' math posttest. Expending effort improved students' math performance, but particularly for high ability students. It illustrated that ALEKS could help students to improve their math performance when they had motivation to learn math.

This study applied the new effort measure instead of the self-report method adopted in previous studies [7] because we sought a more objective approach to measure effort from educational data mining angle. Our method emphasized mental effort in ALEKS, which stressed students' attention and involvement in tasks [6], more than physical effort. In the future, we can also consider to measure effort from physical aspect, for instances by using the mean of time on one topic. Due to be customized for ALEKS or similar adaptive system, this effort measure had limited external validity to be generalized to other intelligent tutoring system. Consequently, it needs to be considered in the future study how to measure effort in different intelligent tutoring systems.

5. ACKNOWLEDGMENTS

This research was supported by the Institute for Education Sciences (IES) Grant R305A090528 to Dr. Xiangen Hu, PI. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of IES.

6. REFERENCES

[1] Craig, S. et al. 2011. Learning with ALEKS: The Impact of Students' Attendance in a Mathematics After-School Program. *Artificial Intelligence in Education* (2011), 435–437.

[2] Falmagne J.C., Cosyn E., Doignon J. P., Thiéry N. 2006. The assessment of knowledge, in theory and in practice. *Formal Concept Analysis Lecture Notes in Computer Science* 3874 (2006), 61-79.

[3] Welsh, Elizabeth T., et al. "E-learning: emerging uses, empirical results and future directions." *International Journal of Training and Development* 7.4 (2003): 245-258.

[4] Hu, X. et al. 2012. The Effects of a Traditional and Technology-based After-school Setting on 6th Grade Student's Mathematics Skills. *Journal of Computers in Mathematics and Science Teaching* 31, 1 (2012), 1738.

[5] Overview of ALEKS, http://www.aleks.com/about_aleks/overview

[6] PARSONS, JACQUELYNNE E., and Diane N. Ruble. "Development of Integration Processes Using Ability and Effort Information to Predict Outcome1." *Developmental Psychology* 10.5 (1974): 721-732.

[7] Timmers, C.F. et al. 2012. Motivational beliefs, student effort, and feedback behavior in computer-based formative assessment. *Computers & Education*. (2012).

[8] Zimmerman, Barry J., and Kallen E. Tsikalas. "Can computer-based learning environments (CBLEs) be used as self-regulatory tools to enhance learning?" *Educational Psychologist* 40.4 (2005): 267-271.

Using Item Response Theory to Refine Knowledge Tracing

Yanbo Xu
Carnegie Mellon University
RI-NSH 4105
5000 Forbes Ave, Pittsburgh, PA 15213
yanbox@cs.cmu.edu

Jack Mostow
Carnegie Mellon University
RI-NSH 4103
5000 Forbes Ave, Pittsburgh, PA 15213
mostow@cs.cmu.edu

ABSTRACT

Previous work on knowledge tracing has fit parameters per skill (ignoring differences between students), per student (ignoring differences between skills), or independently for each <student, skill> pair (risking sparse training data and overfitting, and under-generalizing by ignoring overlap of students or skills across pairs). To address these limitations, we first use a higher order Item Response Theory (IRT) model that approximates students' initial knowledge as their one-dimensional (or low-dimensional) overall proficiency, and combines it with the estimated difficulty and discrimination of each skill to estimate the probability *knew* of knowing a skill before practicing it. We then fit skill-specific knowledge tracing probabilities for *learn*, *guess*, and *slip*. Using synthetic data, we show that Markov Chain Monte Carlo (MCMC) can recover the parameters of this Higher-Order Knowledge Tracing (HO-KT) model. Using real data, we show that HO-KT predicts performance in an algebra tutors significantly better than fitting knowledge tracing parameters per student or per skill.

Keywords

Knowledge tracing, Item Response Theory, higher order models

1. Introduction

Traditional knowledge tracing (KT) [1] estimates the probability that a student knows a skill by observing attempted steps that require it, and applying a model with four parameters for each skill, assumed to be the same for all students: the probabilities *knew* of knowing the skill before practicing it, *learn* of acquiring the skill from one attempt, *guess* of succeeding at the attempt without knowing the skill, and *slip* of failing despite knowing the skill. Prior work shows that fitting such parameters for individual students can improve the model's accuracy in predicting student performance [2] or reduce unnecessary practice [3]. Such per-student parameters, however, ignore differences between skills. Fitting KT parameters separately instead for each <student, skill> pair risks sparse training data and overfitting, and under-generalizes by ignoring overlap of students or skills across pairs.

Item Response Theory (IRT) [4, 5] predicts a student's performance on an item based on the difficulty and discrimination of the skill(s) the item requires, and a one- (or low-) dimensional static estimate of the student's overall proficiency. Prior work adapted IRT to estimate the static probability of knowing a given skill [6], or dynamic changes in overall proficiency [7]. Here we *dynamically* estimate *individual skills* required in observed steps.

2. Approach

IRT's 2-Parameter Logistic model [4] estimates the probability $knew_{nj}$ of student n already knowing skill j as a logistic function of student proficiency θ_n, skill discrimination a_j, and difficulty b_j:

$$knew_{nj} = \frac{1}{1 + \exp\left(-1.7a_j(\theta_n - b_j)\right)}$$

Deriving $knew_{nj}$ instead of fitting it separately makes it a higher order model. We then fit each skill's KT parameters $learn_j$, $guess_j$, and $slip_j$. Figure 1 shows this hybrid Higher Order Knowledge Tracing (HO-KT) model's graphical representation. The observable state $Y^{(t)}$ tells if a skill is applied correctly at time t. The latent state $K^{(t)}$ models knowing it at time t; $\Pr(K^{(0)}) = knew$.

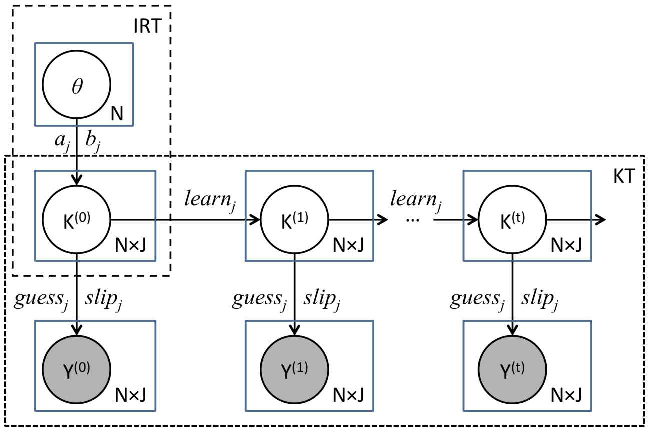


Figure 1. Graphical representation of Higher-Order Knowledge Tracing (HO-KT) model

For Markov Chain Monte Carlo (MCMC) estimation of HO-KT's parameters, we specify their prior distributions as follows:

$$\begin{aligned} \theta_n &\sim Normal(0,1) \\ b_j &\sim Normal(0,1) \\ a_j &\sim Uniform(0,2.5) \\ learn_j &\sim Beta(1,1) \\ guess_j &\sim Uniform(0,0.4) \\ slip_j &\sim Uniform(0,0.4) \end{aligned}$$

Given observations **Y,** MCMC finds vectors $\boldsymbol{\theta}$, $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{l}$ ***(learn), g (guess),*** and $\boldsymbol{s}$ ***(slip)*** with maximum posterior probability, namely:

$$P(\boldsymbol{\theta},\boldsymbol{a},\boldsymbol{b},\boldsymbol{l},\boldsymbol{g},\boldsymbol{s}|\boldsymbol{Y}) \propto L(\boldsymbol{Y}|\boldsymbol{g},\boldsymbol{s},\boldsymbol{K})P\left(\boldsymbol{K}^{(0)}\middle|\boldsymbol{\theta},\boldsymbol{a},\boldsymbol{b}\right)\times \prod_{t=1}^{T} P\left(\boldsymbol{K}^{(t)}\middle|\boldsymbol{K}^{(t-1)},\boldsymbol{l}\right)P(\boldsymbol{\theta})P(\boldsymbol{a})P(\boldsymbol{b})P(\boldsymbol{l})P(\boldsymbol{g})P(\boldsymbol{s})$$

HO-KT fits parameters to all data so far, in contrast to using IRT to fit $\boldsymbol{\theta}$, $\boldsymbol{a}$, and $\boldsymbol{b}$ to early data and KT to fit $\boldsymbol{l}$, $\boldsymbol{g}$, and $\boldsymbol{s}$ to later data.

3. Experiment

We first generated synthetic data with N=100 students, each of whom practices J=4 skills required in a series of T=100 steps. We used OpenBUGS [8] to implement MCMC estimation for HO-KT in the BUGS language. We simultaneously ran the model in 5 chains for 10,000 iterations with a burn-in of 3000, each chain starting from randomly generated initial values, and considered MCMC to converge when all 5 chains overlapped in OpenBUGS' monitor window. Table 1 shows how well the estimated value of *learn* for each simulated skill recovered its true value; estimates of other parameters were similarly accurate but omitted here for lack of space. Moreover, MCMC correctly recovered 99.4% of the simulated students' 10,000 hidden binary knowledge states.

Table 1. Estimation of *learn* in synthetic data

Skill *j*	*learn*	Estimate (95% C.I.)	s.d.	MC_error
1	0.8	0.81 (0.48, 0.99)	0.13	0.006599
2	0.6	0.60 (0.52, 0.70)	0.05	0.002132
3	0.5	0.57 (0.38, 0.84)	0.11	0.005432
4	0.3	0.29 (0.25, 0.33)	0.02	7.79E-04

We then evaluated HO-KT on real data from the Algebra Cognitive Tutor® [9], containing a total number of 41,762 observations from 123 students performing 157 problem steps. Our model assumed each problem step required a single skill. We split the data evenly into training and test sets with no overlapping <student, skill> pairs. We limited the observed sequence length of each student to T=100, and still ran 5 chains starting from random initial values for 10,000 iterations with a burn-in of 3000.

For comparison, we also used BNT-SM [10] to fit knowledge tracing parameters per skill and per student to the algebra data. The data are unbalanced (85.10% are correct steps), so we also computed within-class and majority class accuracy. Table 2 compares the models' prediction accuracy and log-likelihood on the unseen test data. HO-KT is significantly higher in overall accuracy than KT per skill and per student, with $p<.0001$ in paired T-tests comparing HO-KT to the two KT models for each of 123 students. HO-KT also achieves the best log-likelihood.

Table 2. Evaluation on real data from algebra tutor

Model:	Accuracy			Log-likelihood
	Overall	Correct	Incorrect	
HO-KT	**87.13%**	97.76%	26.43%	**-5442.50**
KT per skill	85.92%	96.19%	27.28%	-5216.23
KT per student	85.15%	99.99%	0.92%	-5102.15
Majority class	85.10%	100.00%	0.00%	--

4. Discussion

HO-KT uses IRT to estimate students' initial knowledge of a skill based on its difficulty and discrimination and their overall proficiency, and KT to model learning over time. It outperforms per-student or per-skill KT by combining information about both. HO-KT estimates every probability *Knew*(student, skill) without requiring training data for every <student, skill> pair, because it can estimate student proficiency based on other skills, and skill difficulty and discrimination based on other students.

Future work should compare HO-KT to other methods and on data from other tutors. We should also test if *k*-dimensional student proficiency captures enough additional variance to justify fitting *k* times as many parameters. Finally, extending HO-KT to trace multiple subskills should use considerably fewer parameters than prior methods [11, 12], thanks to combining IRT and KT.

ACKNOWLEDGMENTS

This work was supported by the Institute of Education Sciences, U.S. Department of Education, through Grants R305A080157 and R305A080628 to Carnegie Mellon University, and by the National Science Foundation under Cyberlearning Grant IIS1124240. The opinions expressed are those of the authors and do not necessarily represent the views of the Institute, the U.S. Department of Education, or the National Science Foundation. We thank Ken Koedinger for his algebra tutor data.

REFERENCES

[1] Corbett, A. and J. Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 1995. *4*: p. 253-278.

[2] Pardos, Z. and N. Heffernan. Modeling Individualization in a Bayesian Networks Implementation of Knowledge Tracing. *Proceedings of the 18th International Conference on User Modeling, Adaptation and Personalization*, 255-266. 2010. Big Island, Hawaii.

[3] Lee, J.I. and E. Brunskill. The Impact on Individualizing Student Models on Necessary Practice Opportunities *Proceeding of the 5th International Conference on Educational Data Mining (EDM)* 118-125. 2012. Chania, Greece.

[4] Hambleton, R.K., H. Swaminathan, and H.J. Rogers. *Fundamentals of Item Response Theory*. Measurement Methods for the Social Science. 1991, Newbury Park, CA: Sage Press.

[5] Birnbaum, A. Some latent trait models and their use in inferring an examinee's ability. *Statistical theories of mental test scores*, 1968: p. pp. 374 - 472.

[6] de la Torre, J. and J.A. Douglas. Higher-order latent trait models for cognitive diagnosis. *Psychometrika* 2004. *69*(3): p. 333-353.

[7] Martin, A.D. and K.M. Quinn. Dynamic Ideal Point Estimation via Markov Chain Monte Carlo for the U.S. Supreme Court, 1953-1999. *Political Analysis*, 2002. *10*: p. 134-153.

[8] Lunn, D., D. Spiegelhalter, A. Thomas, and N. Best. The BUGS project: Evolution, critique and future directions. *Statistics in Medicine* 2009. *28*: p. 3049–306.

[9] Koedinger, K.R., R.S.J.d. Baker, K. Cunningham, A. Skogsholm, B. Leber, and J. Stamper. A Data Repository for the EDM community: The PSLC DataShop. In C. Romero, et al., Editors, *Handbook of Educational Data Mining*, 43-55. CRC Press: Boca Raton, FL, 2010.

[10] Chang, K.-m., J.E. Beck, J. Mostow, and A. Corbett. A Bayes Net Toolkit for Student Modeling in Intelligent Tutoring Systems. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, K. Ashley and M. Ikeda, Editors. 2006: Jhongli, Taiwan, p. 104-113.

[11] Koedinger, K.R., P.I. Pavlik, J. Stamper, T. Nixon, and S. Ritter. Avoiding Problem Selection Thrashing with Conjunctive Knowledge Tracing. In *Proceedings of the 4th International Conference on Educational Data Mining*. 2011: Eindhoven, NL, p. 91-100.

[12] Xu, Y. and J. Mostow. Using Logistic Regression to Trace Multiple Subskills in a Dynamic Bayes Net. *Proceedings of the 4th International Conference on Educational Data Mining* 241-245. 2011. Eindhoven, Netherlands.

Estimating the benefits of student model improvements on a substantive scale

Michael V. Yudelson
Carnegie Learning, Inc.
437 Grant St.
Pittsburgh, PA 15219, USA
myudelson@carnegielearning.com

Kenneth R. Koedinger
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213, USA
koedinger@cmu.edu

ABSTRACT
Educational Data Mining researchers use various prediction metrics for model selection. Often the improvements one model makes over another, while statistically reliable, seem small. The field has been lacking a metric that informs us on how much practical impact a model improvement may have on student learning efficiency and outcomes. We propose a metric that indicates how much wasted practice can be avoided (increasing efficiency) and extra practice would be added (increasing outcomes) by using a more accurate model. We show that learning can be improved by 15-22% when using machine-discovered skill model improvements across four datasets and by 7-11% by adding individual student estimates to Bayesian Knowledge Tracing.

1. INTRODUCTION
In this work we are discussing an approach that translates differences in statistical metrics between the two models into the potential differences in the number of practice attempts students would be prescribed and the time students could allocate more optimally if a better-fitting student model is deployed. We consider two types of model comparisons. First, we compare alternative skill models of the problem domain while keeping the student modeling algorithm the same. Second, we compare keeping the skill model the same. Second, keeping the skill model the same and changing the student modeling algorithm. We discuss results obtained for several datasets that cover domains such as middle school algebra and geometry, English, and numberline games. Our investigation shows that, despite the improvement in model accuracy metric being seemingly small, representing the differences in terms of missed practice opportunities and time reveals substantial differences.

2. DATA
We used the datasets from the KDD Cup 2010 EDM Challenge and from the Pittsburgh Science of Learning Center (PSLC) DataShop (`www.pslcdatashop.org`): Algebra I dataset, and Bridge to Algebra collected in 2008-09. We also used 4 PSLC DataShop datasets addressing Geometry (1996-97 and 2010), Articles (2009), and Numbeline Games (2011). The KDD Cup 2010 data was donated by Carnegie Learning Inc. PSLC DataShop datasets were collected by various researcher partners of PSLC (`www.learnlab.org`). The KDD Cup 2010 Algebra I dataset has 8,918,054 practice attempts of 3,310 students and has 2 skill models: 'KTraced-Skills' (kts) used in cognitive tutor, and an alternative 'Sub-Skills' (ss) model. The KDD Cup 2010 Bridge to Algebra dataset contains data of 6,043 students comprised of 20,012,498 rows and has the two skill models as well. PSLC DataStop Geometry 1996-97 data covers of 5,388/59 transactions/students, Articles 2009 data - 6,887/120, Geometry 2010 - 140,854/120, Numberline games 2011 data - 4,341/51.

3. MODELS
We will use Bayesian Knowledge Tracing (BKT) [2] to fit models of student learning. BKT is a method often used in Intelligent Tutoring Systems (ITS). In addition to standard BKT, we will use an individualized BKT (iBKT) models described in [3]. Namely, the model where the *p-learn* has a per-skill and per-student component. We implemented a tool capable of fitting standard and individualized BKT models on large datasets (such as KDD Cuo 2010 dataset) in an efficient way. Our tool is implemented in C/C++ and can fit BKT models from large datasets very quickly. For more details please refer to [3]).

4. METHOD
First, we fit original and alternative models for all of the datasets and skill modelswe have. For the two KDD Cup 2010 datasets we fit BKT and iBKT models. For the four DataShop datasets, we fit BKT model only. Out of several skill models available for each DataShop dataset we select original one and the best skill model discovered using a human-machine Learning Factors Analysis procedure [1].

We then compute probabilities of skill mastery for all student attempts. We used a threshold probability of 0.95 (a traditionally accepted value) to determine the moment of mastery. If, according to the model, student did not reach mastery for a particular skill within the recorded student data, we calculate the number of under-practice attempts. If student's skill reaches mastery earlier than the latest attempt recorded, we compute the number of over-practice attempts. The mastery data is aggregated by student taking under-practice attempts into consideration.

Table 1: Comparing models in terms of root mean squared error, percent cases number of prescribed practice opportunities differs by at least one, average student/skill practice opportunities, and time

(a) Estimated prediction improvements and *practical benefits* of replacing hand-made by LFA machine-discovered KC models across four DataShop datasets (RMSE values are given for a student-stratified 10-fold cross-validation).

	Time	KCs		RMSE		% diff Orig.-LFA			Mean stud. opp/KC				Stud. time		
Dataset	/step	Orig.	LFA	Orig.	LFA	≤ -1	(-1,1)	≥ 1	Orig.	LFA	diff	%	total	diff	%
Geometry 1996-97	17.12s	15	18	0.410	0.400	10%	29%	61%	8.8	7.8	1.5	18-20	26m	2m	9
Articles 2009	15.09s	13	26	0.437	0.420	5%	53%	43%	7.9	7.1	1.2	15-17	14m	3m	23[†]
Geometry 2010	15.10s	46	43	0.240	0.239	95%	5%	0%	8.6	10.5	1.9	18-22	88m	6m	7
Numberline 2011	12.77s	12	22	0.459	0.457	41%	22%	37%	15.1	15.2	2.8	19	18m	32m	182[†]

[†]These values could be inflated due to absence of mastery learning in respective tutors and as a result the amount of student work being less optimal.

(b) Estimated prediction improvements and *practical benefits* of replacing standard BKT models by individualized BKT models across two KDD Cup 2010 datasets (RMSE values are given for a student-stratified 10-fold cross-validation).

	Time		RMSE		% diff BKT-iBKT			Mean stud. opp/KC				Stud. time		
Dataset	/step	KCs	BKT	iBKT	≤ -1	(-1,1)	≥ 1	BKT	iBKT	diff	%	total	diff	%
Algebra 1(kts)	n/a	515	0.363	0.361	24%	72%	4%	12.1	12.9	1.1	9	n/a	n/a	n/a
Algebra 1(ss)	n/a	541	0.342	0.341	34%	63%	3%	12.5	13.6	1.4	10-11	n/a	n/a	n/a
B.to Algebra(kts)	12.81s	807	0.363	0.359	22%	74%	5%	14.3	14.9	1.0	7	361m	15m	4
B.to Algebra(ss)	12.81s	933	0.359	0.355	27%	68%	5%	18.3	19.2	1.2	7	485m[†]	22m	5

[†]Difference in times between *its* and *ss* KC models is due to change in the subset of data selected.

Finally, we compute the time it takes a student to solve one tutor step. This time is used to compute the typical length of all student sessions in the system. Having the sum of the number of practice opportunities it takes the student to master all skills (correcting for under-practice) from the both models being compared and plugging in the average step duration, we compute the overall amount of time student *wastes* for under-practicing and over-practicing.

5. RESULTS AND DISCUSSION

Table 1 is a summary of model comparisons. Table 1a compares original skill models and best fitting machine-discovered skill models for the DataShop datasets. Table 1b compares standard BKT and individualized BKT modeling methods for the same skill models in KDD Cup 2010 datasets. Despite the vast difference in the size of the datasets (inherently the size of curriculum), improvements with respect to student-stratified cross-validated RMSE are quite small. Just like the improvements in RMSE, the mean absolute difference in mean student opportunities are small: from 1.1 to 2.8 practice attempts. However, in terms of percent practice opportunities, those differences constitute 15-22% in DataShop datasets, and 7-11% in KDD Cup 2010 datasets.

The practice opportunity differences are shown in Table 1a and Table 1b under *% diff Orig-LFA* and *% diff BKT-iBKT* respectively. Here the column marked '≤ -1' indicates the percent of student-KC experiences for which the model built on the LFA-discovered KC model prescribes at least one opportunity /less/ on average than the model built on the Original KC model. Similarly, column '≥ 1' indicates the percent that the LFA-discovered skill model prescribes at least one *more* opportunity..

The overall amount of time students spend with the tutor differs from dataset to dataset: from 14-18 mimutes to 8 hours. The absolute and percent time values for time differences in Table 1 reflect both over-practice and under-practice together. The absolute average and percent average time difference between the models are given next to the total time students spend o average. The percent of the time students are *wasting* is 7-9% on Geometry DahaShop datasets (able 1a) and 4-5% on Bridge to Algebra KDD Cup 2010 dataset (Table 1b). A higher values of nearly a quarter of time misused (23%) in the case of Articles 2009 dataset and almost twice the time (182%) misused on the Numberline 2011 DataShop dataset, are due to the fact that both did not implement mastery learning and, contrary to the cases of tutors used to collect other datasets, problems were not sequenced in attempt to maximize students' learning.

6. ACKNOWLEDGMENTS

This work was supported by the PSLC DataShop team, Carnegie Learning Inc., and National Science Foundation (award #SBE-0836012). Most of the work was done when the first author was at Carnegie Mellon University.

7. REFERENCES

[1] H. Cen, K. R. Koedinger, and B. Junker. Learning factors analysis - a general method for cognitive model evaluation and improvement. In M. Ikeda, K. D. Ashley, and T.-W. Chan (Eds.), *8th International Conference on Intelligent Tutoring Systems (ITS 2006)*, pp. 164–175, Jhongli, Taiwan, June 26-30, 2006. Springer.

[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, 1995.

[3] M. V. Yudelson, K. R. Koedinger, and G. J. Gordon. Individualized bayesian knowledge tracing models (in press), In: Artificial Intelligence in Education, 2013.

A Dynamic Group Composition Method to Refine Collaborative Learning Group Formation

Zhilin Zheng
Department of Informatics
Clausthal University of Technology
Clausthal-Zellerfeld, Germany

zhilin.zheng@tu-clausthal.de

ABSTRACT

Group formation strategies have the goal of providing the participating students with the good initial conditions for collaborative learning. Continuing with the existing methods to set up the initial conditions to make peer interaction more likely happen, we propose a method for dynamically recomposing learning groups based on intra-group iteration analysis to optimize the learning group formation iteratively.

Keywords

Group Formation, Educational Data Mining, Collaborative Learning, Dynamic Group Composition, Interaction Analysis

1. INTRODUCTION

Group formation plays a critical role for the success of collaborative learning groups [2]. Through pedagogical experiments, both homogeneous and heterogeneous group formation strategies can effectively promote collaboration [1]. In order to compose heterogeneous or homogeneous learning groups, plenty of composition approaches have been suggested [2; 3; 6]. These approaches pay most attention to the performance of the proposed algorithms, such as solution optimization and time cost, while the peer interaction within the formed groups is typically not considered for refining the groups. In addition, some data mining technologies have recently been proposed to analyze the peer interaction, with results indicating that there are recurring interactions within groups with strong peer interaction [7]. Therefore, if we could find some way of group composition which would lead to groups showing these interaction patterns, then an effective peer interaction within these groups might be triggered with higher probability.

2. PROPOSED APPROACH

The proposed method is to dynamically recompose groups based on interaction analysis. We expect to distinguish groups with strong interaction from weaker ones and learn group composition rules from this. In this paper, group composition rules denote that which types of group members work together could trigger either strong or weak peer interaction. Initially, the collaborative groups are composed by existing composition approaches (e.g. Graf and Bekele's method)[2]. Learners in each group are then instructed to complete team tasks collaboratively. After the completion of the tasks, the peer interaction in the learning groups is analyzed. Data mining technologies are used to extract interaction patterns (e.g. sequential patterns) from group interaction logfile. These patterns together with tutor's assessment could be used to distinguish the effective interaction groups from the weak ones. Based on this classification and group member compositions, the group composition rules can be learned using decision tree induction methods. These composition rules are used to re-group the learners into a new group formation. At the new group formation, learners are given new collaborative learning tasks. After the completion of these new tasks, new interaction patterns are extracted again, and new group composition rules are learned as well. Then, this new set of composition rules is utilized to re-group learners again. This grouping process is kept on iteratively. Over time, the group formation will change dynamically, with the goal of composing the groups with highest chances for effective peer interactions.

3. PLANS FOR SYSTEM DESIGN

A software system for the proposed method of dynamically recomposing learning groups is designed, which is outlined as shown in Figure 1. The following sections describe the primary components of the system.

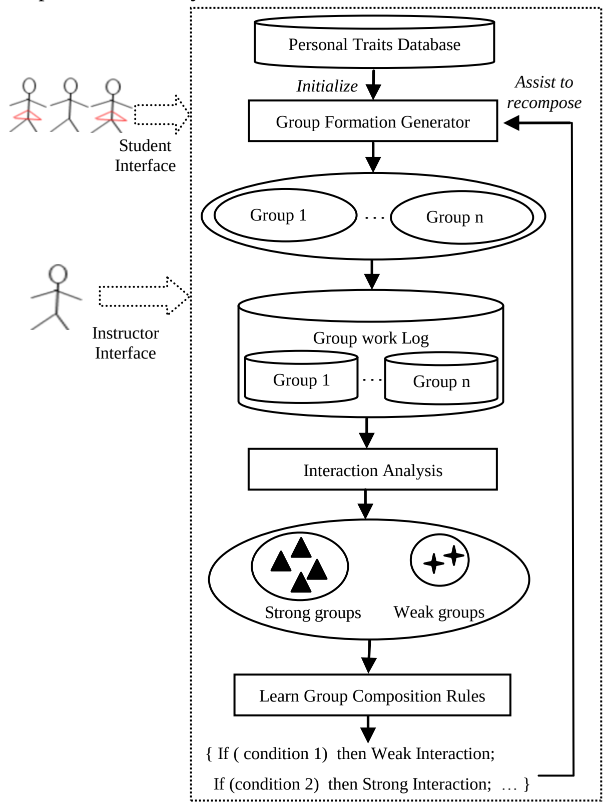


Figure 1. Outline of dynamically recomposing system

3.1 Student Interface
Student interface of the system has twofold functions. First of all, it is designed to collect learners' personal characteristics. In the light of the previous research on this topic [2-4; 6], leadership, previous knowledge, interest for the subject, group work attitude, self-confidence, shyness, gender are surveyed by questionnaires. The surveyed result for each student is then stored in a personal traits database. In addition, learners can choose one or more group members to interact with each other through the student interface.

3.2 Instructor Interface
Using instructor interface to the system, tutors can post collaborative tasks for each group, monitor groups working collaboratively, and assess the groups (e.g., for outcome quality).

3.3 Group Formation Generator
Group formation generator includes two functions. One is to generate the initial group formation and the other is to produce a new group formation at each recomposing iteration. Initially, we employ the Graf and Bekele's approach to compose heterogeneous learning groups [2]. At each iteration of recomposing, we first use an exhaustive method to generate all possible group formations. For each group formation, we then count up the groups which are predicted to produce strong peer interaction according to group composition rules. At the end, the group formation with the most "high potential" groups is selected as the final group formation for the next iteration.

3.4 Interaction Analysis
Interaction analysis is to verify the effective interaction really happen in learning groups. The effectiveness of interaction should be measured by both the outcome of group work and frequency of interactive events. Tutors are able to assess the outcome of group work. But it's hard for them to do assessment of the collaborative activities between group members because of the difficulties to deduce the actual peer interaction based on the interaction logfile. Fortunately, sequential pattern mining techniques have been developed to analyze peer interaction [5; 7]. The result of relevant research shows that the best interaction groups have high frequency of certain sequential behaviors [7]. Using the frequency of these uncovered sequential patterns together with the outcome of group work, the effective interaction groups could be distinguished from the negative ones.

3.5 Learn Group Composition Rules
Group composition rules indicate that which types of learners placed into a group could trigger either strong or weak peer interaction. Each member of the group is represented by a set of personal characteristics which are surveyed in Section 3.1. We firstly need to cluster all learners based on these personal characteristics. Then the resulting clusters are labeled respectively as cluster A, cluster B, etc. According to each group's performance, we can conclude the group composition and its interaction level in a dataset, as illustrated in Table 1. The numbers in the table signify how many students in each group belong to the clusters.

Decision tree learning algorithms (e.g. ID3) are then applied to construct a decision tree for classification based on the dataset. The interaction types construct the leaf node of the decision tree while the clusters of students (constructed based on personal characteristics) construct the inner nodes of the tree. When the decision tree is constructed, group composition rules are simply generated through traversing all paths from the root of the tree to every leaf node.

Table 1. Example of dataset

	Cluster A	Cluster B	…	Interaction type
Group1	1	2	…	strong
Group2	2	1	…	weak

4. CONCLUSION AND FUTURE WORK
This paper proposes a dynamic group composition method to refine collaborative learning group formation and outlines the designed software system. Our future work will be focused on implementation and evaluation of the proposed idea in a collaborative learning context.

5. ACKNOWLEDGMENTS
This research is supported by China Scholarship Council (CSC).

6. REFERENCES

[1] ABRAMI, P.C. and CHAMBERS, B. 1996. Research on Cooperative Learning and Achievement: Comments on Slavin. *Contemporary Educational Psychology* 21, 1 (Jan. 1996), 70-79.

[2] GRAF, S. and BEKELE, R. 2006. Forming heterogeneous groups for intelligent collaborative learning systems with ant colony optimization. In *the Proceedings of the 8th international conference on Intelligent Tutoring Systems* (Jhongli, Taiwan, June 26-30, 2006). ITS2006. Springer-Verlag, Heidelberg, 217-226.

[3] LIN, Y.T., HUANG, Y.M., and CHENG, S.C. 2010. An automatic group composition system for composing collaborative learning groups using enhanced particle swarm optimization. *Computers & Education* 55, 4 (Dec. 2010), 1483-1493.

[4] LOU, Y., ABRAMI, P.C., SPENCE, J.C., POULSEN, C., CHAMBERS, B., and D'APOLLONIA, S. 1996. Within-Class Grouping: A Meta-Analysis. *Review of Educational Research* 66, 4 (Winter, 1996), 423-458.

[5] MARTINEZ, R., YACEF, K., KAY, J., KHARRUFA, A., and AL-QARAGHULI, A. 2011. Analysing frequent sequential patterns of collaborative learning activity around an interactive tabletop. In *the Proceedings of the Fourth International Conference on Educational Data Mining* (Eindhoven, The Netherlands, July 6-8, 2011). EDM 2011.111-120.

[6] MORENO, J., OVALLE, D.A., and VICARI, R.M. 2012. A genetic algorithm approach for group formation in collaborative learning considering multiple student characteristics. *Computers & Education* 58, 1 (Jan. 2012), 560-569.

[7] PERERA, D., KAY, J., KOPRINSKA, I., YACEF, K., and ZAIANE, O.R. 2009. Clustering and sequential pattern mining of online collaborative learning data. *IEEE Transactions on Knowledge and Data Engineering* 21, 6 (June 2009), 759-772.

Poster Presentations (Late Breaking Results)

Educational Data Mining: Illuminating Student Learning Pathways in an Online Fraction Game

Ani Aghababyan[1]
Taylor Martin[2]
Utah State University
Instructional Technology
and Learning Sciences
2830 Old Main Hill
Logan, UT 84322-2830
+1-435-754-6285[1]
+1-435-797-0814[2]
anie.aghababyan@gmail.com[1]
taylormartin@usu.edu[2]

Nicole Forsgren Velasquez
Utah State University
Management of Information Systems
3515 Old Main Hill
Logan, UT 84322-3515
+1-435-797-3479
nicolefv@gmail.com

Philip Janisiewicz
University of Texas at Austin
Curriculum and Instruction
1 University Station D5700
Austin, TX 78712
(512) 496-1829
pjanisiewicz@gmail.com

ABSTRACT

This study investigates ways to interpret and utilize the vast amount of log data collected from an educational game called Refraction to understand student fraction learning. Study participants are elementary students enrolled in an online virtual school system who played the game over the course of multiple weeks. Findings suggest that students use a variety of splitting strategies when solving Refraction levels and that these strategies are related to learning gains.

Keywords

Educational data mining; hierarchical clustering; learning analytics; mathematics education; fractions; educational games.

1. INTRODUCTION

Electronic games have become a regular part of childhood and adolescence [1]. In recent years, the interest in games for learning has grown, and educational games have increased in their popularity as means of instruction [3]. These games are unstructured environments where students can learn educational concepts through engaging interfaces and at their own pace.

Educational data mining techniques have the potential to illuminate learning patterns across a large number of students who play these games. By analyzing the data generated through these activities and assignments, data scientists can gain insights into when students have mastered a concept or skill, what excites them, where they are getting stuck, and what works to support learning. The ability to discern this for each student and for all students is a key contributing factor in improving the quality of education in the U.S.

2. REFRACTION

Refraction (http://play.centerforgamescience.org/refraction/site/) is an online game based on fraction learning through splitting. It is an open-access, interactive, and spatially challenging game that allows researchers to discover students' fraction learning pathways. In the game level used for this study, students are required to create laser beams of 1/6 and 1/9 using a combination of 1/2 and 1/3 splitters. Four 1/2 splitters, four 1/3 splitters, and seven benders are provided to achieve this goal. One possible solution is shown in Figure 1(b).

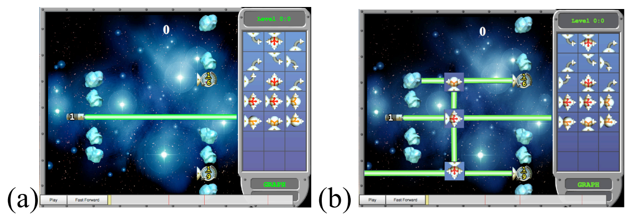


Figure 1. Refraction pretest and posttest levels (a) at the start of a gameplay and (b) at the completion of the level

3. METHOD

3.1 Procedures

The game begins with a short series of introductory levels, and then students play an in-game pretest level. Following gameplay, which can include several levels, they play an in-game posttest level. The pretest and posttest levels are identical. For this study, we mined only the in-game pretest and in-game posttest levels. We chose this problem because it requires students to move beyond repeated halving (such as creating 1/4 or 1/8), and it requires students to use a combination of 1/2 and 1/3 splitters. Players reach the pretest after introductory levels that teach them the mechanics of the game, in order to avoid any pre-post differences simply being attributable to game's unfamiliarity.

3.2 Variables

Every time a splitter is placed on the laser beam in the Refraction environment, a new board state is logged. We used this data to examine the process of learning by splitting using hierarchical cluster analysis and included the following variables:

a. *Initial 1/2*: this is the percent of board states in a level that have 1/2 splitter as the initial splitter.
b. *Initial 1/3*: this is the percent of board states in a level that have 1/3 splitter as the initial splitter.
c. *Backtrack*: this is a binary variable indicating whether the player returned to using 1/2 as the initial splitter after having used 1/3 as the initial splitter.
d. *Average distance from goal*: Average distance from each board state to the goal state.

3.3 Hierarchical Clustering

Cluster analysis is a common technique for classifying a large amount of information into meaningful groups [2]. Hierarchical cluster analysis was conducted using between-groups linkage, within-groups linkage, centroid clustering, median clustering, and Ward's method [4]. The results for cluster solutions with two to seven clusters were compared in terms of: (a) change in agglomeration coefficients; (b) number of cluster memberships (number of students in each clusters solutions; and (c) results of univariate F-tests (solutions wherein the clusters did not differ in terms of any of the classifying variables were excluded). Based on these analyses, we found the solution with four clusters using between-group linkage method performed the best. Based on F statistics (see Table 1), these four clusters were significantly different from each other in terms of each of the four variables presented above.

Table 1 Cluster definitions based on Duncan post hoc multiple range test.

Variables	*F-values*[a]	*Cluster 1 Halving Strategy*	*Cluster 2 Thirds Strategy*	*Cluster 3 Exploring Thirds Strategy*	*Cluster 4 General Exploring Strategy*
Backtrack	1017.187***	0.58 M[b]	0.00 L	0.00 L	0.63 H
Initial 1/2	17415.955***	3.57 VH	1.00 L	1.04 M	2.99 H
Initial 1/3	4800.857***	1.46 L	3.53 VH	1.82 M	2.88 H
Avg. Distance from Goal	5816.343***	3.58 VH	1.44 L	3.03 H	2.12 M

[a] The significance levels of these F-values are indicated as follows: *** 0.001 level; ** 0.01; and * 0.05 level.

[b] VH, H, M, and L indicates that the mean for the cluster was very high, high, medium, or low, respectively, based on Duncan's Multiple Range Test. If two clusters have the same letter, that indicates that these clusters' means did not have significantly different means.

To interpret the clusters, post hoc comparisons of the means of all four clustering variables were performed. Duncan's Multiple Range Test was used to compare the means of these four variables across four clusters (Hair, Anderson, Tatham, & Grablosky, 1979). In this test, pairwise comparisons are done across clusters and significant differences are identified at the pre-defined significance level; in this case, $p < 0.1$. Furthermore, the test sorts the clusters into groups wherein the means of the clusters within a group are not significantly different from each other, but differ at a statistically significant level from clusters in other groups. For example, for the variable backtrack, the test sorted our four clusters into three distinct groups, as seen by the designation of L, M, and H in Table 1. In this case, the mean of Cluster 1 is significantly higher than the means for Clusters 2 and 3, but significantly lower than the mean for Cluster 4.

4. RESULTS

Clustering results show that there are four distinct ways that students solved the pre-post level. The four clusters can be described as follows:

a. *Halving strategy:* Students using this strategy are primarily exploring the 1/2 space of the game. They display a high percentage of board states that start with a 1/2 splitter. They also have high average distance from the goal and a low percentage of board states that start with a 1/3 splitter. When they do use the 1/3 splitter, they often backtrack to using the 1/2 splitter.

b. *Thirds strategy:* Students using this strategy spend the majority of their time in the 1/3 space of the game. They have a high percentage of 1/3 initial board states. They rarely backtrack. Their average distance from the goal is low.

c. *Exploring Thirds Strategy:* While students using this strategy still experiment with initial 1/2 board states, they have a higher percentage of 1/3 initial board states. They do not backtrack often, but still have a high average distance from the goal.

d. *General Exploring Strategy:* Students using this strategy are exploring the mathematical space of the game more broadly. They have high percentage of board states using both the 1/2 and 1/3 splitters, and they backtrack often. They have medium average distance from the goal.

We conducted one-way ANOVAs on the classifications of students' game play strategy (pre- and postlevel) for both the pre- and posttest. We found that there was a significant main effect for prelevel strategy type on transfer pretest score, $F(3, 2494) = 7.79$, $MSE = 23.10$, $p < .001$. Post hoc tests showed that this effect was primarily accounted for by the Thirds group's significantly greater performance than the Halving and Exploring Thirds groups ($p < .05$). The General Exploring group did not perform significantly differently than any other group.

5. DISCUSSION

Overall, we found that how students used splitting on the prelevel was associated with test performance at that point, but all students developed fraction knowledge by using splitting as they played Refraction, regardless of their splitting strategy.

6. REFERENCES

[1] Greenberg, B. S., Sherry, J., Lachlan, K., Lucas, K., & Holmstrom, A. 2008. Orientations to video games among gender and age groups. Simulation & Gaming, 41(2), 238–259.

[2] Lorr, M. (1983). *Cluster analysis for social scientists*. Jossey-Bass, San Francisco, CA.

[3] Rodrigo, M., Baker, R., D'Mello, S., Gonzalez, M., Lagud, M., Lim, S., … & Viehland, N. (2008). Comparing learners' affect while using an intelligent tutoring system and a simulation problem solving game. Intelligent Tutoring Systems. 40-49.

[4] Ulrich, D. and B. McKelvey (1990), "General Organizational Classification: An Empirical Test Using the United States and Japanese Electronic Industry," Organization Science, 1, 99-118.

Automatic Gaze-Based Detection of Mind Wandering during Reading

Sidney D'Mello
University of Notre Dame
384 Fitzpatrick
Notre Dame, IN 46556, USA
(001)-574-631-8322
sdmello@nd.edu

Jonathan Cobian
University of Notre Dame
384 Fitzpatrick
Notre Dame, IN 46556
574-367-1941
jcobian@nd.edu

Matthew Hunter
MIT
233 Massachusetts Avenue
Cambridge, MA 02139
574-370-7597
mjhunter@mit.edu

ABSTRACT

We present a fully-automated person-independent approach to track mind wandering by monitoring eye gaze during reading. We tracked eye gaze of 84 students who engaged in an approximately 30-minute self-paced reading task on research methods. Mind wandering reports were collected by auditorily probing students in-between and after reading certain pages. Supervised classifiers trained on global and local features extracted from students' gaze fixations 3, 5, 10, and 15 seconds before each probe were used to predict mind wandering with a leave-several-subjects-out cross validation procedure. The most accurate model tracked both global and local eye gaze in a 5-second window before a probe and yielded a kappa (accuracy after correcting for chance) of 0.23 on a downsampled corpus containing 50% yes and 50% no responses to probes. Implications of our findings for adaptive interventions that restore attention when mind wandering is detected are discussed.

Keywords

Mind wandering, eye gaze, affective computing, affect detection

1. INTRODUCTION

Mind wandering (or zoning out) is a phenomenon in which attention drifts away from the primary task to task-unrelated thoughts [1]. It is critically important to learning because active comprehension of information involves extracting meaning from external sources of information (e.g., text, audio, image) and aligning this information with existing mental models that are ultimately consolidated into long-term memory structures. Mind wandering signals a breakdown in this coupling of external information and internal representations. Hence, it is no surprise that mind wandering has disastrous effects on learning and comprehension because it negatively impacts a learner's ability to attend to external events, to encode information into memory, and to comprehend learning materials [2, 3]. Therefore, there is a crucial need for interventions to track and restore attention when mind wandering is detected.

A system that responds to mind wandering must first detect when minds wander. In line with this, Drummond and Litman [4] attempted to identify episodes of "zoning out" while students were engaged in a spoken dialog with an intelligent tutoring system (ITS). Students were periodically interrupted to complete a short survey to indicate the extent to which they were focusing on the task (low zoning out) or on other thoughts (high zoning out). J48 decision trees trained on acoustic-prosodic features extracted from the students' utterances yielded 64% accuracy in discriminating high vs. low zone-outs. This study was pioneering in that it represents the first attempt to automatically detect zone-outs. However, it suffers from two notable limitations. First, the study used a leave-one-instance-out cross-validation method where training and testing sets were not independent; therefore it is unclear if the model generalizes to new students. Second, the model is only applicable to spoken tutorial sessions instead of more general learning tasks.

Taking a somewhat different approach, we report initial results of a study that uses eye gaze data to develop *student-independent* predictive models of mind wandering during reading. Our emphasis on reading is motivated by the fact that reading is perhaps the most ubiquitous learning activity. Our focus on eye gaze to track mind wandering is motivated by decades of scientific evidence in support of an *eye-mind* link, which posits that there is a tight coupling between external information (words on the screen) and eye movements [5]. For example, previous research has found that individuals are less likely to fixate, re-fixate, and look backward through previously read text [6] and blink more frequently [7] when mind wandering compared to normal reading. The present study builds on these findings by developing the first gaze-based mind wandering detector.

2. METHOD

2.1 Labeled Data Collection

A Tobii T60 eye tracker was used to record gaze patterns of 84 students while they read four texts on research methods (e.g., random assignment, experimental bias) for approximately 30 minutes. Students read the texts on a page-by-page basis (roughly 144 words per page) and used the space bar to navigate forward. Mind wandering was measured via auditory probes, which is the standard and validated method for collecting online mind wandering reports [1]. When a student's gaze fixated on previously determined "probe words", which were pseudo-randomly inserted in the texts, the system played an auditory cue (i.e., a beep) to prompt the student to indicate whether or not he or she was mind wandering by pressing keys marked "Yes" and "No." These probes are referred to as *in-between page probes.* In addition, *end of page probes* were triggered when students pressed the space bar to advance to the next page. There were approximately 10 probes per text and reports of mind wandering were obtained for 35% of the probes, which is comparable to rates obtained in previous studies on reading [2].

2.2 Feature Engineering

Gaze fixations were estimated from the raw gaze data using OGAMA, an open source gaze analyzer. The series of gaze fixations were segmented into windows of varying length (3 secs, 5 secs, 10 secs and 15 secs), each culminating with a mind wandering probe. The windows ended immediately before the probe was triggered in order to avoid confounds associated with motor activities in preparation for the key press. Furthermore, windows with less than five fixations were eliminated because these windows do not contain sufficient data to meaningfully compute gaze features.

Two sets of gaze features were extracted from gaze fixations in each window. *Global features* were independent of the actual words being read and consisted of fixation frequency, fixation durations, variability in fixation durations, saccade lengths, etc. *Local features* were sensitive to the words being read and included relationships between word length and fixation duration, number of words skipped, length of first pass fixations, etc. There were 17 global features, 12 local features, and 25 global-local features (two features were eliminated due to multicollinearity).

2.3 Supervised Classification

A host of 33 supervised machine learning algorithms from Weka [8] were used to build models to discriminate mind wandering (responding "yes" to the probe) from mindful reading (responding "no" to the probe). The 33 classifiers (with default parameters as specified in Weka) were run on the 3 feature types (global, local, global + local), 4 window sizes (3, 5, 10, and 15 seconds before probes), and on 4 configurations of the data (raw data, raw data with outliers removed, downsampled data with 50% "yes" and "no" responses, and downsampled data with outliers removed), yielding 1584 models in all. A leave-several-subjects-out validation method was employed in which data from a random 66% of the *subjects* were selected for the training set and the remaining 34% of subjects were used in the test set. This process was repeated for 25 iterations and classification performance was averaged across these iterations. The kappa metric was used to quantify classifier performance because it controls for random guessing.

3. RESULTS

The best results were obtained from the downsampled corpus without outlier removal. Mean kappas and standard deviations (across 25 iterations and shown in parentheses) for the best performing models for each feature type are shown in Table 1.

Table 1. Results for best performing models

Features	Kappa	RR.	Win	N	Classifier
Global (G)	.14 (.10)	56.2 (5.55)	5	374	Multiboost Adaboost
Local (L)	.08 (.06)	53.7 (4.21)	15	402	Naïve Bayes Updatable
Global + Local	**.23 (.08)**	60.0 (6.35)	5	374	Locally-weighted learning

Note. Standard deviation is in parentheses. RR. = recognition rate. Win = window size. N = number of instances.

The results support several conclusions on the feasibility of automatic detection of mind wandering by tracking eye gaze. First, although classification accuracies are moderate, the best performing models (Global + Local) detected mind wandering at rates significantly greater than chance. Second, the training and testing data were completely independent, so we have some confidence that the models generalize to new students. Third, the global models yielded higher accuracies than the local models. Fourth, a combination of global-local features resulted in a substantial improvement over the individual feature sets. Finally, gaze tracking over shorter window sizes (5 secs) was more effective than longer windows.

4. GENERAL DISCUSSION

Mind wandering is a ubiquitous phenomenon that has disastrous consequences for learning because it a quintessential signal of waning attention. We present a proof-of-concept of the possibility of automated tracking mind wandering during reading. Although we had some success in developing person-independent models to detect mind wandering, the accuracy of our models was moderate. We are currently in the process of refining our models by both increasing the size of the training data while simultaneously considering a larger feature space and more sophisticated classifiers. When coupled with the falling cost of eye trackers and the potential use of web-cams for low-cost gaze tracking, we expect that these improvements will yield sufficiently robust and scalable detectors of mind wandering. In turn, these detectors can be used to trigger interventions to restore engagement by reorienting attention to the task at hand.

5. ACKNOWLEDGMENTS

We gratefully acknowledge our collaborators at the University of Memphis. This research was supported by the National Science Foundation (NSF) (ITR 0325428, HCC 0834847, DRL 1235958). Any opinions, findings and conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of NSF.

6. REFERENCES

[1] J. Smallwood, et al., "When attention matters: The curious incident of the wandering mind," *Memory & Cognition*, vol. 36, no. 6, 2008, pp. 1144-1150; DOI 10.3758/mc.36.6.1144.

[2] J. Smallwood, et al., "Counting the cost of an absent mind: Mind wandering as an underrecognized influence on educational performance," *Psychonomic Bulletin & Review*, vol. 14, no. 2, 2007, pp. 230-236.

[3] S. Feng, et al., "Mindwandering while reading easy and difficult texts," *Psychonomic Bulletin & Review*, in press.

[4] J. Drummond and D. Litman, "In the Zone: Towards Detecting Student Zoning Out Using Supervised Machine Learning," *Intelligent Tutoring Systems, Part Ii*, Lecture Notes in Computer Science 6095, V. Aleven, et al., eds., Springer-Verlag, 2010, pp. 306-308.

[5] K. Rayner, "Eye movements in reading and information processing: 20 years of research," *Psychological Bulletin*, vol. 124, no. 3, 1998, pp. 372-422.

[6] E.D. Reichle, et al., "Eye movements during mindless reading," *Psychological Science*, vol. 21, no. 9, 2010, pp. 1300.

[7] D. Smilek, et al., "Out of Mind, Out of Sight: Eye Blinking as Indicator and Embodiment of Mind Wandering," *Psychological Science*, vol. 21, no. 6, 2010, pp. 786-789; DOI 10.1177/0956797610368063.

[8] M. Hall, et al., "The WEKA data mining software: an update," *ACM SIGKDD Explorations Newsletter*, vol. 11, no. 1, 2009, pp. 10-18.

DARE: Deep Anaphora Resolution in Dialogue based Intelligent Tutoring Systems

Nobal B. Niraula
Institute for Intelligent Systems
University of Memphis
nbnraula@memphis.edu

Vasile Rus
Institute for Intelligent Systems
University of Memphis
vrus@memphis.edu

Dan Stefanescu
Institute for Intelligent Systems
University of Memphis
dstfnscu@memphis.edu

ABSTRACT
Anaphora resolution is a central topic in dialogue and discourse processing that deals with finding the referents of pronouns. There are no studies, to the best of our knowledge, that focus on anaphora resolution in the context of tutorial dialogues. In this paper, we present the first version of DARE (Deep Anaphora Resolution Engine), an anaphora resolution engine for dialogue-based Intelligent Tutoring Systems. The development of DARE was guided by dialogues obtained from two dialogue-based computer tutors: DeepTutor and AutoTutor.

Keywords
Anaphora Resolution, Tutoring System, Dialogue Systems

1. INTRODUCTION
Anaphora resolution is the task of resolving what a pronoun or a referential noun phrase refers to. As an example consider the dialogue segment in Table 1 (a). Here, the pronoun "it" in the STUDENT turn refers to the first mention of the word "force" earlier in the turn.

In dialogue based Intelligent Tutoring Systems (ITSs), pronouns are quite frequent in students' natural language responses. Examples of student responses that contain anaphoric pronouns are shown in Table 1. These examples are from DeepTutor, a conversational ITS[1]. DeepTutor mimics the dialogue between a computer tutor and tutee and is based on constructivist theories of learning according to which students construct their knowledge themselves and only get help when floundering. The help consists of hints in the form of questions - see the DeepTutor dialogue turns in Table 1. Students responses are assessed for accuracy and appropriate feedback is provided by DeepTutor. Students can ask questions themselves as well.

Solving anaphors in student responses in dialogue-based ITSs is very important as it has a direct impact on assessing the correctness of student responses.

[1]www.deeptutor.org

Table 1: Use of pronouns in students' responses

(a) Intra-turn : **DEEPTUTOR**:What does Newton's second law say? **STUDENT**:for every force, there is another equal force to counteract it
(b) Inter-turn immediate: **DEEPTUTOR**:What can you say about the acceleration of the piano based on Newton's second law and the fact that the force of gravity acts on the piano? **STUDENT**: It remains constant.
(c) Inter-turn history: **DEEPTUTOR**: Since the ball's velocity is upward and its acceleration is downward, what is happening to the ball's velocity? **STUDENT**: increasing **DEEPTUTOR**: Can you please elaborate? **STUDENT**: it is increasing

While anaphora resolution is a well-studied problem in Natural Language Processing [1, 2], there is no previously reported work, to the best of our knowledge, which addresses the problem of anaphora resolution in dialogue based ITSs. As already mentioned, resolution of pronouns in ITSs is a key step towards understanding students' responses which impacts the accuracy of the student model. Failing to resolve pronouns can make the computer tutor assess incorrectly a student response and react ineffectively which would lead to suboptimal learning. Incorrect feedback from the system could frustrate students sometimes to the point of quitting interacting with the system.

Given the importance of accurate assessment of student responses in ITSs, a highly accurate anaphora resolution mechanism is needed. To this end, we have been developing Deep Anaphora Resolution Engine (DARE) for conversational ITSs. The design of DARE is guided by an analysis of actual interactions between students and two dialogue based ITSs : AutoTutor[2] and DeepTutor. DeepTutor is a fully online tutoring systems that has been used by close to a thousand students who can access DeepTutor anytime, anywhere. In its first version, DARE distinguishes two categories of pronouns in dialogue-based ITSs based on the lo-

[2]www.autotutor.org

cation of the referents: *intra-turn* anaphors and *inter-turn* anaphors. The *intra-turn* anaphors refer to an entity located in the current student dialogue turn. The *inter-turn* anaphors refer to entities in previous dialogue turns (or dialogue history) or entities present in the problem description or other contextual elements, e.g. even world knowledge. Examples of intra-turn and inter-turn anaphoric pronouns are shown in Table 1.

2. THE METHODOLOGY

As a starting point for developing DARE, we analyzed pronoun use in 24,945 student responses from DeepTutor log files and 1,978 student responses from AutoTutor logs. The results are shown in Figure 1. Both the students and computer tutor use first-person pronouns (i.e. "i","me", "we", "us") during the interaction. It should be noted that these pronouns do not need be resolved for assessment purposes (for space reasons we do not elaborate). Of the remaining pronouns, the top two most frequent pronouns, one of which is "it", account for more than 80% of the anaphors. Thus, considering a very few, very frequent anaphors may be a good start for developing DARE. Moreover, it was observed (see Section 3) that most pronouns used in student responses can be resolved withing the same responses or just looking at the previous system turn, i.e. the previous hint from the system. Although these aspects of anaphors in dialoge-based ITSs simplify the problem of pronoun resolution, the task is still challenging because "it" is often an pleonastic, i.e. "it" is not always an anaphoric pronoun [1].

Another important aspect of anaphors in tutorial dialogues is the location of pronouns in students' responses. In our data, we observed that most pronouns at the beginning of a student response refer to an antecedent present in the most recent system response, i.e. the previous dialogue turn. On the other hand, pronouns that occur in the middle or last part of a student response most likely refer to an entity in the current/same student response. Thus, we added in DARE a classifier that relies on the position of the anaphors to identify the text where the antecedent should be searched for. Given this fact, input to the DARE's resolution engine is the concatenation of previous tutor turn and current student response if the pronoun to be resolved occurs at the beginning of the student response and current student response only if the pronoun occurs in the middle or near the end of the student response. DARE then uses the coreference resolution module in the Stanford CoreNLP package in order to perform anaphora resolution. It should be noted that sometimes a pronoun in the student response may refer to entity in the problem description, e.g. the current Physics problems the student is working on. Also, it may be possible that a pronoun refers to something mentioned much earlier in the dialogue than the previous system turn. In the current version of DARE, we do not handle these latter cases. Another case we do not directly handle in DARE currently is the use of eliptic anaphors - see the first student response in (c) in Table 1 where instead of saying "it is increasing" the student simply says "increasing" (a typical exaple of ellipsis).

3. EXPERIMENTS AND RESULTS

In order to evaluate the performance of the DARE system, we extracted student-tutor interactions from the DeepTutor's log files. We only considered student-tutor interactions which contain at least one pronoun in the student responses. Since the pronouns in students' responses could refer to entities that were mentioned at any moment during the dialogue, i.e. the whole dialogue history, or the problem descriptions, we retain all this information for each instance in the data set. In total, we extracted 5,589 instances out of which 112 were annotated manually.

Figure 1: Pronouns used by (left) DeepTutor students in 24,945 dialogue turns (right) AutoTutor students in 1,978 dialogue turns

Pronoun	Count
it	3249
they	1152
i	526
you	207
her	163
she	158
he	90
them	89
we	81
her	62
us	56
me	48

Pronoun	Count
it	285
they	227
he	39
them	25
i	18
you	11
him	6
me	3
we	3
one	1
us	1
she	1

Antecedents in	Count(%)
H_0	85 (75.89%)
A	22 (19.64%)
H_1	5 (4.46%)
H_2	0 (0.00%)

Table 2: Location of antecedents for anaphors

The analysis of the annotated instances is shown in Table 2. Almost 76% of the time, students' anaphors refer to entities in the most recent tutor turn (hint/question H_0). They also use pronouns to refer to entities in their response (intra-turn anaphors) which accounts for almost 19% of the time. Next, they refer 4% of the time to entities in the hint that immediately precedes the most recent hint (i.e. H_1). Interestingly, there were no references to entities beyond H_1. Currently, DARE does not look back for referents beyond H_0. The accuracy of the current version of DARE is 46.36 %.

4. CONCLUSION AND FUTURE WORK

The current version of DARE presented here offers a good baseline which we plan to improve in future iterations of development. We also intend to evaluate the performance on a larger data set which we will make publicly available.

5. ACKNOWLEDGMENTS

This research was supported in part by Institute for Education Sciences under awards R305A100875.

6. REFERENCES

[1] R. Mitkov. Anaphora resolution: The state of the art. Technical report, University of Wolverhampton, 1999.

[2] A. Rahman and V. Ng. Supervised models for coreference resolution. In *Proceedings of EMNLP*, pages 968–977. ACL, 2009.

Are You Committed? Investigating Interactions among Reading Commitment, Natural Language Input, and Students' Learning Outcomes

Laura K. Varner
Tempe, AZ, USA
Arizona State University
Laura.Varner@asu.edu

G. Tanner Jackson
Tempe, AZ, USA
Arizona State University
Tanner.Jackson@asu.edu

Erica L. Snow
Tempe, AZ, USA
Arizona State University
Erica.L.Snow@asu.edu

Danielle S. McNamara
Tempe, AZ, USA
Arizona State University
Danielle.McNamara@asu.edu

ABSTRACT

The current study identifies relations among students' natural language input, individual differences in reading commitment, and learning gains in an intelligent tutoring system. Students (n = 84) interacted with iSTART across eight training sessions. Linguistic features of students' generated self-explanations (SEs) were analyzed using Coh-Metrix. Results indicated that linguistic properties of students' training SEs were predictive of learning gains, and that the strength and nature of these relations differed for students of low and high commitment to reading.

Keywords

Natural Language Processing, Learning, Intelligent Tutoring Systems, Reading Commitment

1. INTRODUCTION

Educational learning environments provide students with instruction intended to enhance particular knowledge, skills, and strategies in various domains. For example, iSTART is an intelligent tutoring system (ITS) that teaches students to use self-explanation (SE) reading strategies to comprehend challenging science texts. In this system, strategies are introduced and demonstrated to students. Then, students are offered the opportunity to practice applying the strategies they have learned to new texts. A natural language processing (NLP) algorithm assesses the quality of students' generated SEs and assigns scores (ranging from 0-3) and feedback to students during training [1].

Empirical studies have found that iSTART improves students' comprehension and strategy use over control groups at multiple education levels [2-3]. More recent work has investigated the impact of individual differences on students' learning gains in the system [4-5]. Jackson, Varner, Boonthum-Denecke, and McNamara (under review), for instance, found that iSTART helped students with low reading commitment to significantly improve their SE performance and ultimately match (or exceed) the performance of high commitment readers.

2. STUDY AND RESULTS

In the current study, we expand upon previous research to determine how individual differences in reading commitment impact students' natural language input and their subsequent learning gains. We employ NLP techniques to identify the linguistic properties of students' SEs that are predictive of learning gains. We then examine how these relations may differ for students with low and high prior reading commitment.

Participants were 84 high-school students randomly assigned to one of two versions of iSTART. Half (n = 43) of the students interacted with the original iSTART system and the other half (n = 41) interacted with a game-based version called iSTART-ME (motivationally enhanced) [6]. Both groups completed the same SE tasks and were assessed with the same algorithm; therefore, the conditions were collapsed for these analyses.

Students' learning gains were assessed using their SE scores (provided by the NLP algorithm) at pretest and posttest. To avoid biases associated with direct gain scores (low performing students have more room for improvement), a relative gain score was calculated. Relative gain scores represent students' improvement as a proportion of their possible improvement [(Posttest Proportion – Pretest Proportion) / (1 – Pretest Proportion)]. Additionally, students' reading commitment was assessed through demographic questions at pretest. Due to the limited scope available for this paper, the current analyses focus on a question that asked students to report the number of hours they spent reading for science courses; however, all reading commitment measures provided similar trends and results.

2.1 COH-METRIX ANALYSIS

Individual (sentence-level) SEs were combined for each text read and self-explained during training. This aggregation method is discussed in greater detail in previously published work [7]. Coh-Metrix [8] indices were calculated for each aggregated SE file. For each student, the mean values of 30 Coh-Metrix were calculated across texts to provide an average score for each linguistic measure. For more details on the linguistic measures presented here, please see [8].

2.2 ANALYSES

We investigated how linguistic properties of students' generated SEs predict relative learning gains. A stepwise regression analysis using each student's average Coh-Metrix scores as predictors of their relative gain scores yielded a significant model, F(2, 83) = 6.90, p = .002; R^2 = .15, retaining two predictors: *Third Person Pronoun Incidence* [β = -28, t(1, 82)= -2.81, p = .01] and *Average Sentence Length* [β = .26, t(1, 83)= 2.51, p = .01]. Results of this analysis indicate that when students used third person pronouns (e.g., he, she, it) and short sentences in their SEs, they were less likely to improve after training. At a global level, this analysis indicates that the use of objective and less elaborate language led to lower gains in the system.

Analyses further investigated how linguistic properties may be predictive of relative gain scores as a function of students' prior commitment to reading. A median split on the pretest reading commitment measure (i.e., hours spent reading for sciences courses) was used to categorize students as having either low (n = 47) or high (n = 37) reading commitment.

A stepwise regression analysis using the average Coh-Metrix scores for *low reading commitment students* as predictors of their relative gain scores yielded a significant model with one predictor, F(1, 46) = 6.33, p = .02; R^2 = .12 (see Table 1). This result indicates that students with low reading commitment who repeated concepts across SEs tended to gain more from training.

Table 1. Stepwise Regression Analyses for Low and High Commitment Students

Linguistic Indices	*β*	ΔR^2
Low Reading Commitment		**.12***
Noun Overlap	.35	
High Reading Commitment		**.56****
Third Person Pronouns	-.45	.19*
Incidence of Infinitives	.47	.10*
Average Polysemy	-.45	.09*
Casual Ratio	.53	.11*
Incidence of Negations	-.32	.07*

p<.05 *, *p*<.001 **

A stepwise regression using the average Coh-Metrix scores for *high reading commitment students* as predictors of relative gain scores yielded a significant model with five predictors, F(5, 36) = 7.77, p < .001; R^2 = .56 (see Table 1). This analysis indicated that the linguistic properties of training SEs accounted for over half of the variance in high reading commitment students' learning gains. Most significantly, high reading commitment students benefitted most when they used less objective language within their SEs.

3. DISCUSSION

This study investigated relations among students' prior commitment to reading, linguistic properties of their generated SEs, and relative learning gains in the iSTART system. Results indicated that the relations between the linguistic features of students' SEs and relative learning gains varied when accounting for students' reading commitment. In particular, one cohesion variable accounted for 12% of the variance in low reading commitment students' relative learning gains, whereas five predictors combined to account for over 50% of the variance in the relative learning gains of highly committed students.[1] As mentioned previously, analyses with other reading commitment measures produced similar results.

[1] Separate analyses confirmed that these results could not be accounted for by individual differences, such as reading ability, as these measures were not predictive of relative learning gains.

This work expands upon previous research, relating features of natural language input to the level of students' cognitive processing of text [9]. The current analyses leverage this prior work to investigate how linguistic differences between groups of students shed light on their potential to gain from training. These results can help researchers gain a better understanding of the learning processes used by different student users, as well as the complex interactions between individual students and learning systems.

4. ACKNOWLEDGMENTS

This research was supported in part by: IES R305G020018-02, IES R305G040046, IES R305A080589, and NSF REC0241144, NSF IIS-0735682. Opinions, conclusions, or recommendations do not necessarily reflect the views of the IES or NSF.

5. REFERENCES

[1] McNamara, D., Boonthum, C., Levinstein, I., Millis, K.: Evaluating Self-explanations in iSTART: Comparing Word-based and LSA Algorithms. In T. Landauer, D. McNamara, S. Dennis, W. Kintsch (eds.) Handbook of Latent Semantic Analysis, pp. 227-241. Mahwah Erlbaum (2007)

[2] Magliano, J., Todar, S., Millis, K., Wiemer-Hastings, K., Kim, H., McNamara, D.: Changes in Reading Strategies as a Function of Reading Training: A Comparison of Live and Computerized Training. Journal of Educational Computing Research 32, 185-208 (2005)

[3] O'Reilly, T., Best, R., McNamara, D.: Self-explanation Reading Training: Effects for Low-knowledge Readers. In: Proceedings of the 26th Annual Conference of the Cognitive Science Society pp. 1053-1058. Erlbaum Portland, OR (2004)

[4] Jackson, G.T., Boonthum, C., McNamara, D.S.: The Efficacy of iSTART Extended Practice: Low Ability Students Catch Up. In: The Proceedings of Intelligent Tutoring Systems pp. 349-351. Springer Berlin/Heidelberg, Pittsburg, PA (2010)

[5] Jackson, G. T., Varner, L. K., Boonthum-Denecke, C., McNamara, D. S.: The Impact of Individual Differences on Learning with an Educational Game and a Traditional ITS. Manuscript submitted to the International Journal of Learning Technology (under review)

[6] Jackson, G., McNamara, D.: Motivation and Performance in a Game- based Intelligent Tutoring System. Journal of Educational Psychology, (in press)

[7] Varner, L. K., Jackson, G. T., Snow, E. L., McNamara, D. S.: Does size matter? Investigating user input at a larger bandwidth. In: Proceedings of the 26th Annual Florida Artificial Intelligence Research Society Conference, AAAI, St. Petersburg, FL (in press)

[8] Graesser, A.C., McNamara, D.S., Louwerse, M., Cai, Z.: Coh-Metrix: Analysis of Text on Cohesion and Language. Behavior Research Methods 36, 193-202 (2004)

[9] Jackson, G.T., Guess, R.H., McNamara, D.S.: Assessing Cognitively Complex Strategy Use in an Untrained Domain. Topics in Cognitive Science 2, 127-137 (2010)

Using Multi-level Models to Assess Data From an Intelligent Tutoring System

Jennifer L. Weston
Department of Psychology and
Learning Sciences Institute
Arizona State University
Tempe, AZ
Jen.weston@asu.edu

Danielle S. McNamara
Department of Psychology and
Learning Sciences Institute
Arizona State University
Tempe, AZ
Danielle.mcnamara@asu.edu

ABSTRACT

Intelligent tutoring systems yield data with many properties that render it potentially ideal to examine using multi-level models (MLM). Repeated observations with dependencies may be optimally examined using MLM because it can account for deviations from normality. This paper examines the applicability of MLM to data from the intelligent tutoring system Writing-Pal using intraclass correlations. Further analyses were completed to assess the impact of individual differences on daily essay scores along with the differential impact of daily vs. mean attitudinal ratings.

Keywords

Multi-Level models, Writing, Intelligent Tutoring Systems

1. INTRODUCTION

With the advent of intelligent tutoring systems (ITSs), the amount and complexity of data available to researchers has increased exponentially. ITSs provide the opportunity for repeated administration of assessments and, in some cases, ease of scoring that data. Though most tutoring systems provide multiple assessments of student progress (i.e., multiple text responses or worked problems), many researchers assess performance using pretest-posttest differences or repeated measures analyses, potentially missing out on rich data collected between these two end points.

When a student produces multiple responses, dependency arises in the data, thus violating central assumptions underlying both regression and ANOVA. Dependency, measured using intraclass correlations (ICC), is a pervasive problem in educational data, ranging from less problematic (a group of students within schools) to highly problematic (observations within individuals) [1]. Even when 5% of the variation in a data set is due to nested structure, (i.e.; dependency) it is advisable to assess differences at the highest cluster level.

The Writing Pal (W-Pal, [2]) is an ITS that provides writing strategy instruction to high school and entering college students. This system teaches writing strategies that encompass the entire writing process from prewriting through revision. Students have the opportunity to watch lesson videos, practice individual strategies within educational mini-games, and write and receive feedback on timed, prompt-based (SAT-style) essays.

In addition to providing instruction, W-Pal affords students the opportunity to practice writing and receive feedback on their essays. Students write prompt-based persuasive essays within an essay writing module. Essays are scored using an algorithm trained on a large corpus of SAT-style essays [3]. In this paper, we evaluate the applicability of multi-level modeling (MLM) for ITS data. Specifically, we examine the level and impact of dependency in the data. We examine a means-as-outcomes model assessing the impact of individual differences on daily essay scores. In addition, we examine a contextual effects model that assesses the differential impact of daily and mean ratings of attitudinal measures.

2. METHODS

Sixty-five high school students from a large urban southwestern city participated for payment in a lab based study to assess the effectiveness of W-Pal. All participants were recruited from the community. The study compared two versions of the W-Pal system: the full W-Pal system, and a version including only Essay Practice. In the W-Pal condition, students had access to the entire W-Pal system, whereas those in the Essay Practice condition only interacted with the essay practice function. These conditions were designed to control for time-on-task.

This study consisted of 10 sessions along with a home survey, which participants completed prior to attending their sessions. The home survey included basic demographics and measures of writing habits. The first session was a pretest session during which participants completed a pretest essay and prior knowledge assessments.

Participants in all conditions began sessions 2-9 by filling out a survey about their previous session and current mood, and then completed a SAT-style practice essay. Based on students' randomly assigned condition, some students interacted with all of W-Pal (n= 33), while others interacted with the Essay Practice module in W-Pal (n=32). Participants were given a maximum of 25-minutes to complete their essay. They then received feedback and were given an additional 10-minutes to revise their essays. Students in the W-Pal condition then completed an assigned lesson and game based practice. Students in the Essay condition completed a second SAT-style essay, also revising this essay.

During the final session, students completed a posttest, which was the same for all participants regardless of condition. For the current paper, only the essay scores, pretest, and attitudinal measures will be considered.

2.1 Measures

2.1.1 Essays

Depending on condition, participants wrote either 8 or 16 practice essays with feedback, ,and a pretest and posttest essay without feedback. The essay prompts were adapted from SAT writing assessments and scored on a 1-6 scale using the W-Pal algorithm validated by Crossley and colleagues [3]. This algorithm displays sufficient accuracy (exact agreement of 55% and adjacent agreement of 92%). The present analyses focus on the eight practice essays with common prompts for both conditions (i.e., the first essay written in the Essay condition). The pretest essay provides a measure of prior writing ability.

2.1.2 Individual Difference Measures

A variety of individual difference measures were administered to assess the impact of these characteristics on essay quality. In the present study, we focus on the measures of self-efficacy, prior reading ability, and motivation. Self-efficacy was measured using the Writing Attitudes and Strategies Self-Report Inventory (WASSI, [4]). Prior reading ability was assessed using the Gates MacGinitie Reading Test (GMRT Ed.3; level 10/12, form S). Motivation was measured using a daily and posttest survey with questions about participants' moods and previous and anticipated interactions with W-Pal.

3. RESULTS

3.1 Applicability of Multilevel Models

A series of unconditional models for all level-1 variables were estimated. The variance estimates from these analyses were used to compute intraclass correlations (ICCs). The ICC for daily essay scores was ICC =.47, suggesting that 47% of the variance in essay score can be attributed to the individual. For daily survey items, these values ranged from .37 - .98, suggesting that a significant portion of the variance for all of the daily survey items can be attributed to the individual.

3.2 Means-as-Outcomes Model

We estimated a means-as-outcomes model in which we used a number of level-2 variables to predict daily essay score. Variables were selected based on prior research on writing and included prior writing ability, reading ability (GMRT), writing self-efficacy, and condition. This model assesses the impact of each prior ability measure on average daily essay score holding all others constant.

A likelihood-ratio test was completed to assess the explanatory power of the level-2 variables. The Likelihood-ratio test was significant $\chi^2(4) = 46.21$, $p < .001$, suggesting that the MLM is superior to a model not containing these variables. The Bayesian Information Criterion (BIC) was also examined. The results from the BIC values mirrored the results found using the likelihood ratio test [5] Additionally, the inclusion of these five variables reduced the between cluster variation by 63%. All predictors had a significant impact on daily essay scores (prior writing ability B = .211, Prior Reading Ability B = .034, Self-Efficacy B = .023, and Condition B = .014)

3.3 Contextual Effects Model

An additional model was estimated using the daily survey data to predict daily essay scores. To investigate the possibility of contextual effects (differential effects at level-1 and level-2), we also included the cluster (person) means as level-2 predictors.

A Wald test of the 10 level-2 coefficients was statistically significant, F(10, 23) = 23.943, p =.007, indicating that the set of contextual effects improved the fit of the model. Further univariate tests indicated that competitiveness ($\gamma1$), feelings of frustration ($\gamma2$), and self-assessments of improvement ($\gamma3$) exerted significant contextual effects, $\gamma1 = .102$, $p = .003$; $\gamma2 = -.070$, $p = .020$; $\gamma3 = -.261$, $p = .049$; the contextual effect for mood ($\gamma4$) was marginally significant, $\gamma4 = .373$, $p = .061$. The signs and magnitude of the level-2 regressions (daily survey means predicting daily essay mean) were stronger than the level-1 predictors; however, the effects of sustained levels of certain feelings about the system (e.g., frustration) seemed to be more complex, warranting further investigation.

4. DISCUSSION

The data examined in this study exhibit high levels of dependency, rendering it ideal for multi-level modeling. The ICC values for the repeated assessments in W-Pal range from .37 - .98, exceeding appropriate values for using regression and analysis of variance. By using a means-as-outcomes model, we were able to account for 63% of the variance due to the cluster (student). The results suggest that there is an advantage for those interacting with the complete W-Pal system, additionally, individual differences were important predictors of average daily essay score.

The analysis using the contextual effects model showed that, for this data, daily and mean values for attitudinal survey items had differential effects on essay scores. For instance, while daily enjoyment has a negative relationship with daily essay score, the participant's average level of enjoyment had a positive relationship with average essay scores.

Further work will be completed to combine these models and to investigate the utility of using random slopes for the level-1 variables. Interactions will also be investigated further. Overall, the data from W-Pal is ideal for using MLM for assessment.

5. ACKNOWLEDGMENTS

The research reported here was supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305A080589 to Arizona State University.

6. REFERENCES

[1] Hedges, L. V., and Hedberg, E. C.(2007). Intraclass correlation values for planning group randomized trials in education. *Educational Evaluation and Policy Analysis.* 29, 1. 60-87.

[2] Roscoe, R. D., and McNamara, D.S. (in press). Writing pal: intelligent tutoring of writing strategies in the high school classroom, *Journal of Educational Psychology.*

[3] Crossley, S. A., Roscoe, R., and McNamara, D. S. (in press). Predicting human scores of essay quality using computational indices of linguistic and textual features. Proceedings of the 15th International Conference on Artificial Intelligence in Education. Auckland, New Zealand: AIED.

[4] Weston, J. L., Roscoe, R., Floyd, R. G., and McNamara, D. S. (2013, May). The WASSI (Writing Attitudes and Strategies Self-Report Inventory): Reliability and validity of a new self-report writing inventory. Poster presented at the 2013 Annual Meeting of the American Educational Research Association, San Francisco, CA

[5] Posada, D., & Buckley, T. R. (2004). Model selection and model averaging in phylogenetics: Advantages of Akaike information criterion and Bayesian approaches over likelihood ratio tests. *Systemic Biology,53*, 5. 793-808.

Oral Presentations (Young Researchers Track)

Evaluation of Automatically Generated Hint Feedback

Michael John Eagle
North Carolina State University
Department of Computer Science
890 Oval Drive, Campus Box 8206
Raleigh, NC 27695-8206
mjeagle@ncsu.edu

Tiffany Barnes
North Carolina State University
Department of Computer Science
890 Oval Drive, Campus Box 8206
Raleigh, NC 27695-8206
tiffany.barnes@ncsu.edu

ABSTRACT

This work explores the effects of using automatically generated hints in problem solving tutor environments. Generating hints automatically removes a large amount of development time for new tutors, and it also useful for already existing computer-aided instruction systems that lack intelligent feedback. We focus on a series of problems, after which, previous analysis showed the control group is to be 3.5 times more likely to cease logging onto an online tutor when compared to the group who were given hints. We found a consistent trend in which students without hints spent more time on problems when compared to students that were provided hints.

1. INTRODUCTION

Problem solving is an important skill across many fields, including science, technology, engineering, and math (STEM). Working open-ended problems may encourage learning in higher 'levels' of cognitive domains [1]. Intelligent tutors have been shown to be as effective as human tutors in supporting learning in many domains, in part because of their individualized, immediate feedback, enabled by expert systems that diagnose student's knowledge states [9]. However, it can be difficult to build intelligent support for students in open problem-solving environments. Intelligent tutors require content experts and pedagogical experts to work with tutor developers to identify the skills students are applying and the associated feedback to deliver [6].

Barnes and Stamper built an approach called the Hint Factory to use student data to build a graph of student problem-solving approaches that serves as a domain model for automatic hint generation [7]. Hint factory has been applied across domains [5]. Stamper et al. found that the odds of a student in the control group dropping out of the tutor were 3.5 times more likely when compared to the group provided with automatically generated hints [8]. The hints also affected problem completion rates, with the number of problems completed in L1 being significantly higher for the hint group by half of a standard deviation, when compared to the control group.

This work extends these results by exploring potential causes for these differences. We hypothesized that there would be differences in the amount of time required to solve problems between the students who received hints and the students who did not. We concentrated on the first five problems, before the dropout differences reported in [8]. We found that while there are no differences in total time in tutor, there were some differences in several problems where students in the control group spent significantly more time attempting to solve the problems when compared to the group of students who were provided with hints. This suggests that while both groups spend similar amount of total time in tutor, the group provided with automatically generated hints was able to get further in the tutor. Exploration of the interaction networks [4] for these problems revealed that the control group often spent this extra time pursuing buggy-strategies that did not lead to solutions.

2. THE DEEP THOUGHT TUTOR

We perform our analysis on data from the Deep Thought propositional logic tutor [2]. Each problem provides the student with a set of premises, and a conclusion, and asks students to prove the conclusion by applying logic axioms to the premises. Deep Thought allows students to work both forward and backwards to solve logic problems [3]. Working backwards allows a student to propose ways the conclusion could be reached. For example, given the conclusion B, the student could propose that B was derived using Modus Ponens (MP) on two new, unjustified propositions: $A \rightarrow B, A$. This is like a conditional proof in that, if the student can justify $A \rightarrow B$ and A, then the proof is solved. At any time, the student can work backwards from any unjustified components, or forwards from any derived statements or the premises.

2.1 Data

We perform our experiments on the Spring and Fall 2009 Deep Thought logic tutor dataset as analyzed by Stamper, Eagle, and Barnes in 2011[8]. In this dataset, three different professors taught two semesters each of an introduction to logic course, with each professor teaching one class with hints available and one without hints in the Deep Thought tutor. In the spring semester there were 82 students in the Hint group and 37 students in the Control group; in the fall semester there were 39 students in the Hint group and 83 in the Control group. Students for which application log-data did not exist were dropped from the study; resulting in 68 and 37 students in the Hint group, and 28 and 70 students in the Control group for the first and second semesters respectively. This results in a total of 105 students in the Hint group and 98 students in the Control group. Students from the 6 sections of an introduction to logic course were assigned 13 logic proofs in the deep thought tutor. The problems are organized into three constructs: level one (L1) consisting of the first six problems assigned; level two (L2) consisting of the next five problems assigned; and level three (L3) consisting of the last two problems assigned. We refer to the group that received hints as the Hint group, and the group that did not receive hints as the Control group.

3. RESULTS

In order to investigate the increased rate of drop-out between the hint group and the control group. We concentrate on the first 5 problems from L1 of the Deep Thought Tutor. We focus here as, while the groups started with similar completion and attempt rates, after level one the groups diverge on both completion and problem attempt rates. Since investigation of the interaction networks for these problems revealed that the control group often pursue buggy-strategies, which do not result in solving the problem, we hypothesized that their would be differences in the amount of time spent in tutor between the groups.

We performed analysis on the student-tutor interaction logs. For each student we calculated the summation of their elapsed time per interaction. To control for interactions in which the student may have idled we filtered any interactions that took longer than then minutes. The descriptive statistics for this are located in Table 1, Prob represents the problem number, H and C represent the Hint group and the Control group.

Table 1: Descriptive Statistics for Time (in seconds) Spent in Each Problem

	N		M		SD	
Prob	H	C	H	C	H	C
1.1	104	93	765.89	1245.24	956.41	1614.30
1.2	88	76	761.65	1114.37	911.24	1526.91
1.3	90	67	664.17	1086.09	733.95	2119.19
1.4	87	71	754.60	1266.39	1217.06	1808.53
1.5	84	67	710.62	1423.22	1192.43	2746.54

The large standard deviations are a sign that perhaps this data is not normal. Exploring the data with Q-Q plots reveals that the data is in fact, not normally distributed. This prevents us from performing between-group statistical tests, such as the student's t-test, as our data violates the assumption of normality. To normalize the data, we use a logarithmic transformation (common log) to make the data more symmetric and homoscedastic. Observation of the Q-Q plot and histogram of the transformed data reveal that we had addressed the normality concerns. The results are presented in Table 2.

To test for differences between the two groups on each problem, we subjected the common log transformed data to t-test. The results from this test are presented in Table 3. There are significant differences for problems one, four, and five. The ratio is calculated by taking the difference between the hint group mean and the control group mean. As $\lg(x) - \lg(y) = \lg(\frac{x}{y})$ the confidence interval from the logged data estimates the difference between the population means of log transformed data. Therefore, the anti-logarithms of the confidence interval provide the confidence interval for the ratio. We provide the C:H ratios and confidence intervals in Table 4.

Table 2: Descriptive Statistics After Common Log Transformation

	N		M		SD	
Prob	H	C	H	C	H	C
1.1	104	93	2.63	2.79	0.48	0.55
1.2	88	76	2.59	2.73	0.54	0.54
1.3	90	67	2.62	2.72	0.44	0.48
1.4	87	71	2.66	2.89	0.40	0.41
1.5	84	67	2.55	2.75	0.48	0.60

Table 3: Ratio Between Groups (H:C) in the Original Scale

		95% Confidence Interval			
Prob	Ratio	low	high	p-value	t
1.1	0.69	0.50	0.97	0.03	-2.18
1.2	0.72	0.49	1.06	0.10	-1.68
1.3	0.78	0.56	1.10	0.15	-1.43
1.4	0.58	0.44	0.78	0.00	-3.61
1.5	0.62	0.42	0.93	0.02	-2.31

Table 4: Ratio Between Groups (C:H) in the Original Scale

		95% CI	
Prob	Ratio	low	high
1.1	1.44	1.04	2.01
1.2	1.39	0.94	2.05
1.3	1.27	0.91	1.78
1.4	1.71	1.28	2.30
1.5	1.60	1.07	2.40

In order to explore what these differences mean, we shall transform the data back to our original scale (seconds.) The transformed data is provided in Table 5. These are the Geometric Means, which are often closer to the original median, than they are the mean. The ratios from Tables 3 and 4 are easily interpreted as the log of the ratio of the geometric means. For example in problem 1.4, in the common log scale, the mean difference between hint and control group is -0.23. Therefore, our best estimate of the ratio of the hint time and control time is $10^{-.23} = 0.58$. Our best estimate of the effect of Hint is it takes 0.58 times as many seconds as the control group to complete the problem. The confidence interval reported above is for this difference ratio.

The geometric mean of the amount of seconds needed to solve problem four for the hint group is 0.58 (95% CI: 0.44 to 0.78) times as much as that needed for students in the control group. Stated alternatively, students in the control group spend 1.71 (95% CI: 1.07 to 2.40) times as long as the Hint group in problem four.

Table 5: Geometric Means and Confidence Intervals in Seconds

		95% CI			95% CI	
P	H	low	high	C	low	high
1	428.66	347.14	529.31	618.19	478.60	798.51
2	387.07	297.97	502.82	537.80	405.75	712.82
3	413.80	335.89	509.78	527.18	405.05	686.13
4	454.43	374.38	551.61	778.01	624.48	969.29
5	352.90	278.06	447.89	565.61	405.34	789.24

Exploring the total time spent between all five problems also required a log transformation. The total time spent on the first 5 problems between the hint group ($M = 3.34, SD = 0.4$) and the control group ($M = 3.44, SD = 0.51$) was not significant, $t(198) = 1.41, p = 0.16$. This corresponds to a H:C ratio of 0.81 (95% CI: 0.60 to 1.09), and a C:H ratio of 1.24 (95% CI: 0.92 to 1.66).

4. DISCUSSION

The results of this analysis show that students in the control group are overall not spending significantly more time in the tutor during these first five problems. However, the control does spend significantly more time in some problems compared to the hint group. Problems one, three and four provided students with the automatically generated hints. While problem two and five had no hints for either group. We would expect there to be differences in time to solve for the hint group, and this was the case for problem one. We would also expect that having no hints on problem two would not display an effect, as the second problem is too early to expect differences to emerge between the groups. Problem three is interesting as this problem is the first in which the groups begin to show preferences towards different solution strategies. With the control group preferring to work backwards, and the hint group preferring to work forwards (hints are only available for solutions working forward). Problem four and five, both of which showed significant differences in time spent, showed a large portion of control group student interactions to be perusing buggy-strategies.

This is interesting as the control group is spending at least as much, and often more, time in tutor and yet meeting with less overall success. The control students are not becoming stuck in a single bottleneck location within the problems and then quitting, which would result in lower control group times. The control students are actively trying to solve the problems using strategies that do not work. The hint group is able to avoid these strategies via the use of the hints. The hint group students also develop a preference for solving problems forward, as that is the direction in which they can ask for hints. It is interesting to see that these preferences remain, even when hints are not available.

The effect of the automatically generated hints appear to let the hint group spend around 60% of the time per problem compared to the control group. Or stated differently, the control group requires about 1.5 times as much time per problem when compared to the hint group. These results show that the hints provided by the Hint Factory, which are generated automatically, can provide large differences in how long students need to solve problems.

5. CONCLUSIONS AND FUTURE WORK

This paper has provided evidence that automatically produced hints can have drastic effects on the amount of time that students spend solving problems in a tutor. We found a consistent trend in which students without hints spent more time on problems when compared to students that were provided hints. Exploration of the interaction networks for these problems revealed that the control group often spent this extra time pursuing buggy-strategies that did not lead to solutions. Future work will explore other data available on the interaction level, such as errors, in order to get a better understanding of what the control group is doing with their extra time in tutor. We will also look into the development of further interventions that can help students avoid spending time on strategies that are unlikely to provide solutions.

6. REFERENCES

[1] B. S. Bloom. *Taxonomy of Educational Objectives: The Classification of Educational Goals*. Taxonomy of educational objectives: the classification of educational goals. Longman Group, New York, 1956.

[2] M. J. Croy. Graphic interface design and deductive proof construction. *J. Comput. Math. Sci. Teach.*, 18:371–385, December 1999.

[3] M. J. Croy. Problem solving, working backwards, and graphic proof representation. *Teaching Philosophy*, 23:169–188, 2000.

[4] M. Eagle, M. Johnson, and T. Barnes. Interaction Networks: Generating High Level Hints Based on Network Community Clustering. *educationaldatamining.org*, pages 1–4.

[5] D. Fossati, B. Di Eugenio, S. Ohlsson, C. Brown, L. Chen, and D. Cosejo. I learn from you, you learn from me: How to make ilist learn from students. In *Proceedings of the 2009 conference on Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling*, pages 491–498, Amsterdam, The Netherlands, The Netherlands, 2009. IOS Press.

[6] T. Murray. Authoring intelligent tutoring systems: An analysis of the state of the art. *International Journal of Artificial Intelligence in Education (IJAIED)*, 10:98–129, 1999.

[7] J. Stamper, T. Barnes, L. Lehmann, and M. Croy. A pilot study on logic proof tutoring using hints generated from historical student data. *Proceedings of the 1st International Conference on Educational Data Mining (EDM 2008)*, pages 197–201, 2008.

[8] J. C. Stamper, M. Eagle, T. Barnes, and M. Croy. Experimental evaluation of automatic hint generation for a logic tutor. In *Proceedings of the 15th international conference on Artificial intelligence in education*, AIED'11, pages 345–352, Berlin, Heidelberg, 2011. Springer-Verlag.

[9] K. VanLehn. The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*, 46(4):197–221, 2011.

Analysing Engineering Expertise of High School Students Using Eye Tracking and Multimodal Learning Analytics

July Silveira Gomes, Mohamed Yassine, Marcelo Worsley, Paulo Blikstein

Stanford University

520 Galvez Mall, room 102

Stanford, California, 94305

{julys, myassine, mworsley, paulob}@stanford.ed

ABSTRACT

In this paper, we describe results of a multimodal learning analytics pilot study designed to understand the differences in eye tracking patterns found to exist between students with low and high performance in three engineering-related computer games, all of which require spatial ability, problem-solving skills, and a capacity to interpret visual imagery. In the first game, gears and chains had to be properly connected so that all gears depicted on the screen would spin simultaneously. In the second game, students needed to manipulate lines so as to ensure that no two intersected. In the final game, students were asked to position gears in specific screen locations in order to put in motion on-screen objects. The literature establishes that such abilities are related to math learning and math performance. In this regard, we believe that understanding these differences in student's visual processing, problem-solving, and the attention they dedicate to spatial stimuli will be helpful in making positive interventions in STEM education for diverse populations.

Categories and Subject Descriptors

H.5.2 [Information Interfaces and Presentation]: User Interfaces. K.3.1 [Computers and Education]: Computer Uses in Education.

Keywords

Eye tracking, simulations, games, multimodal learning analytics, constructionism, spatial ability.

1. INTRODUCTION

The need to engage and motivate more students to learn science and engineering has raise considerable awareness about Constructionist [9] and project-based pedagogies in classrooms. Understanding students' behaviors and cognitive evolution in these open-ended environments is a challenge that is being tackled in the nascent field of Multimodal Learning Analytics [4, 11]. In particular, this study uses eye tracking to examine students' capacity to interpret visual imagery in the context of engineering problem solving.

Computer-based learning tools such as games and simulations have become pervasive in learning environments. These technologies can be used by learners to improve their cognitive abilities and to acquire specific skills [6], including those involving visuospatial attention and perception [1]. Video and computer games habits have been shown to be related to the improvement of visuospatial abilities, including mental rotation and visual memory. Likewise, the enhancement of performance in visual memory recall tasks has been associated with the duration of game exposure, even when the gender has been controlled [10].

In this paper, our interest is not in the games themselves but in the engineering, mathematical, and problem solving skills required to solve the puzzles presented in the game. Visuospatial abilities are involved in the processes of manipulating spatial forms, and these abilities are associated with different kinds of scientific thinking [12]. Performance in standardized visuospatial tasks has been associated with performance on math evaluation tests as early as primary school [5]. A study with low- and typically achieving students demonstrated that low achievers have poorer overall performance and a higher number of errors in online game-like visuospatial working memory tasks. The same study found that low achievers also demonstrated more errors and higher reaction times for arithmetic tasks [2].

Another study showed that difficulty in manipulating internal and external visuospatial representations are related to conceptual errors in chemistry, even when the problem to be solved is not explicitly spatial. These authors suggest that designing and developing tools and software to train students' spatial visualization capacities may improve their representational and conceptual skills, which should be helpful for learning chemistry. The principles involved in this process include: 1) the provision of multiple representations of the process; 2) ensuring that referential connections between the conceptual elements of the lesson are easily grasped through visual representations; 3) the presentation of the dynamic and the interactive nature of the process; 4) promoting transformations between 2D and 3D representation; and 5) reducing students' cognitive load by making the information more explicit and integrated [12]. Another researcher suggests that manual rotation is also useful to the improvement of the mental rotation skills [11].

Considering these five principles, the present effort presents data from a pilot study (n=7), where students played three online games requiring visuospatial ability in order to explore individual gaze characteristics found to be related to performance.

2. EXPERIMENTAL DESIGN

Seven high school students were invited to play three online games in two separate sessions six days apart. During the first session, they played "Wheels," a game that required them to connect gears and chains until all gears were spinning (Figure 1), as well as "Lines," a game in which they were required to uncross lines until no intersecting lines remained. During the second session, they played "Gears," a second gears game, the object of which was to place gears in specific locations on the screen to set on-screen objects in motion, and they also played the Lines game from the previous session. In each session, they had 5 minutes to play Wheels and Gears and 4 minutes to play Lines.

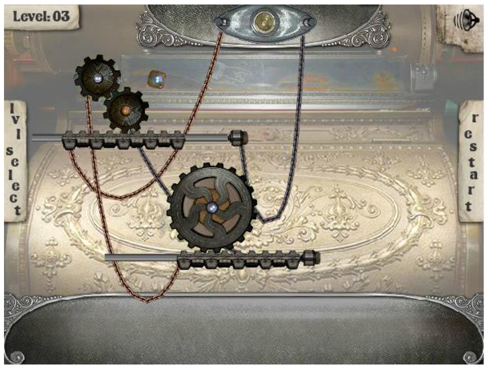


Figure 1: Screenshot of the "Wheels" game.

3. METHODOLOGICAL APPROACH

Since each game was divided into separate levels of increasing complexity, but of brief duration, where only one challenge had to be solved, it seemed appropriate to group the collected data by level in order to gain insight into the strategy used by the learners at each level, as well as the evolution of their strategy during their advancement through the game.

To achieve this degree of differentiation, we wrote a Python script to compute from the collected data the variables that describe a learner's eye patterns at each game level. The variables per student included the number of mouse clicks per level, the time spent on each level, the number and duration per level of unique gaze points, or eye fixations, the direction of the saccadic eye movements (i.e. subjects are moving their eyes from left to right, top to bottom, or any combination thereof), and the type of eye pattern for every trigram, or sequence of three gaze points. Examples of eye patterns of a trigram include when the subject looks right, and then left, and then right again; or when the subject looks up, and then down, and then up again. In the literature such movements are described as A-B-A patterns [10].

To identify differences in eye patterns among the students, a k-means clustering algorithm was performed on the data with variable *k* values.

4. RESULTS

The sample was composed of 4 males and 3 females, and all of them played the games on the same two days. The average level reached during the first session was 5.71 (1.11) for Wheels and 3.71 (0.49) for Lines. The average level during the second session was 8.8 (2.1) for Gears and 3.6 (0.84) for Lines. Since the students had a limited time to play, they were given the option to stop playing at any time for any reason. When they skipped the game, they were led to the next game. Only one student stopped the games prior to completion, and he did so in all of the games. During the first session, he stopped playing "Wheels" at level 4 and "Lines" at level 3. In the second day, he skipped "Gears" at level 9 and "Lines" at level 3.

After performing clustering on the data, we obtained 3 clusters (*k*=3): the first cluster had 2 students, the second, 3 students, and 2 students in the third. The first cluster is composed of the students with the best performance, based on levels reached. This cluster differed significantly from clusters 2 and 3, especially for Wheels (z = 1.15, -0.64, -0.19, respectively) and Gears (z = 0.47, 0.38 and -1.04, respectively). However, the differences were less marked for Lines. During the first session of Lines, there was no difference in performance between clusters 0 and 2 (z = 0.59), although a difference was observed for cluster 1 (z = -0.78). In the second session, no difference was found between the clusters for the lines game.

Taking this in account, we performed analysis on the last game level all students reached on the Wheels (level 4) and Gears (level 6) games to determine whether specific patterns of eye movement at these levels might correlate with overall performance levels. To proceed with the analysis, a new variable defining groups 1, 2 and 3 (corresponding to clusters 1, 2 and 3), was set on the eye tracker software. The goal of this procedure was to identify the visual areas of interest for each group. Figure 2 shows the gaze point clusters for level 4 of the "Wheels" game, which represent the different regions of the screen where the students' vision focused.

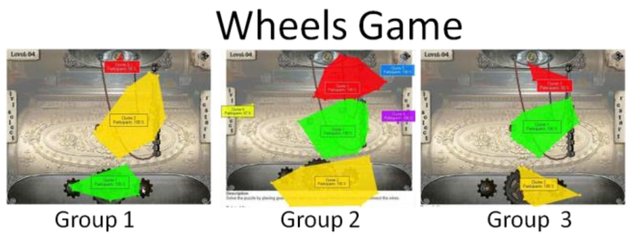


Figure 2: Gaze point clusters for groups 1, 2, and 3 for level 6 of the Wheels game. Each color represents one gaze point cluster: green cluster 1, yellow cluster 2 and red cluster 3.

We can observe a spatial difference between the clusters of group 1, which is the group with the best performance on the task, compared with groups 2 and 3, especially with regard to the screen positions for the first cluster. The first group looked first to the bottom of the screen, where the different gear options were available. The time of the first fixation was 0.52s for group 1, which is significantly shorter than the 6.53s and 1.26s for groups 2 and 3, respectively. Group 1 also used fewer clicks (z= -0.82, 0.17, and 0.57, respectively), more unique fixation points (z= 0.7, 0.34, and 0.19, respectively) and a longer duration on average for each eye fixation (z= 1.05, -0.55, and -0.23, respectively), which can be associated with more engagement and cognitive processing prior to taking action through a mouse click.

For the Gears game, the gaze points clusters for level 6 are shown in Figure 3. Here, we observe a spatial pattern similar to what was found for the Wheels game for the positioning of the first cluster for group 1, as, again, distinguished from groups 2 and 3. All groups showed the first fixation in less than 0.2s. The unique fixation points and the duration of these fixations found for the Gears game followed the same patterns observed in the Wheels game; there were more unique gaze points for group 1 compared with 2 and 3 (z= 0.66, -0.27, and -0.25, respectively) and longer durations on average for each eye fixation (z= 1.16, -0.63, and -

0.23, respectively). The number of mouse clicks for group 1 remained below the average of all students, but higher for group 3 (z= -0.34 and 0.86, respectively), while the number of clicks for group 2 (z= -0.35) was nearly identical to that of group 1.

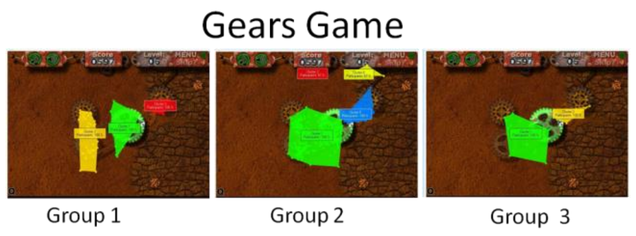


Figure 3: Gaze point clusters for groups 1, 2, and 3 at level 6 of the Gears game. Each gaze point cluster is represented by a color: green for cluster 1, yellow for cluster 2, and red for cluster 3.

5. CONCLUSION AND IMPLICATIONS

We have presented preliminary results from a study designed to determine how children approach engineering-related tasks embodied in interactive games. This work is situated within a larger research agenda, which is to apply analytics and data-mining techniques for open-ended, constructionist [8] learning activities ("multimodal learning analytics" [4, 10]).

From this small sample, we have observed that students have different eye movement patterns while interacting with the games. Shorter durations for first fixations after a stimulus presentation have been correlated with higher attentional readiness [9] and in this study they were associated with more time spent on the cognitive processing of the task prior to taking action through a mouse click. This pattern may suggest more engagement and reasoning prior to action, which is a valuable skill for students. On the other hand, longer times of first fixations after stimulus presentation, a higher number of mouse clicks, and shorter durations for each fixation point may suggest a "trial and error" approach, where the subject looks for different points on the screen without focusing on strategy or reasoning about the task.

These preliminary results need to be tested with a larger sample and more systematic tasks, but they may point to novel ways of determining students' expertise levels in engineering-related tasks. We believe that those kinds of games can be used as tools for training visuospatial abilities, especially if the task can bring "hands on" elements into mental rotation exercises, as has been done by others researching educational game design [11]. A second issue regarding the development of engineering and science thinking is the use of Bifocal Models [3], where students can undertake computer simulations of tasks through games similar to those that we have presented here, and proceed from that point to the performance of the activity with tangibles objects. Comparisons could then be drawn between the results gathered for the virtual and real undertakings.

We also suggest that further studies take into account ecological variables from the environment to be correlated with performance and eye patterns, such as school performance. A second approach for further studies could be the tracking of eye pattern changes in cognitive tasks after an interventions focusing on skill development.

By identifying elements in students' gaze that were correlated to higher performance in open-ended tasks, this paper contributes to the identification of markers of expertise [4, 10] that might help educators and practitioners learn to detect and assess expertise in unscripted tasks.

6. ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under the CAREER Award #1055130 and the Lemann Center at Stanford University.

7. REFERENCES

[1] AEHTMAN, R. L., C. S. GREEN, AND D. BAVELIER. 2003. Video games as a tool to train visual skills. *Restorative Neurology and Neuroscience* 26, 4354146.

[2] BERNARDI LUFT, C. D., GOMES, J. S., PRIORI, D., & TAKASE, E. 2013. Using online cognitive tasks to predict mathematics low school achievement. *Computers & Education*. 67, 291-228.

[3] BLIKSTEIN, P. 2012. Bifocal modeling: a study on the learning outcomes of comparing physical and computational models linked in real time. In *Proceedings of the 14th ACM International Conference on Multimodal Interaction, Los Angeles, California, ACM,* 257-264.

[4] BLIKSTEIN, P. 2013. Multimodal Learning Analytics. In *Proceedings of the Third International Conference on Learning Analytics and Knowledge*, Leuven, Belgium, April 2013, ACM, 102-106.

[5] BULL, R., & ESPY, K. A. 2006. Working memory, executive functioning, and children's mathematics. In S. J. Pickering (Ed.), *Working Memory and Education*, Burlington, MA: Academic Press, 94-123.

[6] CHUANG, T-Y., AND WEI-FAN C. 2007. Effect of Computer-Based Video Games on Children: An Experimental Study. *Educational Technology & Society* 12, 2, 1-10.

[7] FERGUSON, C. J., CRUZ, A. M., & RUEDA, S. M. 2008. Gender, video game playing habits and visual memory tasks. *Sex Roles*, *58*, 3-4, 279-286.

[8] PAPERT, S., *Mindstorms: children, computers, and powerful ideas.* 1980, New York: Basic Books.

[9] POOLE, A., AND BALL, L. J. 2005. Eye Tracking in Human-Computer Interaction and Usability Research: Current Status and Future. Prospects, In C. Ghaoui (Ed.): *Encyclopedia of Human-Computer Interaction*. Pennsylvania.

[10] WORSLEY, M. AND BLIKSTEIN, P. 2013. Towards the Development of Multimodal Action Based Assessment. In *Proceedings of the Third International Conference on Learning Analytics and Knowledge*. *Proceedings* April 2013, ACM, 94-101.

[11] WIEDENBAUER, G., & JANSEN-OSMANN, P. 2008. Manual training of mental rotation in children. *Learning and instruction*, *18*, 1, 30-41. [12] WU, H. K., & SHAH, P. 2004. Exploring visuospatial thinking in chemistry learning. *Science Education*, *88*, 3, 465-492.

[12] Wu, H. K., & Shah, P. (2004). Exploring visuospatial thinking in chemistry learning. *Science Education*, *88*(3), 465-492.

Investigating the efficacy of algorithmic student modelling in predicting students at risk of failing in tertiary education.

Geraldine Gray, Colm McGuinness, Philip Owende
Institute of Technology Blanchardstown
Blanchardstown Road North
Dublin 15, Ireland
geraldine.gray@itb.ie

ABSTRACT

The increasing numbers enrolling for college courses, and increased diversity in the classroom, poses a challenge for colleges in enabling all students achieve their potential. This paper reports on a study to model factors, using data mining techniques, that are predictive of college academic performance, and can be measured during first year enrolment. Data was gathered over three years, and focused on a diverse student population of first year students from a range of academic disciplines (n≈1100). Initial models generated on two years of data (n=713) demonstrate high accuracy. Advice is sought on additional analysis approaches to consider.

Keywords

Educational data mining, academic performance, personality, motivation, specific learning difficulties, self-regulation.

1. INTRODUCTION

In tertiary education, learning is typically measured by student performance based on a variety of assessments that are aggregated to generate a single measure of academic performance. Factors impacting on academic performance have been the focus of research for many years [9, 14]. It still remains an active research topic [5, 12], indicating the inherent difficulty in defining robust deterministic models to predict academic performance [16]. Typically, methodologies for quantitative research in this domain focus on statistical analysis of performance metrics and their correlations with, or dependencies on, a wide variety of factors including measures of aptitude, motivation, organisation skills, personality traits, prior academic achievements and demographic data [6, 11, 18]. More recently, Educational Data Mining (EDM) has emerged as an evolving and growing research discipline, covering the application of data mining techniques in educational settings [1, 4, 10, 19]. There have been calls for greater use of data mining by educational institutes to realise the potential of the large amounts of data gathered by institutes each year [7, 20]. While initial studies show promising results, a greater body of work is needed to determine if data mining techniques can offer an improvement over statistical methods [6, 11, 15].

It is increasingly evident that significant numbers of students in Institutes of Technology[1] (IoT) in Ireland do not complete the courses on which they enrolled [13]. Increased numbers enrolling in first year, and increased diversity in the student population, adds to the challenge of both identifying students at risk of failing, and planning appropriate supports to enable students perform optimally [13]. This study aims to investigate the suitability of classification techniques in generating a robust student model at enrolment which could identify students at risk of failing. The study focuses on two areas of research (1) an investigation of additional measures to augment the data currently gathered by college administration which will assist in the identification of students at risk, and (2) an investigation of suitable data mining techniques to accurately model this augmented dataset.

Study Hypothesis:. That educational data mining techniques can generate an accurate, deterministic model of academic performance based on factors measured at enrolment to tertiary education.

Study Objectives:

1. Identify and investigate factors most likely to determine academic performance in tertiary education, with a focus on factors that can be measured at enrolment.
2. Investigate the accuracy and stability of a range of classification techniques in predicting students at risk of failing in first year.
3. Compare the suitability of a data mining approach with a statistical approach for modelling a diverse student population.

2. EXPECTED CONTRIBUTION

This study adds to existing knowledge in the following ways:

[1]The Institute of Technology sector is a major provider of third and fourth level education in Ireland, focusing on the skill needs of the community they serve (www.ioti.ie).

1. Extends existing research in Education Data Mining: EDM has given much attention to datasets generated from students' behaviour on Virtual Learning Environments (VLE) and Intelligent Tutoring Systems (ITS) [4]. Less focus has been given to modelling datasets from outside virtual or online learning environments. This research focuses on models of college students that can be applied early in semester one.

2. Focus on third level students outside the university sector: Students enrolling in IoTs have, on average, a weaker academic history than university students[2], and there are increasing admissions of non-standard students [13]. This is an under-studied group compared to university students. A study of computing students [2] has shown that there is a difference between factors influencing the academic performance of university students compared to students in an IoT. This research extends that work by incorporating a wider range of factors and a diversity of students from across several academic disciplines.

3. The novel inclusion of data on specific learning difficulties: This study is based at the Institute of Technology Blanchardstown (ITB), who in partnership with the National Learning Network Assessment Services[3] (NLN) located on campus, provide assessment and follow up support for all students in four areas of specific learning difficulty: reading and spelling, organisation and co-ordination, social and communication, and attention and concentration. Profiling has shown one in five students at ITB report difficulties in at least one of these areas [8]. There is insufficient research including measures of specific learning difficulties in student modelling.

3. RESULTS SO FAR

3.1 Study criteria

Limited profiling of students in terms of specific learning difficulties and some learning preferences was already underway at ITB. This study extended that initiative, adding measures relating to four additional factors: aptitude, personality, motivation and learning strategies. These were chosen firstly because research highlights these factors as being directly or indirectly related to academic performance [21], and secondly because these factors can be measured early in semester one. An online questionnaire was developed to profile students and give immediate feedback (www.howilearn.ie). Data already available to college administration on prior academic performance was also used[4]. A full list of the factors used is given in Table 1.

[2]The majority of students in the IoT sector will have attained between 200 and 400 points in the Leaving Certificate exam, the state exam at the end of secondary school. The majority of students in the Irish university sector will have attained over 400 points, including some with the maximum score of 600 points [13, Appendix A].

[3]The NLN assessment team includes an educational psychologist, assistant psychologist and occupational therapist (http://www.nln.ie/Learning-and-Assessment-Services.aspx).

[4]Prior academic performance is based on state examinations completed by all students at the end of secondary school in Ireland.

Table 1: Measures included in the study

Prior Academic Performance	
English Grade	Did Honours English
Maths Grade	Did Honours Maths
Highest Mark	Humanities Average
Science Average	Creative/Practical Average
Aggregate Mark (CAO points)	
Personality, Goldbergs IPIP scales (http://ipip.ori.org)	
Conscientousness	Openness
Motivation, based on MSLQ [17]	
Intrinsic Goal Orientation	Self Efficacy
Extrinsic Goal Orientation	
Learning style, based on R-SPQ-2F [3]	
Deep Learner	Shallow Learner
Strategic Learner	
Self-regulated Learning, based on MSLQ [17]	
Self Regulation	Study Effort
Study Time	
Specific Learning Difficulties, Do-IT profiler (www.doitprofiler.info)	
Reading and Spelling	Social and Communication
Organisation and Co-ordination	Attention and Concentration
Preferred learning channel	
Visual, Auditory, Kinaesthetic, or a combination of these	
Other preferences, using Learning Styles Questionnaire from NLN (www.nln.ie)	
Organised or Disorganised	Morning or Evening
Meticulous or Approximate	Group work or solo
Logical or Creative	Like background noise
Other factors:	
Age	Gender

3.2 Study participants

Data was gathered on first year students over three academic years, 2010, 2011 and 2012. All students in the first year of study were invited to complete the online questionnaire as part of first year induction. 1,332 students completed the questionnaire. End of year results are available for two of the three years, giving a current sample size of (n=713). The final sample size is expected to be approximately 1,100 as to date 16% of students either gave an invalid student ID during profiling, or did not give permission for their data to be used in the study. Average CAO[5] points is 257.9 ± 75. 59% of the students were male. The students are from Computing, Engineering, Business, Social Care, Creative Digital Media, Sports Management and Horticulture.

3.3 Initial results

Modelling was done on the 2010 and 2011 data, using five dimensions, namely: prior academic performance, motivation, learning orientation, personality and age. A binary class label was used based on end of year GPA, range [0-4]. The two classes included poor academic achievers who failed overall (GPA<2, n=296), and strong academic achievers who achieved honours overall (GPA$\geq$2.5, n=340). To focus on patterns that distinguish poor and strong academic achievements, students with a GPA of between 2.0 and 2.5 were excluded from initial models, giving a dataset of (n=636). Six algorithms were used: Support Vector Machine(SVM), Neural Network, k-Nearest Neighbour, Naïve Bayes, Decision tree and Logistic Regression, using RapidMiner V5.2 (rapid-i.com). When modelling all students, model performance was comparable across the six learners, with Naïve

[5]CAO Points are an aggregate measure of prior academic performance in Ireland, range [0,600]. It represents the combined score achieved in six subjects.

Bayes achieving the best accuracy at 75.74%. However when modelling subgroups split by age, model accuracies for algorithms that can learn more complex patterns increased, with SVMs getting the best accuracy (82.62% for students under 21, 93.45% for students over 21). Other subgroups were not consider in the initial analysis.

4. OUTSTANDING QUESTIONS

Feedback and discussion is welcome on all aspects of the study, and specifically in the following areas:

1. The dataset has 44 attributes, primarily generated from a questionnaire using Likert scales, and so have a small range of discrete numeric values. Attributes are based on factors that have widely published correlations and interdependencies, although the reported significance of those dependencies varies. Opinions are sought on modelling approaches to consider for this type of dataset.

2. There have been calls from the EDM community for the use of statistical methodologies in data mining research [15]. Feedback on how this study should adhere to a statistical methodology to validate modelling results would be of value.

3. Also of interest are opinions on the value and limitations of early student modelling, before data on student engagement in course work is available. Is there value in also considering measures of early engagement based on activity on a VLE such as Moodle?

5. REFERENCES

[1] S. R. Barahate and M. Shelake, Vijay. A survey and future vision of data mining in educational field. *Second International Conference on Advanced Computing and Communication Technologies*, 2012.

[2] S. Bergin and R. Reilly. Predicting introductory programming performance: A multi-institutional multivariate study. *Computer Science Education*, 16, No. 4:303–323, 2006.

[3] J. Biggs, D. Kember, and D. Leung. The revised two-factor study process questionnaire: R-spq-2f. *British Journal of Education Psychology*, 71:133–149, 2001.

[4] T. Calders and M. Pechenizkiy. Introduction to the special section on educational data mining. *SIGKDD*, 13(2):3–3, 2011.

[5] S. Cassidy. Exploring individual differences as determining factors in student academic achievement in higher education. *Studies in Higher Education*, pages 1–18, 2011.

[6] G. Dekker, M. Pechenizkiy, and J. Vleeshouwers. Predicting students drop out: a case study. In T. Barnes, M. C. Desmarais, C. Romero, and S. Ventura, editors, *Proceedings of the 2nd International Conference on Educational Data Mining*, pages 41–50, Cordoba, Spain, 2009.

[7] N. Delavari, M. R. A. Shiraze, and M. R. Beikzadeh. A new model of using data mining technology in higher education systems. In *Proceedings of 5th International Conference in Information Technology Based Higher Education and Training*, Istanbul, Turkey, 2004.

[8] D. Duffin and G. Gray. Using ict to enable inclusive teaching practices in higher education. *AAATE, Florence*, Sept 2009.

[9] T. Farsides and R. Woodfield. Individual differences and undergraduate academic success: the roles of personality, intelligence, and application. *Personality and Individual Differences*, 34:1225–1243, 2003.

[10] Y. Gong, D. Rai, J. E. Beck, and N. T. Heffernan. Does self-discipline impact students' knowledge and learning? *Proceedings of the 2nd International Conference on Educational Data Mining*, pages 61–70, 2009.

[11] S. Herzog. Estimating student retention and degree-completion time: Decision trees and neural networks vis-à-vis regression. *New Directions For Institutional Research*, pages 17–33, 2006.

[12] M. Komarraju, A. Ramsey, and V. Rinella. Cognitive and non-cognitive predictors of college readiness and performance. role academic discipline. *Learning and Individual Differences*, 24, 2013.

[13] O. Mooney, V. Patterson, M. O'Connor, and A. Chantler. A study of progression in higher education: A report by the higher education authority. Technical report, Higher Education Authority, Ireland, October 2010.

[14] M. A. Moran and M. J. Crowley. The leaving certificate and first year university performance. *Journal of Statistical and Social Enquiry in Ireland*, XXIV, part 1:231–266, 1979.

[15] B. Nelson, R. Nugent, and A. A. Rupp. On instructional utility, statistical methodology, and the added value of ecd: Lessons learned from the special issue. *Journal of Educational Data Mining*, 4 (1):227–233, 2012.

[16] Z. A. Pardos, R. S. J. D. Baker, S. M. Gowda, and N. T. Heffernan. The sum is greater than the parts: Ensembling models of student knowledge in educational software. *SIGKDD Explorations*, 13(2), 2011.

[17] P. Pintrich, D. Smith, T. Garcia, and W. McKeachie. A manual for the use of the motivated strategies for learning questionnaire. Technical Report 91-B-004, The Regents of the University of Michigan, 1991.

[18] S. B. Robbins, K. Lauver, H. Le, D. Davis, and R. Langley. Do psychosocial and study skill factors predict college outcomes? a meta analysis. *Psychological Bulletin*, 130 (2):261–288, 2004.

[19] C. Romero and S. Ventura. Educational data mining: A survey from 1995 to 2005. *Expert Systems with Applications*, 33:135–146, 2007.

[20] C. Romero and S. Ventura. Educational data mining: A review of the state of the art. *IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews*, 40(6):601–618, 2010.

[21] N. Schmitt, F. L. Oswald, T. Pleskac, R. Sinha, and M. Zorzie. Prediction of four-year college student performance using cognitive and noncognitive predictors and the impact on demographic status of admitted students. *Journal of Applied Psychology*, 2009.

BOTS: Harnessing Player Data and Player Effort to Create and Evaluate Levels in a Serious Game

Andrew Hicks
NC State University
890 Oval Drive
Raleigh, NC 27606

ABSTRACT

BOTS is a socially multiplayer online game designed to teach students about introductory computer science concepts such as loops and functions. Using this game, I plan to explore the use of user-generated content (UGC) in game-based tutors, increasing replayability and, ideally, player engagement. BOTS has so far been used for work towards identifying what makes a level or puzzle in the game "good" and how I identify that quality in new submissions, as well as investigating several mechanisms for moderation of submitted content. The use of UGC has the potential to revolutionize how game-based tutors are created, drastically reducing the burden of content creation on developers and educators.

Keywords

Game Based Tutors, Moderation, Player Engagement, Self-Evaluation, Serious Games, Social Games, User Generated Content..

1. INTRODUCTION

1.1 Background

Computer assisted learning, and game-based learning in particular has been shown to be able to be nearly as effective as one-on-one human tutoring [8, 10], however the developers and educators are required to use a great deal of time and expert knowledge [2, 14]. Murray estimated approximately it takes 300 hours to create a single hour of educational content. If concerns for game design, user immersion, and content creation are considered, this time cost would only increase. Additionally, problems created by educators or developers are often presented in a sequence, and once the in-game content is exhausted, the experience is generally over. Replayability is a major component of successful games [20], and games constructed in this way simply cannot be replayable experiences.

According to Scott Nicholson, allowing users to create game content, such as new levels and puzzles, "extends the life of a game and allows the designers to see how creative users can be with the toolkits provided." Many principles from the use of User-Generated Content (UGC) can be used to improve Serious Games by allowing players to set their own goals [16]. Additionally, design patterns for educational games identified by a team at Microsoft Research in [18, 19] indicate that allowing users to create their own challenges is a very powerful motivator.

Previous work done with UGC in serious games showed that level creation increased player motivation, especially for players interested in creativity [6]. By creating games that are solitary, non-replayable experiences, serious games developers are failing to harness the community experience that modern games provide, and may fail to provide ways to refine learned skills outside of rigidly structured areas. Even outside of ITS, software like Scratch, Alice, and other programs often feature communities that highly resemble Steam, Miiverse, and even communities like Wikipedia or YouTube [9, 15], in an effort to provide this type of experience.

1.2 User Generated Content in BOTS

To investigate how best to use UGC in a Serious Game, I present BOTS. BOTS is a socially multiplayer online game built using the Unity 3D Game Engine where players take on the task of programming various robots. Programs take the form of a graphical pseudo-code, where players drag and drop icons representing various commands to direct the robot. Players also can condense a sequence of commands into a single icon by creating a function, and can create loops and conditional statements to further optimize their solutions.

In BOTS, players have to manage several resources to succeed. The size of the players' programs is restricted to 25 commands. Players also have a limited number of instances of each command to work with. Both of these constraints are designed to encourage players to minimize the repetition of code, using functions and loops where necessary. These are important lessons for novice programmers to learn, and in this environment, can be taught independent of any specific language's syntax.

BOTS also has a collaborative/social aspect. Players can create new puzzles and share them with the game community. Our goal with this feature is to promote a higher level of engagement so that game-based tutors like BOTS can be used as more than a novel substitute for a homework assignment. BOTS should be a full-fledged educational tool that can be used by players throughout their introduction to programming The game was designed using the "Flow of Inspiration" principles outlined in [21]. This creates an environment where players can continually challenge their peers to find better solutions for difficult levels. Previous work has shown that content creators spend more time on their tasks when they have a target in mind [1]. I hypothesize that orienting content-creation as a social task may increase the quality of levels created.

While UGC certainly has a lot to offer for a system like ours, there are also several downsides to it, which can hinder or disrupt game-play. If it is possible to create a system within which users can be trusted to create useful, quality content, then developers will be able to spend more time developing the core of the game, ensuring that the game mechanics are both fun for players and in line with learning objectives. With these advances, we will be able to address many of the NSF's goals for Cyberlearning [17], and BOTS and systems like it may expand to play a more important role in early STEM education.

2. METHODS

In [2], the authors used a Machine Learning approach based on the tagging habits of users to identify low-quality Wikipedia articles. I hope to be able to use similar data-driven methods to analyze user-created levels. Lacking a large installed user-base to tag submitted levels, I work with player solutions as they are submitted, with the first being the author's own solution to the submitted level.

While the quality of a game level is subjective, I developed a set of criteria for a level in our game to be "useful", inspired by the use of design patterns in level design analysis [11]. To identify common design patterns, I examined levels created over the course of several game sessions with three groups. 24 levels were created by the first group, 13 by the second, and 11 by the third. Once I identified these patterns, I detailed how they could impact gameplay, why a level creator could be motivated to create them, and how developers could affect that motivation through game mechanics or incentives.

2.1 Identifying Low-Quality Submissions

Through examination of the existing levels, I identified several patterns of unwanted UGC. Interestingly, these types of levels fall in line with the player behavior types outlined in Bartle's work with online communities [4, 5], **killers, achievers, explorers,** and **socializers.** In the interest of space, I will specifically name only four of these low-quality design patterns here.

- **"Sandbox" levels**, which feature erratically-placed elements. I believe these levels are often created by users who are unfamiliar with the creation interface.
- **"Punisher" levels**, which feature unusually difficult or tedious solution paths, characterized by programs which are trivial but time-consuming to write.
- **"Griefer" levels**, which feature visual obstacles or other abuses of game mechanics, which I believe are intended to frustrate the user. These levels may or may not have solutions.
- **"Trivial" levels**, whose optimal solution is readily apparent to players, and which requires no use of the game's more advanced or difficult concepts to complete.

Based on these classes of unwanted UGC, I developed an evaluation rubric to score levels based on the features they contain. For our purposes, a "high quality" level should:

- Contain an obvious trivial solution
- Contain a different, optimized solution
- Contain structural cues for that optimization
- Contain few unnecessary structural cues
- Take less than 5 minutes for an expert to solve

To explain the last criteria, compare this to a long, completely featureless level in a 2D platform game. The task itself is not providing difficulty, but achieving what should be a simple goal has become unnecessarily obtrusive [12, 13]. I used the above criteria to evaluate levels in the next part of our investigation.

2.2 Moderating User Submissions

To see how different game mechanics affected the levels created, we implemented several different types of moderation which we believed could discourage players from submitting low-quality UGC. Students at a STEM-related after-school program played BOTS for one hour under one of three conditions, and we examined the levels they created using the rubric I had previously developed. The three types of moderation investigated were as follows:

Condition 1: Unrestricted Level Submission

There is no filtration process in place and the puzzle must only pass the base conditions of each level having a starting point and goal. If the level has those conditions, it will be made public and immediately available for play as soon as the participant submits it.

Condition 2: Self-Evaluation

The participant must first submit a solution for the level they just created before it would be made accessible to the public. I expect that this will reshape the level creation process so that more successful creators will build levels while already having a solution in mind.

Condition 3: Moderator Approval

When a participant submits a level, it will be placed in a queue where an admin can examine the level and determine if it is appropriate to publish. The admin will then reply to the participant either accepting or rejecting that level which was submitted for approval.

2.3 Preliminary Results

After the session, a researcher who was blind to the conditions each level was created under "graded" each level on a simple rubric addressing the criteria discussed above.

	Condition 1	Condition 2	Condition 3
Created	3	4	9
Published	3	2	4
M(Quality)	2	2.5	2.3
M(Published Quality)	2	5	4.5

Figure 1 - Measured quality of submitted levels

I also analyzed the best solutions to these levels using an expert-solver, looking at the difference (in terms of number of commands used) between a naive solution using neither loops nor functions and a master solution using a combination of both techniques.

	Condition 1	Condition 2	Condition 3
Mean	9	11.5	18.6666667
SD	4.58257569	14.0830868	19.4250697
Max	14	32	40
Min	2	0	2

Figure 2 - Differences between naive and expert solutions

Though we were able to collect relatively few levels, the collected data are encouraging. The quality of the published levels in

Condition 2 is similar to that of the levels under Condition 3. Interestingly, in both conditions where some form of moderation is present, the average quality of all levels, including those left incomplete or unpublished. is slightly higher. Though I have a very small sample size in this study, I hope to be able to investigate these effects with a larger group of players.

3. FUTURE WORK

In addition to replicating the above experiment with a larger group of students, I have already begun an investigation of how to further use student data to moderate and evaluate submitted levels. Being able to assess UGC in this way allows us to provide meaningful problem orderings even with levels I have not analyzed in depth, as well as provides us with a metric which can be used to reward players for creating specific types of levels, or levels which fill in gaps in content or difficulty. In the future, I will experiment with different methods of directed level creation using the information gained, to see if level creation can be better integrated into the system as a learning activity in and of itself.

4. ACKNOWLEDGEMENTS

Thanks to my advisor, Dr. Tiffany Barnes, and to the developers who have worked on BOTS so far, including Veronica Catete, Monique Jones, Andrew Messick, Thomas Hege, Michael Clifton, Vincent Bugica, Victoria Cooper, Dustin Culler, Shaun Pickford, Antoine Campbell, and Javier Olaya. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No 0900860 and Grant No 1252376.

5. REFERENCES

[1] Aleahmad, T., Aleven, V., and Kraut, R. Open community authoring of worked example problems. In Proceedings of the 8th international conference on International conference for the learning sciences (ICLS'08), Vol. 3. 3-4.

[2] Anderka, M., Stein, B., and Lipka, N. Predicting quality flaws in user-generated content: the case of wikipedia. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval* (SIGIR '12). ACM, New York, NY, USA, 981-990.

[3] Bayliss, J. D. Using games in introductory courses: Tips from the trenches. In *Proceedings of the 40th ACM Technical Symposium on Computer Science Education*. (SIGCSE '09). ACM, New York, NY, 337-341

[4] Bartle, R. A.. Hearts, clubs, diamonds, spades: Players who suit MUDs. *Journal of MUD research*, 1-19, 1996.

[5] Bartle, R. A. *Designing Virtual Worlds.* Boston, MA: New Riders / Pearson Education. 2004

[6] Boyce, A., Doran, K., Pickford, S., Campbell, A., Culler, D., and Barnes, T. BeadLoom Game: Adding Competitive, User Generated, and Social Features to Increase Motivation. In *Proceedings of the Foundation of Digital Games* (FDG '11). ACM, New York, NY, USA, 243-247.

[7] Carmel, D., Roitman, H., and Yom-Tov, E. 2012. On the Relationship between Novelty and Popularity of User-Generated Content. *ACM Trans. Intell. Syst. Technol.* 3, 4, Article 69 (September 2012)

[8] Chaffin, A., Doran, K. Hicks, D., and Barnes, T. Experimental evaluation of teaching recursion in a video game. *In Proceedings of the 2009 ACM SIGGRAPH Symposium on Video Games* (Sandbox '09), Stephen N. Spencer (Ed.). ACM, New York, NY, USA, 79-86.

[9] de Kereki, I.F. Scratch: Applications in Computer Science 1. In *Proceedings of Frontiers in Education Conference*, (FIE 2008). 22-25 Oct. 2008

[10] Eagle, M., Barnes, T. Experimental evaluation of an educational game for improved learning in introductory computing. In *Proceedings of the 40th ACM technical symposium on Computer science education* (SIGCSE '09). ACM, New York, NY, USA, 321-325.

[11] Hullett, K. and Whitehead, J. Design patterns in fps levels. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games* (FDG '10) ACM, New York, NY, USA, 2010.

[12] Juul, J. In search of lost time: on game goals and failure costs. In Proceedings *of the Fifth International Conference on the Foundations of Digital Games* (FDG '10) ACM, New York, NY, USA, 2010.

[13] Juul, J. Easy to use and incredibly difficult: on the mythical border between interface and gameplay. In *Proceedings of the Sixth International Conference on the Foundations of Digital Games* (FDG '11) ACM, New York, NY, USA, 2011.

[14] Murray, T. An overview of intelligent tutoring system authoring tools: Updated analysis of the state of the art. In *Authoring Tools for Advanced Learning Environments*. T. Murray, S. Blessing, and S. Ainsworth (Eds.) Chapter 17, 491-544. Dordrecht, the Netherlands: Kluwer Academic Publishers.

[15] Malan, D, Leitner, H, Scratch for budding computer scientists. In *Proceedings of the 38th SIGCSE technical symposium on Computer science education* (SIGCSE '07). ACM, New York, NY, USA, 223-227.

[16] Nicholson, S. A User-Centered Theoretical Framework for Meaningful Gamification. In *Games+Learning+Society* (GLS 2012), Madison, WI.

[17] NSF Publication. Cyberlearning: Transforming Education. www.nsf.gov/pubs/2010/nsf10620/nsf10620.htm

[18] Plass, J., and Homer, B. Educational Game Design Pattern Candidates. White Paper, Institute for Games for Learning, 2009 http://g4li.org/research

[19] Plass, J., and Homer, B. Learning Mechanics and Assessment Mechanics for Games for Learning. White Paper, Institute for Games for Learning, 2011 http://g4li.org/research

[20] Prensky, M. Computer Games and Learning: Digital Game-Based Learning. In *Handbook of Computer Games Studies*. Cambridge MA, MIT Press; 2005

[21] Repenning, A., Basawapatna, A., and Koh, K. H. Making university education more like middle school computer club: facilitating the flow of inspiration. In *Proceedings of the 14th Western Canadian Conference on Computing Education* (WCCCE '09). ACM, New York, NY, 9-16.

Helping Students Manage Personalized Learning Scenarios

Paul Salvador Inventado[*] , Roberto Legaspi and Masayuki Numao
The Institute of Scientific and Industrial Research, Osaka University
8-1 Mihogaoka, Ibaraki, Osaka, Japan, Osaka, 567-0047
{inventado,roberto}@ai.sanken.osaka-u.ac.jp, numao@sanken.osaka-u.ac.jp

ABSTRACT

In personalized learning scenarios, students have control over their learning goals and how they want to learn which is advantageous since they tend to be more motivated and immersed in what they are learning. However, they need to regulate their motivation, affect and activities so they can learn effectively. Our research deals with helping students identify the long-term effects of their learning behavior and identify effective actions that span across learning episodes which are not easily identified without in depth analysis. In this paper, we discuss how we are trying to identify such effective learning behavior and how they can be used to generate feedback that will help students learn in personalized learning scenarios.

Keywords

personalized learning, self-regulated learning, reinforcement learning, user modeling

1. INTRODUCTION

Governments and educational institutions have called for reforms on how students are taught in school to enable them to have more control over their learning [1]. Allowing students to engage in personalized learning grant them skills that prepare them for the needs of the current society and more importantly help shape them into life-long learners.

In personalized learning, students have control over what they learn and how they learn causing them to be more motivated and immersed in what they are learning. Teachers no longer serve as the main sources of information but instead become facilitators of the students' learning process. Although teachers can guide students and give them suggestions about what they are learning, teachers can only assess and provide support for a small number of the challenges that students face. Especially because students learn in situations where teachers are unavailable, students can easily get overwhelmed by challenges and not achieve their aspired learning goals. It is also possible that students would engage in non-learning related activities which might hinder them from learning. Thus, in this kind of learning scenario, self-regulation is essential for students to manage their goals, time, motivation, affective states and hindrances to learning.

[*]also affiliated with: Center for Empathic Human-Computer Interactions, College of Computer Studies, De La Salle University, Manila, Philippines

Self-regulation is not an easy task because it requires much motivation and effort [5]. There is a high cognitive load when students perform learning tasks while managing it. They would need to continuously monitor the effects of their actions and decide if they should continue doing it or if they should change it. Furthermore, students also keep track of effective learning behavior so they can use them in future learning episodes.

We have been developing a software that helps students monitor their behavior and reflect on what transpired during the learning episode with the help of webcam and desktop snapshots [3]. After each learning episode, students who used the system were asked to review their learning episode then annotate their intentions, their activities and their affective states so they could further understand and analyze their behavior. According to the results, students who used the system discovered behaviors they were initially unaware of and were able to identify ways to improve ineffective learning behavior. We were also able to analyze and process the students' annotated data to have a better understanding of their learning behavior.

Students' reflections from the experiment however, seemed to focus only on immediate effects of their actions and did not consider its long term effects in the learning episode. Also, their reflections did not incorporate their realizations from previous learning episodes. Currently, we are investigating how we can help students identify actions that benefit learning not only in the short-term but also in the long-term. We also want to help students to identify effective learning behavior that span over different learning episodes.

2. STUDENT LEARNING BEHAVIOR

The data we used for this research was gathered from four students engaging in research-related work, which is an example of a personalized learning scenario. One male masteral student and one female doctoral student created a report about their research involving activities such as information search, reading papers, reading books and creating a power point presentation. One male undergraduate student and one female doctoral student wrote a conference paper about their research involving activities such as information search, reading papers, reading books, running programs and simulations to retrieve data from their experiments and paper writing. We gathered two hours of data for five different learning episodes from each student within a span of one week.

Unlike other research, our work dealt with students who freely decided on the time, location and type of activities they did including non-learning related activities. However, they were required to learn in front of a computer running the software we developed for recording and annotating learning behavior.

Although the students worked on different topics and used different applications, all of them processed and performed experiments on previously collected data, searched for related literature and created a report or document about it. Analyzing the data showed that students performed six types of actions – information search (e.g., using a search engine), view information source (e.g., reading a book, viewing a website), write notes, seek help from peers (e.g., talking to a friend), knowledge application (e.g., paper writing, presentation creation, data processing) and off-task (e.g., playing a game).

3. BEHAVIOR EFFECTIVENESS

In a learning episode, students perform many different actions to achieve their goal. Although students can identify the effectiveness of the current action by monitoring its effect, it is more difficult to identify how it will affect or how it has affected their learning in the long run. For example, students spending a long time learning about a topic would seem to be performing well, however they may experience more stress and have a higher chance of making mistakes and getting confused more easily. It would probably be advantageous for the student to also take a rest once in a while. We adapted the concept of *returns* in reinforcement learning [4] to account for this situation wherein the effectiveness of an action was not measured only by its immediate effects but rather its long term effects on the learning episode. Moreover, as the student engaged in more learning episodes, a reinforcement learning algorithm updated the rewards of each action which incorporated the effects of actions from previous learning episodes.

Due to the lack of control in the students' activities while learning, it was not possible to directly gauge the students' learning progress which could have been used to define the rewards of their actions. However, their affective states gave an idea about the events that transpired during the learning episode. D'Mello and Graesser's model of affective dynamics [2] describes the relationship between affective states and events that occur in a learning scenario. For example, confusion indicates instances wherein students need to exert more effort to progress in the current activity. Frustration arises when students are too confused, get stuck and no longer progress in their learning. Too much frustration results in boredom or disengagement from the learning activity. Engagement and delight on the other hand are indicators that a student is moving towards or has achieved the learning goal. Although D'Mello and Graesser's model does not discuss off-task activities in particular, it is logical to consider that they will not directly lead to learning progress. Negative affective states experienced while performing off-task activities might cause a decrease in motivation so it is probably best to avoid them while learning. Based on how each affective state and type of activity affected learning progress, we constructed a reward system (see Table 1) that would be used to update the returns of performing an action.

Table 1: Action-Affect Reward System

Affect	On-task Behavior	Off-task Behavior
Engaged	3	-
Confused	2	-
Frustrated	1	-
Bored	-1	-
Neutral	0	-2
Delighted	3	-2
Surprised	2	-2
Sad	-	-3
Angry	-	-3
Disgusted	-	-3
Afraid	-	-3

Using the reward system we defined, the long-term effectiveness of the actions performed in the learning episode can be discovered using a reinforcement algorithm. Specifically we used Q-learning [4] to discover actions that maximize return when performed in a particular state. In our case, we represented states using the learning context and actions using the activities performed by the student. Specifically, each state was represented using – the current affective state, the amount of time spent in the current state, the previous action performed, the previous affective state experienced, the dominant action previously used and the dominant affective state experienced. States changed when students chose to perform a different activity (e.g., shfting from viewing an information source to seeking help) so this was used to represent an action.

The collected data was manually processed and then converted into state-action pairs. Q-learning was then applied to uncover the returns of performing actions in a particular state. The state-action pairs with their corresponding expected returns were called the student's learning policy.

4. RESULTS

The Q-learning algorithm was applied on each of the student's data separately since we assumed that each student would have a different learning policy. Due to the number of features we used for state representation , there were a lot of states and many of them had high return values. Due to space limitations, we only present some of the notable state-action pairs from one of the student's learning policy in Table 2. Majority of the states with high return values contained state-action pairs that represented transitions inherent to the domain. For example, high returns were given when students applied knowledge after viewing an information source, which happens naturally for example when a

Table 2: Sample State-Action Returns

State	Action	Reward
Engaged while viewing an information source for <5min, Previously engaged while applying knowledge, Mostly felt engaged while applying knowledge	Apply knowledge	6469.20
Confused while applying knowledge for <5min, Previously engaged while viewing information source, Mostly felt engaged while applying knowledge	Apply knowledge	982.80
Engaged while applying knowledge for 5-10min, Previously off-task, Mostly felt engaged while applying knowledge	Off-task	164.30
Neutral while applying knowledge for <5min, Previously engaged while applying knowledge, Mostly felt engaged while applying knowledge	Off-task	-228.60
Delighted while doing off-task behavior for 5-10min, Previously confused while viewing an information source, Mostly felt engaged while viewing an information source	View information source	521.10

student shifts between reading information sources and creates a power point presentation. However, some interesting strategies were discovered such as shifting from an engaged on-task activity to an off-task activity indicating that off task activities may actually have positive long term effects.

Students' answers from surveys and personal interviews regarding their thoughts on a recent learning episode correlated with the reward values produced by the algorithm. For example, students identified the need to continue learning despite encountering challenges (e.g., confusion) and not spending too much time in off-task activities.

5. FUTURE DIRECTION

The next step in the research is helping students find ways to improve their learning behavior. We believe that the behavior identified using the reinforcement learning approach can be used to support students by making them aware of the behavior's long term effects and also informing them of effective learning behavior that have spanned across their learning episodes.

Students' behavior in a learning episode can be evaluated by comparing the actions that a student took in a particular state with the optimal action according to the student's updated learning policy. When a student selects a suboptimal action, the system can inform the student that an ineffective learning strategy might have been used and taking the optimal action could improve their learning effectiveness. Effective learning behavior that span across learning episodes can be identified by keeping track of frequently used state-action pairs that constantly garner high returns in different learning episodes. Students can be informed of such behavior so they will be aware of them and can make sure to apply them in succeeding learning episodes.

We also plan to investigate how students will react to feedback using the policy generated by the reinforcement learning algorithm and observe if it will help students select more effective learning behavior. It will also be interesting to see how differently students will react to feedback when different reward systems are used. Apart from using a students' learning policy we also think that they can benefit from learning about other students' effective learning behaviors taken from other students' learning policies. Moreover, learning behaviors identified by experts which are not exhibited by the student can also be suggested.

Another way to identify more accurate reward values would be to include effectiveness ratings of the actions performed by the students. It might also be good to explore other features for our state representations and see how they affect the resulting learning policy. Lastly, we are also investigating other reward mechanisms that are more flexible so it can handle students' individual differences.

Acknowledgements

This work was supported in part by the Management Expenses Grants for National Universities Corporations from the Ministry of Education, Culture, Sports, Science and Technology of Japan (MEXT) and JSPS KAKENHI Grant Number 23300059. We would also like to thank all the students who participated in our data collection.

6. REFERENCES

[1] D. E. Atkins, J. Bennett, J. S. Brown, A. Chopra, C. Dede, B. Fishman, L. Gomez, M. Honey, Y. Kafai, M. Luftglass, R. Pea, J. Pellegrino, D. Rose, C. Thille, and B. Williams. *Transforming American education: Learning powered by technology.* Nov. 2010.

[2] S. D'Mello and A. Graesser. Dynamics of affective states during complex learning. *Learning and Instruction*, 22(2):145–157, Apr. 2012.

[3] P. S. Inventado, R. Legaspi, R. Cabredo, and M. Numao. Student learning behavior in an unsupervised learning environment. In *Proceedings of the 20th International Conference on Computers in Education*, pages 730–737, Dec. 2012.

[4] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning).* The MIT Press, Mar. 1998.

[5] B. J. Zimmerman. Self-regulated learning and academic achievement: An overview. *Educational psychologist*, 25(1), 1990.

Determining Problem Selection for a Logic Proof Tutor

Behrooz Mostafavi
North Carolina State University
Raleigh, NC 27695
bzmostaf@ncsu.edu

Tiffany Barnes
North Carolina State University
Raleigh, NC 27695
tmbarnes@ncsu.edu

ABSTRACT

When developing an intelligent tutoring system, it is necessary to have a significant number of highly varied problems that adapt to a student's individual learning style. In developing an intelligent tutor for logic proof construction, selecting problems for individual students that effectively aid their progress can be difficult, since logic proofs require knowledge of a number of concepts and problem solving abilities. The level of variation in the problems needed to satisfy all possibilities would require an infeasible number of problems to develop. Using a proof construction tool called Deep Thought, we have developed a system which chooses existing problem sets for students using knowledge tracing of students' accumulated application of logic proof solving concepts and are running a pilot study to determine the system's effectiveness. Our ultimate goal is to use what is learned from this study to be able to automatically generate logic proof problems for students that fit their individual learning style, and aid in the mastery of proof construction concepts.

Keywords

Logic Proof, Problem Selection, Knowledge Tracing, Intelligent Tutor.

1. INTRODUCTION

Logic proof construction is an important skill in several fields, including computer science, philosophy, and mathematics. However, proof construction can be difficult for students to learn, since it requires a satisfactory knowledge of logical operations and their application, as well as strategies for problem solving. These required skills make developing an intelligent tutor for logic proof construction challenging, since a number of variables must be taken into account when selecting problems for students that promote learning of proof concepts that fit their individual learning styles.

We describe the on-going development of an intelligent tutor, and an initial experiment to determine the effectiveness of knowledge tracing methods used to select sets of problems for students. For the study, we have built upon an existing, non-intelligent proof construction tool called Deep Thought, which has previously been used for proof construction assignments, and from which student performance data has been collected.

Our long-term goal is to provide a system for logic proof construction that adapts to a student's individual learning abilities, using that student's previous performance in logic rule application and problem solving in order to automatically generate problems that aid in mastery of core proof construction concepts. It is also our goal to develop the system in such a manner that it is domain independent, and can be applied to other fields that have multiple concepts and skills that need to be demonstrated.

2. THE DEEP THOUGHT TUTOR

2.1 The Original System

Deep Thought is a web-based proof tool with a graphical user interface that provides a set of problems that display logical premises, buttons for logical rules, and a conclusion that a student must prove by applying those rules to the premises (Figure 1). Deep Thought was originally developed as a practice tool and system for proof construction assignments. In its original form, Deep Thought provides students with three levels of problems, with problems in each level requiring a different set of logical rules for completion (Level 1: Inference rules; Level 2: Inference rules [more difficult]; Level 3: Inference and Replacement rules). Problems are selected from a drop-down menu, and students can select and complete problems in any order. As a student works through a problem, each step is logged in a data file that records a number of attributes, including the current problem, the rule being applied, any errors made (such as attempting to use a rule that is logically impossible), completion of the problem, time taken per step, and elapsed time taken to solve the problem.

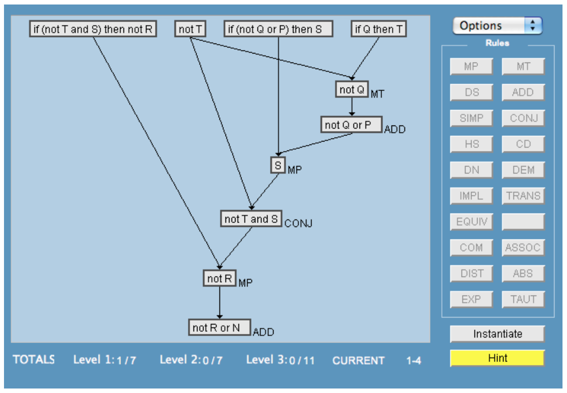


Figure 1. The Deep Thought user interface.

2.2 The New System

A number of changes were made to Deep Thought in order to create an intelligent system. Notable changes important to this study are described below.

2.2.1 Problem Set

Instead of allowing students to select problems at will, the new system provides an ordered set of problems for students to solve. Problem selection is determined based on the level of rule application and difficulty students are expected to demonstrate. Students can skip problems within the current level; however, they must complete all problems within that level to proceed to the next.

The original problem set for Deep Thought was expanded to give a wide variety of problems while maintaining the rule applications required and difficulty level of the original set. These changes were made and tested by domain experts to ensure consistency between the old and new system for performance comparison. The problem set was changed as follows:

- Levels 1 & 2: Inference rules
- Levels 3 & 4: Inference rules [more difficult]
- Levels 5 & 6: Inference and Replacement rules
- Level 7: Inference and Replacement rules [more difficult, not present in original set]

Level 1 contains 3 problems common to all users. With no prior performance data available, these 3 problems serve the purpose of collecting initial performance data to select problems in the next level.

Levels 2 – 6 are split into two difficulty tracks (easy and hard), to which students are sent based on their prior performance. Both tracks within each level contain problems that require similar rule applications and proof concept demonstration. The hard path contains 2 problems and the easy path contains 3 problems, each with an alternate problem (the alternate problem contains the same number of steps and same rules as the original, but with different ordering of required rule applications). The difficulty of problem sets were determined by domain experts who have many years of experience working with the types of proofs presented in Deep Thought and with students working through those problems.

Level 7 contains 3 problems common to all users. These problems were not present in the original set, but were added to test student skills obtained by working through the rest of the tutor. The problems in this level were more difficult than any other problems in Deep Thought.

2.2.2 *Problem Selection*

Problem selection in Deep Thought is determined using two methods. The first is the decision process that occurs between levels that sends a student down difficulty paths. The second is the process that selects problems within the current level.

For the difficulty path decision process, data from a student's work in Deep Thought is recorded and used to update a set of action scores. The scores for each rule are given an initial value, and are then updated based on the actions taken by the user, with correct applications of rules increasing the rule score, and incorrect actions (errors) decreasing them. The calculations for rule updates are made using a Bayesian knowledge-tracing model [2].

At the end of each level, the scores for each action are compared to average scores from historical student data collected using the old version of Deep Thought. The scores from the old version were calculated using the same bayesian knowledge tracing model after students had worked through the existing problems sets, and were used as a threshold value. Each rule is given a value of 1 if the score is higher than the threshold and given a value of -1 if the score is lower than the threshold. For each action, these values are weighted based on the rule priority for each level (primary or secondary), and then summed. A sum less than zero sends the students down the easy path, and a sum greater than zero send the students down the hard path.

Within each level, problems are selected using a decision tree process, based on whether or not students skip problems. Students who are working within the easy difficulty track are given the alternate problem if they choose to skip the original problem presented, with the idea that the difference in rule application order can allow them to approach the concept in a different manner. For students working in the hard difficulty track, skipping more than the first problem in the set will send them to the easy difficulty track. If students solved one problem in the hard difficulty path before being sent to the easy difficulty path, they are not required to solve the corresponding problem in the easy path, in order to maintain the number of problems required to complete the level.

The reason for the skipped problem decision process is to compensate for students who may have shown proficiency in a previous level, but have a harder time solving the next set of problems. Students who have been sent down the hard difficulty track are expected to have satisfactory mastery of concepts needed for the next set of problems, without the need for alternate problems. If students have difficulties with the harder set, they are given the opportunity to work through a greater number of easier problems in order to practice those concepts required before moving on to the next level.

3. METHOD & INITIAL RESULTS

The new system of Deep Thought was used as an assignment in two sections of a Computer Science Logic & Algorithms class taught by the same instructor. Deep Thought was run as a web applet, with students allowed to work through the problem sets at their own pace. The more difficult Level 7 problems were made optional to students, as they were not presented in the original curriculum for the course. Student data was recorded in two separate tables in a database stored on a server which communicated with the Deep Thought applet. The two tables used were:

- Log Table: This table was used to track information specific to individual students, including information used for tracking a student's progress in the system (log-in information, current working level / difficulty / problem, skipped and completed problems in the current level, levels completed and at which difficulty track) as well as data used for the knowledge tracing process (updated scores for individual rules and concepts).
- Data Table: This table was used, as in the original system, to track each action taken by students while solving proofs for analysis (level / difficulty / problem, the rule being applied, errors made, screen state, hints used, action step time, and total elapsed time for the current problem).

A total of 63 students worked through the new version of Deep Thought. Of these students, 32 completed at least through Level 6 of the problem sets, with the majority of drop-outs occurring after Level 4. The number of students who did not complete Deep Thought was high (over 50%), however it should be noted that the professor for the class used for the experiment had not completely covered Replacement rules (used in Level 5 onwards) at the time these results were reported. A flow diagram showing the path students travelled while using Deep Thought is shown in Figure 2.

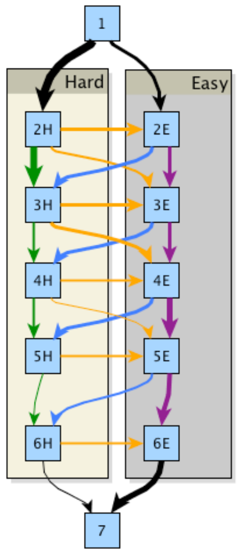


Figure 2: Flow diagram of student path through Deep Thought problem sets. The thickness of the arrows is weighted based on the number of students travelling that path.

Based on the diagram in Figure 2, the following conclusions can be drawn regarding the paths commonly taken by the students. Most of the class was able to complete the hard paths for levels 1 and 2, with most of the students being sent down the hard path once level 1 was completed and staying through level 2. At level 3 however, some of the students were sent to the easy path, either at the same level or at level 4. From level 4 onwards most of the class stayed on the easy path (those who completed Deep Thought). From Level 5 onwards, most of the students stayed on the easy path until completion.

Based on the system and our goals for it, these paths are what would be expected. The problems at levels 1 and 2 are basic inference problems, and are designed to be easier to solve for students with the expected requisite knowledge. Level 3 was where the problems were designed to increase in difficulty. Students should not have been able to complete level 3 without showing a higher level of proficiency than had been required up until that point if the problem selection was effective. The fact that most of the class was transferred to the easy path at level 3 indicates that this is the case; students were given problems that were difficult enough to challenge them on the hard path (to the point of being sent to the easy path at the next level) while still being manageable on the easy path.

Since most students did not complete Deep Thought past this point, the paths from level 4 on are somewhat skewed. However, the fact that the students who did complete Deep Thought through level 7 remained on the easy path indicates that the problems in levels 4, 5, and 6 were overall appropriately difficult. These problems were meant to be challenging regardless of the path the student was on, particularly considering that the students did not have requisite knowledge of replacement rules at this point. Therefore the fact that most students stayed on the easy path through level 7 indicates that the problems given were at an expected level of difficulty for them. Conversely, if more students had been able to stay on, or move to the hard path at these levels, it would indicate that either the system is selecting problems that are too easy, or the problems themselves were not designed to be challenging enough. Since the students were continually put on the easy path at these levels, neither of these situations is the case.

4. FUTURE WORK

The data from this initial experiment needs further analysis before any new features are added to the system. However, initial results are promising, and it appears that the system is effective in selecting problem sets for students at a general level. Once this data has been analyzed further and compared to previous data from the old version of Deep Thought, we can make more definite assumptions about the effectiveness of our problem selection.

The next step is to apply the system within levels to test specific problem selection based on rule scores and rule ordering, rather than just problem sets. If that proves effective, we can apply methods in development for automatic generation of problems based on individual rule component construction. Overall, we plan to continue development of Deep Thought into a more effective intelligent tutor in logic proof construction.

5. REFERENCES

[1] Barnes, T. and Stamper, J. 2007. Toward the Extraction of Production Rules for Solving Logic Proofs. In *Proceedings of the 13th International Conference on Artificial Intelligence in Education, Educational Data Mining Workshop (AIED 2007)*, 11-20

[2] Corbett, A.T. and Anderson, J. R. 1994. Knowledge Tracing: Modeling the acquisition of procedural knowledge. User Modeling and User-Adapted Interaction, 3, 253-278

[3] Croy, M., Barnes, T., and Stamper, J. 2008. Towards an Intelligent Tutoring System for Propositional Proof Construction. In Briggle, A., Waelbers, K., and Brey, E. (Eds.) *Current Issues in Computing and Philosophy*, 145-155 IOS Press, Amsterdam, Netherlands

[4] Eagle, M., Johnson, M., and Barnes, T. 2012. Interaction Networks: Generating High Level Hints Based on Network Community Clusterings. In *Proceedings of the 5th International Conference on Educational Data Mining (EDM 2012)*, 164-167

[5] Mostafavi, B., Barnes, T., Croy, M. 2011. Automatic Generation of Proof Problems in Deductive Logic. In *Proceedings of the 4th International Conference on Educational Data Mining (EDM 2011)*, 289-294

[6] Murray, T. 1999. Authoring Intelligent Tutoring Systems. In *International Journal of Artificial Intelligence in Education*, 10, 98-129

Demos/Interactive Events

Demonstration of a Moodle student monitoring web application

Angela Bovo
Andil
Université Toulouse 1
IRIT
angela.bovo@andil.fr

Stéphane Sanchez
Université Toulouse 1
IRIT
sanchez@univ-tlse1.fr

Olivier Héguy
Andil
olivier.heguy@andil.fr

Yves Duthen
Université Toulouse 1
IRIT
yves.duthen@univ-tlse1.fr

ABSTRACT
We would like to demonstrate a web application using data mining and machine learning techniques to monitor students's progress along their e-learning cursus and keep them from falling behind their peers.

Keywords
Moodle, analysis, monitoring, machine learning, statistics

1. AIMS OF THE APPLICATION
We would like to demonstrate a web application developed during a project supported jointly by computer science researchers and an IT firm specialized in e-learning software. Another partner firm, a professional training institute, connects our project with real data from its past and current e-learning courses on various Moodle platforms.

The aim of our application is to use the methods of IT, data mining and machine learning to give educators better tools to help their e-learning students. More specifically, we want to improve the monitoring of students, to automate some of the educators' work, to consolidate all of the data generated by a training, and to examine this data with classical machine learning algorithms. This application is called GIGA, which means Gestionnaire d'indice général d'apprentissage (French for General Learning Index Manager).

The reasons for monitoring students are that we want to keep them from falling behind their peers and giving up, which can be noticed earlier and automatically by data mining methods; we also want to see if we are able to predict their end results at their exams just from their curriculum data, which would mean we could henceforth advise students on how they are doing.

However, currently, Moodle offers very few statistics, and they are hard to examine and analyse. Notably, they are purely individual, so we cannot have a global vision of a group or compare students. We can view a list of logs but hardly anything synthetic, except a few graphs for login data. Only the date of last login and the number of quizzes done were actually used by the training manager we talked to, which seems a waste compared to all the logging done by Moodle. Hence, the need felt for our application.

2. DESCRIPTION OF THE CURRENT IMPLEMENTATION
The web application that we wish to demonstrate is already in use for student monitoring in our partner training institute. This application gathers data from a LMS and other sources and allows to monitor students with raw figures, statistics and machine learning.

Our implementation uses the language Java with frameworks Wicket, Hibernate, Spring and Shiro. The data is stored in a MySQL database.

In our case, the LMS is a Moodle platform where the courses are located. Moodle registers some events in its logging system, which we then import and mine. Hence, we are also constrained by what Moodle does and does not log. For instance, our partner firm had to create a Moodle plugin to have better logout estimates, that will be deployed on future trainings. However, the application could be very simply extended to other LMSes that have a similar logging system.

2.1 Data consolidation
We have decided to consolidate into a single database most of the data produced by an e-learning training. Currently, the data is scattered in two main sources: the students' activity data are stored by the LMS, whereas some other data (administrative, on-site training, contact and communication history, final exams grades) are kept by the training managers, their administrative team and the diverse educa-

tors, sometimes in ill-adapted solutions such as in a spreadsheet. This keeps teachers from making meaningful links: for instance, the student has not logged in this week, but it is actually normal because they called to say they were ill.

We have already provided forms for importing grades obtained in offline exams, presence at on-site trainings and commentaries on students. In the future, we will expand this to an import directly from a spreadsheet, and to other types of data. From Moodle, we regularly import the relevant data: categories, sections, lessons, resources, activities, logs and grades.

2.2 Data granularity

All raw data imported from Moodle or from other sources is directly available for consultation, such as the dates and times of login and logout of each student, or each grade obtained in quizzes.

We then provide statistics built from these raw data, such as the mean number of logins over the selected time period. This is already a level of granularity not provided by Moodle except in rare cases.

We also felt a need for a normalized indicator that would make our statistics easy to understand, like a grade out of 10, to compare students at a glance. We have defined a number of such indicators, trying to capture most aspects of a student's online activity. The features we have selected are: the login frequency, the date of last login, the time spent online, the number of lessons read, the number of lessons downloaded as a PDF to read later, the number of resources attached to a lesson consulted, the number of quizzes, crosswords, assignments, etc. done, the average grade obtained in graded activities, the average last grade obtained, the average best grade obtained, the number of forum topics read, the number of forum topics created, and the number of answers to existing forum topics. For every "number of x" feature, we actually used a formula that would reflect both the distinct and total number of times that this action had been done.

From these indicators, we built by a weighted mean higher level ones representing a facet of learning, like online presence, study, graded activity, social participation and results. Then, at an even higher level but by the same process, a single general grade, which we called the General Learning Index and which gave its name to the application.

2.3 Machine learning

For a more complex output, we use different machine learning methods to analyse the data more in depth and interpret it semantically [1]. We use classical clustering and classification algorithms, in their implementation by the free library Weka.

We provide the following algorithms: for clustering, Expectation Maximisation, Hierarchical Clustering, Simple K-Means, and X-Means; for classification, Logistic Regression, LinearRegression, Naive Bayes and Multilayer Perceptron. They can be used with or without cross-validation, and the random seed and number of folds can be manually selected. For clustering algorithms where the number of clusters is not decided by the algorithm, we allow to select a fixed number of clusters.

We use our indicators listed in §2.2 as features for learning, and for the classification algorithms, we use the mean grade obtained at the final exams as the class feature. As an output, we obtain groups of students. In the case of clustering, we have to look at their indicators to understand the meaning of these groups. With classification, we can try to see which indicators can predict the final grade.

3. FUTURE WORK

We have already thought of new features that we would like to implement.

We want to compare the results obtained by all machine learning algorithms to see if one seems better suited. Later, we will also implement another HTM-based machine learning algorithm, and again compare results. We also want to add regression to try and predict the final grade.

Another facet that the data we have gathered could reveal is the quality of the study material: is a quiz too hard, so that students systematically fail it? Is a lesson less read than the others - maybe it is boring? Do the students feel the need to ask many questions in the forums?

We could partly automate the training managers' work by creating an intelligence virtual tutor that will directly interact with students and teachers. It could suggest students a next action based on their last activity and graded results, or also give them a more global view of where they stand by using the machine learning results. It could also send them e-mails to advise them to login more frequently or warn them that a new activity has opened. It could also warn the training manager of any important or unusual event.

4. CONCLUSION

This application uses data mining and machine learning methods to solve the problem of student monitoring in e-learning. We have detailed how the implementation allows to meet our goals by a good mix of different levels of granularity in the viewing of the data (raw data, statistics and data processed by different clustering and classification machine learning algorithms). Such a tool is very appreciated by our first users and is very innovative.

To do this, we have had to define indicators that serve both for statistics and for machine learning features. These indicators are both relevant to our project and generic enough to be of use for the community.

We will also present a poster in this conference to describe some preliminary clustering results obtained using this application.

5. REFERENCES

[1] C. Romero, S. Ventura, and E. García. Data mining in course management systems: Moodle case study and tutorial. *Computers and Education*, pages 368–384, 2008.

Students Activity Visualization Tool

Marius Ștefan Chirițoiu
University of Craiova
Bvd. Decebal, 107
+40-251 438198
cmariusstefan@gmail.com

Marian Cristian Mihăescu
University of Craiova
Bvd. Decebal, 107
+40-251 438198
mihaescu@software.ucv.ro

Dumitru Dan Burdescu
University of Craiova
Bvd. Decebal, 107
+40-251 438198
burdescu@software.ucv.ro

ABSTRACT
One of the primary concerns in on-line educational environments is the effective and intuitive visualization of the activities performed by students. This paper presents a tool that is mainly designed for the use of professors in order of assist them in a better monitoring of students activity. The tool presents in an intuitive and graphical way the activities performed by students. The graphical presentation creates a mental model of the performed activities from the perspective of former generations of students that followed the same activities. The tool integrates k-means clustering algorithm for grouping students and for facilitating the customization of parameters and number of clusters that are displayed.

Keywords
activity visualization, k-means, e-Learning, PCA.

1. INTRODUCTION
One of the main issues of on-line educational environments is the effective and intuitive visualization of activities performed by students. For an e-Learning platform which has an average of more than 100 students per module it may become quite difficult for the professor to visualize the on-line activity performed by each student at a time or by all students at one time. The paper presents a tool that improves the productivity of a professor by providing an effective way of visualizing and interacting with the students.

This paper presents a tool that displays in a graphical format the activities performed by students. There are several characteristics of the tool that make it user friendly and very efficient in presenting in synthetic form the results. The main characteristic consists of the fact that the display is in 2D (2-dimensional) space and for each coordinate a single parameter is used. The display presents a specified number of groupings according with the settings of the professor. The number of groupings (i.e., clusters) represents the main parameter for the clustering algorithm that effectively places the students into clusters. For each cluster there is clearly presented the centroid. The students from the same cluster are presented with the same distinct geometric sign in color and shape. Each cluster is divided into three areas: center, close area and far area. Each area gathers students that have the same behavioral pattern regarding the activities performed within the on-line educational environment.

2. RELATED WORKS
There are several examples of educational data mining tools. The latest research trends place an important emphasis on developing tools that successfully integrate and prove the latest findings in the domain. In [1] there is presented a section with the more than twenty educational data mining tools. Among them, the ones that are more closely to the tool presented in this paper is GISMO [2], EDM Visualization Tool [3] or SNAPP [4].

GISMO is a tool whose main purpose is to visualize what is happening in distance learning classes. EDM Visualization Tool is mainly designed to visualize the process in which students solve procedural problems in logic. The SNAPP tool may be used to visualize the evolution of participant relationships within discussions forums.

3. SOFTWARE ARCHITECTURE
The application is divided into packages that contain classes that implement related functionalities. The main classes that perform the business logic of the tool are *ClusteringServlet*, *RunScheduledJobServlet*, *BuildArffFileScheduledJob*, *BulidClusterersScheduledJob*, *KMeansClustererStart*, *ClientStervlet* and *ClustererClientApplet*.

The web server administration interface (*index.html*) allows building a number of clusters and viewing them. This is performed by the *ClusteringServlet* which in turn uses the *KMeansClustererStart* class. *KMeansClustererStart* class generates the clusters (i.e., the model) based on ARFF file using KMeans algorithm. It also contains various methods for manipulating clusters data. In this class PCA (Principal Component Analysis) algorithm is used to reduce the dimensionality (number of attributes) of a given dataset.

The *RunScheduledJobsServlet* class is a servlet that starts at server startup and runs the scheduled job at specified time. The time and frequency are specified in a *xml* configuration file. The scheduled jobs are represented by *BuildArffFileScheduledJob* and *BuildClustersScheduledJob* classes. These, as their name implies, deal with building the *arff* file from database and building clusters from *arff*. The *BuildArffFileScheduledJob* class uses *ArffGenerator* class for building the *arff* file containing the training dataset.

On the client side we have a java applet that runs in an Internet browser. It connects to server, takes data needed and displays the students grouped according to certain features chosen by the professor. The interface of the client application allows specifying data for a new student and viewing its position on the chart (in the cluster to which it belongs).

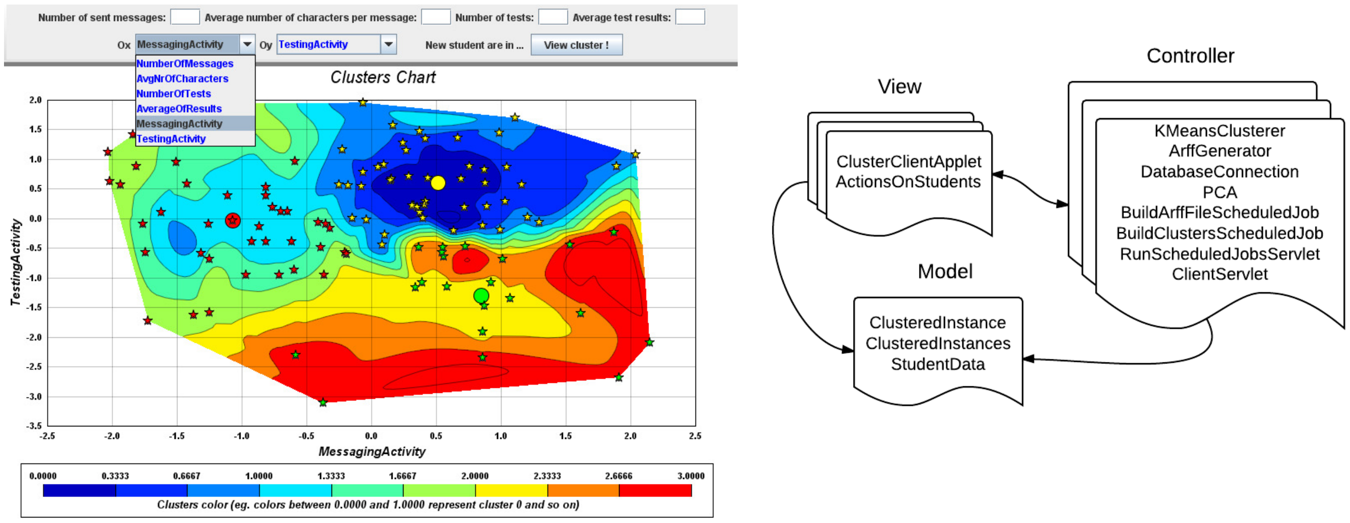


Figure 1. GUI of the visualization tool & Soft Architecture

4. SOFTWARE TOOL

Figure 1 presents the GUI of the visualization tool. In this running there are three clusters of students built according to the two features of the axes (i.e., testing activity and messaging activity). The application allows the professor to select other features by which to build the clusters. When you keep the mouse pointer over a point on the graph you can see the features values for that student. The points representing the students have different colors for each cluster. For a better visualization each cluster is divided into three areas: center, middle area and far area. Each area is colored differently giving thus intuitive information regarding how close from the centroids are the points belonging to a cluster.

The tool uses a total of six features with which we can build clusters as presented in figure 1. The last two features are composed features resulting by combining two simple features using Principal Component Analysis (PCA). The *MessagingActivity* feature is computed as a combination between the *NumberOfMessages* feature and *AvgNrOfCharacters* feature. In the same way, the *TestingActivity* feature is computed as a combination between *NumberOfTests* and *AverageOfResults* feature.

If features values are provided for a new student the tool places a big X mark with the same color as other instances from the same cluster in corresponding position and thus the cluster is immediately determined. After viewing clustered students, the teacher can select one or more students from the chart and save their data in a PDF or send an e-mail to them.

The tool may be used for two purposes. One regards easy visualization of the student's activity based on different criteria. Once the visual information is retrieved, the tool may be used to interact (i.e., send messages) with a specific set of students that may be easily selected. The tool may be also successfully used for outlier detection, which in our case is represented by students that hardly can be assigned to a cluster.

5. CONCLUSIONS

This paper presents a visualization tool based on a clustering algorithm. The tool presents the clusters of students in a very intuitive way. The clustered students may be selected and a set of specific actions may be performed: sending messages, export to pdf, etc. In future, the tool may be extended by integrating other features for data representation and by providing other advanced functionalities for professors.

6. REFERENCES

[1] Cristobal Romero and Sebastian Ventura, Data mining in education, WIREs Data Mining Knowl Discov 2013, 3: 12–27 doi: 10.1002/widm.1075.

[2] Mazza R, Milani C. GISMO: a graphical interactive student monitoring tool for course management systems. In: International Conference on Technology Enhanced Learning. Milan, Italy; 2004, 1–8.

[3] Johnson M, Barnes T. EDM visualization tool: watching students learn. In: Third International Conference on Educational Data Mining. Pittsburgh, PA; 2010, 297–298.

[4] Bakharia A, Dawson S. SNAPP: a bird's-eye view of temporal participant interaction. In: Proceedings of the 1st International Conference on Learning Analytics and Knowledge. Vancouver, British Columbia, Canada; 2011, 168–173.

FlexCCT: Software for Continuous CCT

Stephen L. France
University of Wisconsin - Milwaukee
Sheldon B. Lubar School of Business
P. O. Box 742
1 414 229 4596
france@uwm.edu

Mahyar Vaghefi
University of Wisconsin - Milwaukee
Sheldon B. Lubar School of Business
P. O. Box 742
1 414 229 4235
mahyar@uwm.edu

William H. Batchelder
University of California, Irvine
2129 Social Sciences Plaza A
Mail Code: 5100
1 949 824 7271
whbatche@uci.edu

ABSTRACT

In this paper, we describe a software package called FlexCCT for analyzing numerical ratings data. FlexCCT is implemented in MATLAB and incorporates a range of different cultural consensus theory (CCT) models. We describe the standalone GUI version of FlexCCT. We give an illustrative example, showing how FlexCCT can be used to analyze and interpret essay rating data.

Keywords

CCT, maximum likelihood, optimization, essay rating

1. INTRODUCTION

Cultural consensus theory (CCT) is a methodology used to analyze cultural values or “truth”. CCT has several educational data analysis/data mining applications. CCT can be used to analyze educational essay/question ratings data to i) evaluate rater competency, ii) evaluate rater bias, iii) calculate accurate competency weighted ratings, and iv) evaluate the easiness/difficulty of rating individual answers. The CCT results can be used to evaluate a set of essay ratings and then recommend actions, for example retraining certain raters or using rater competency to determine the number of raters assigned to rating tasks. CCT can also be used as part of the rating/grading process; for example, the item easiness CCT models can be used to assign additional raters to answers that are deemed difficult to rate.

We present FlexCCT, a software package for implementing maximum likelihood CCT. We do not give axiomatic or mathematical descriptions of the class of CCT models described in this paper. These can be found in [1,3,4]. We give an intuitive description of several of the CCT models implemented in the FlexCCT software, concentrating on CCT models for continuous data. We summarize the model features and describe the software implementation of the models. We then describe work that uses CCT to analyze essay grading data.

2. THE CCT MODELS

Consider a situation where there are n subjects or raters. Each rater assigns a score to each of m questions. Each question could be a quality rating, an estimation of quantity, or any other type of question that may elicit a numerical response. There is no a-priori known correct answer to any of the questions, which is why CCT has taken the moniker of “test theory without an answer key” [2].

Let $\mathbf{X}$ be an n rater × m item matrix of item ratings or scores. A simple method of calculating a 1 × m latent answer vector $\mathbf{z}$ would be to calculate the average score for each item across all raters. However, this ignores the fact that some raters may be more competent than other raters and that some raters may be “biased” to giving lower or higher scores. The basic CCT models are based upon a set of axioms [1,2,3,4]. In this paper, we do not describe these axioms formally, but we give some basic intuition.

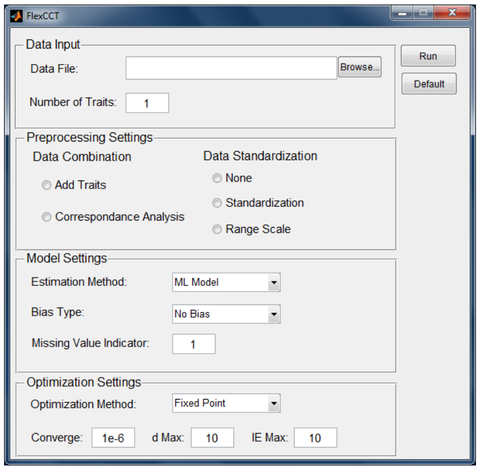
User competency is defined as a measure of inverse error variance. For each rater i, the rater competence $d_i \in \mathbb{R}^+$ is defined as the inverse error variance, so that $d_i^{-1} = \sigma^2(\varepsilon_{ik})$. The maximum likelihood function for the basic CCT model is given in (1).

$$L(\mathbf{d},\mathbf{z}|\mathbf{X}) = \prod_{k=1}^{m}\prod_{i=1}^{n}\sqrt{(d_i/2\pi)}\,e^{\left(-d_i(x_{ik}-z_k)^2/2\right)} \qquad (1)$$

Bias operationalizes the tendency of raters to consistently rate either lower or higher than the latent answer key values. Either additive bias or multiplicative bias can be incorporated into the basic model. For additive bias, the inner likelihood term $ILT = (x_{ik} - z_k)$ is replaced by $(x_{ik} - b_i - z_k)$ and for multiplicative bias it is replaced by $(x_{ik} - b_i z_k)$. Only one type of bias can be included in the model, as including both additive and multiplicative biases over parameterizes the model [4]. Three models for “item easiness”, described in [4], are incorporated into FlexCCT. The first model is an error variance model, where $\sigma^2(\varepsilon_{ik})$ is split into rater components and answer components. The second model incorporates a multiplicative easiness scaling factor, so that for each combination of rater i and item j, the competency is scaled by an easiness parameter β_j, so that d_i is replaced by $d_i\beta_j$. The third model is similar to the second model, except that the easiness parameter is additive, so that d_i is replaced by $d_i + \beta_j$.

3. SOFTWARE DESCRIPTION

FlexCCT consists of a set of MATLAB functions and a compiled, standalone GUI version of the software. The GUI consists of a single input screen, where the user configures the software parameters and an output screen, which displays results from the CCT model optimization. The output screen has an option to save the output parameter values. The input screen is given in Figure 1 and a description of the associated options is given in Table 1.



Data File:
Number of Traits: 1
Estimation Method: ML Model
Bias Type: No Bias
Missing Value Indicator: 1
Optimization Method: Fixed Point
Converge: 1e-6 d Max: 10 IE Max: 10

Figure 1: Input Screen

Table 1. Input Parameters

Name	Options
Data File	A csv file containing the values of **X**. Rows correspond to raters and columns correspond to items. The data should not contain a header.
Data Combination	**Add Traits:** If more than one trait/attribute then add the trait values together. **Correspondence analysis:** Calculates a single continuous trait from multiple qualitative traits.
Data Standard-ization	**None:** Use raw data. **Standardize:** Subtract column μ and divide by column σ. **Range Scale:** Divide by column range.
Estimation Method	**Simple Average**: $d_i = 1$ for all raters and z_k is the average value of $\mathbf{x}_{\cdot k}$. **Factor Analysis:** Utilizes a minimum residual factor analysis as per classical CCT [5]. **ML Model:** Basic maximum likelihood model from (1). **IE Error Variance:** Item easiness error variance model. **IE Multiply:** Item easiness where $d_{ij} = d_i\beta_j$. **IE Add:** Item easiness where $d_{ij} = d_i + \beta_j$.
Bias Type	**No Bias:** (see section 2). **Additive Bias:** (see section 2). **Multiplicative Bias:** (see section 2).
Optimization Method	**Fixed point:** Fixed point estimation. **Two Stage Fixed Point**. The values of **z** and **d** are estimated first, followed by other parameters. **Derivative Free**: Standard MATLAB routine. **Gradient**: MATLAB Gradient descent optimization, utilizing first order derivatives
Converge	Converge criteria for optimization, (default = 1e-6).
MissingVal	Indicator for a missing value, defaults to -1
Max d	Upper bound for **d**. Prevents a single rater having dominant competency (default = 10)
Max IE	Upper bound for item easiness **β** (default = 10).

When the "Run" button is pressed and the model optimization is completed, the user is presented with an output screen, which displays a summary of the model output. This summary includes the maximum log-likelihood, the algorithm run time, and values for all of the output parameters. The output button allows users to save the output parameter values to a csv file. In the csv file, each set of parameters (e.g. **z**, **d**, **b**, **β**) is assigned to a column in the file. Row vectors are transposed. The file output parameters are summarized in Table 2.

Table 2. Output Parameters

Name	Options
z	An $1 \times m$ latent answer vector.
d	An $n \times 1$ vector of competencies.
b	An $n \times 1$ vector of biases.
β	An $1 \times m$ vector of item easiness parameters
pll (1-3)	Three $1 \times m$ vectors of partial log likelihoods corresponding to values calculated with floor(**z**), ,**z** , and floor(**z**)+1.

4. EDUCATIONAL EXAMPLE

In [4], a detailed example is given to show how CCT can be used in essay (or more general) rating applications. A subset of 50 essays was taken from a set of high school essays. The prompt for the essays was to describe a situation involving laughter. A grading rubric was defined and each essay was graded on 6 attributes, with each attribute having a range from 1-6. An overall continuous score was calculated using two approaches. In the first approach, the assumptions of classical test theory were used and the scores for each attribute were added together to give a total score in the range of 6–36. In the second approach, multiple correspondence analysis was used to explicitly scale the multiple ordinal attribute scales into one continuous scale. The essays were graded by 2 expert graders and 10 student graders. Each student grader was given 30 minutes for training and 3 minutes to grade each essay.

Some overall conclusions reached in [4] are that CCT provides useful measures of rater competency, rater bias, and item easiness/difficulty. CCT can be used to help train and evaluate raters and to identify essays where accurate evaluation is difficult. The CCT competencies can be used to produce competency weighted averages of essay ratings. In the essay rating data analyis, incorporating bias gave improved model fit and additive bias gave better model fit then multiplicative bias. Likewise, the multiplicative item easiness model gave better model fit than the additive item easiness model.

5. CONCLUSIONS AND FUTURE WORK

The current version of FlexCCT (1.0.0) provides a flexible framework for implementing CCT and can be used to analyze a wide range of ratings/questionnaire data. FlexCCT is implemented as a set of MATLAB functions. There is a standalone GUI version of the software, which does not require a MATLAB license and provides a wrapper for a set of continuous CCT models. For future versions of FlexCCT, we plan to incorporate clusterwise CCT, which simultaneously assigns raters to clusters/cultures and calculates the CCT model for each culture.

6. ACKNOWLEDGMENTS

The third author acknowledges the support of a grant from the Army Research Office (ARO) and a fellowship from the Oak Ridge Institute for Science and Education (ORISE).

7. REFERENCES

[1] Batchelder, W. H. and Romney, A. K. New results in test theory without an answer key. In Roskam, E. E. ed. *Mathematical Psychology in Progress*. Springer-Verlag, Heidelberg, Germany, 1989, 229-248.

[2] Batchelder, W. and Romney, A. 1988. Test theory without an answer key. *Psychometrika*, 53, 1 (Mar. 1988), 71-92.

[3] France, S. L. and Batchelder, W. H. 2012. *Unsupervised Consensus Analysis for On-line Review and Questionnaire Data*. Working paper, UC Irvine.

[4] France, S.L. and Batchelder, W. H. 2013. *A Maximum Likelihood Item Difficulty Model for Consensus Analysis*. Working paper, UC Irvine.

[5] Romney, A. K., Weller, S. C. and Batchelder, W. H. 1986. Culture as Consensus: A Theory of Culture and Informant Accuracy. *American Anthropologist*, 88, 2 (Oct. 1986), 313-338

Visual Exploration of Interactions and Performance with LeMo

Agathe Merceron,
Sebastian Schwarzrock
Beuth University of Applied Sciences
Amrumer Straße 10
13353 Berlin, Germany
{merceron, sschwarzrock}@beuth-hochschule.de

Margarita Elkina,
Andreas Pursian
Hochschule für Wirtschaft und Recht
Alt Friedrichsfelde 60
10314 Berlin, Germany
{margarita.elkina, andreas.pursian}@hwr-berlin.de

Liane Beuster,
Albrecht Fortenbacher,
Leonard Kappe, Boris Wenzlaff
Hochschule für Technik und Wirtschaft
Wilhelminenhofstraße 75
12459 Berlin, Germany
{liane.beuster, albrecht.fortenbacher, kappe, boris.wenzlaff}@htw-berlin.de

ABSTRACT

The demo will show how the LeMo (**Le**rnprozess**Mo**nitoring) tool supports teachers to explore visually how students interact with learning resources and how they perform. The information obtained in this visual exploration guides subsequent more involved analyses.

Keywords

Learning Management System, Visualization, Interaction, Performance, Visual Analytics.

1. INTRODUCTION

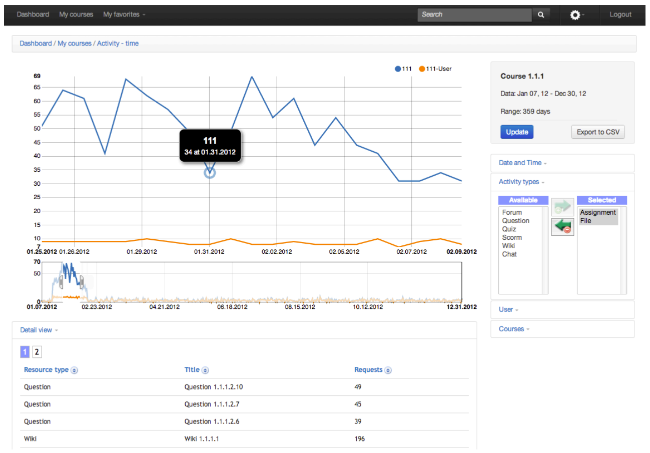
The use of a Learning Management System (LMS) to support teaching and learning is widespread. The usage data such systems store is not analyzed in a *routine basis* by different stakeholders to retrieve pedagogical information that could support reflection. For example, if a teacher notices that some non-compulsory exercise she has made available in her course is hardly attempted, she might do some further analysis: does it seem to have a positive impact on the mark of the final exam for the few students who solved it? If not, she might consider deleting it from the course for the next semester; if yes, she might change her teaching style so that more students attempt it.

The aim of LeMo is to support different stakeholders in their analysis of usage data stored by LMS or learning portals [1]. All what is needed is a module that exports the data stored by the LMS into the data model of LeMo. Presently 3 export modules exist for the LMS Moodle and Clix and for the learning portal ChemgaPedia. The current prototype focuses on teachers as stakeholders. It aims at supporting them to explore whether and how their students interact and succeed with the resources they have made available online. In order for teachers to grasp at a glance what happens and so to ease the integration of such a tool in their practice, exploration is primarily carried out through various interactive visualizations that follow the “overview, zoom and filter” principle [2]. A few visualizations are presented below. Attendees at the conference will have the opportunity to try out the tool by themselves.

2. VISUALIZING INTERACTIONS

The aim of Figure 1 and 2 is to help answering questions such as: do students access them, when, in which order?.

Figure 1 shows accesses on all resources of a course over time. The upper line gives the total number of accesses, while the lower line gives the number of distinct students. The diagram is interactive. Placing the cursor over the line will produce a tool tip that shows the exact number and point in time. To allow both the overview of a selected time-period and the focus on detail, the user can pick out a certain time slot from the lower diagram to get an amplified view in the main diagram above. The user can deactivate and activate any line to concentrate on only one if needed. Below the diagrams all learning-objects of the course are listed in a table. The columns show the type of the learning-object, title and the number of accesses for it. Each column can be sorted by clicking on the title. This enables for example to view the most-accessed and least-accessed learning-objects. Filters allow for selecting particular subsets of the data. A filter allows for choosing another time period. A second one filters students, and a third one filters learning objects according to their type. A type can be ‘forum’, ‘assignment’, ‘wiki’, ‘file’ a generic term to design objects such as slides, and so on.



Course 1.1.1
Data: Jan 07, 12 - Dec 30, 12
Range: 359 days

Resource type	Title	Requests
Question	Question 1.1.1.2.10	49
Question	Question 1.1.1.2.7	45
Question	Question 1.1.1.2.6	39
Wiki	Wiki 1.1.1.1	196

Figure 1. Access to learning objects.

Figure 1 shows this filter with the types assignment and file selected. All available visualizations have these three filters. Figure 2 shows the network of learning objects that results when order of access or navigation is taken into account. Learning objects are nodes colored according to their type. The size of a node is proportional to the number of accesses. Navigational steps between learning-objects are depicted by edges. The edges are weighted and color-coded to encode the amount of navigational steps. Placing the cursor over a node will bring up a tool-tip that includes information concerning the learning-object's name, the type as well as the total number of accesses or requests for it. For further exploration, a single click on a specific node will rearrange the graph in a way that it focuses on the node of interest, displaying neighboring nodes in the immediate proximity, and moves other nodes further away.

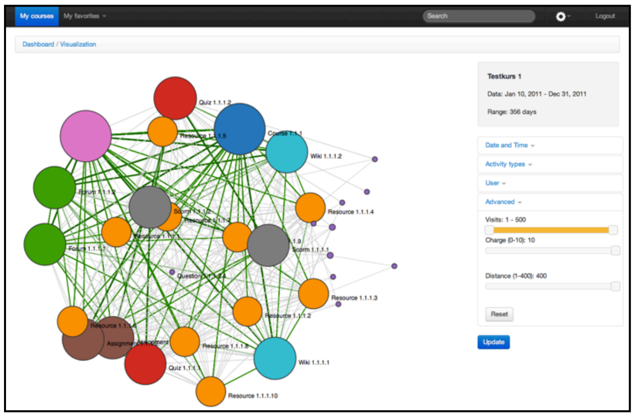


Figure 2. Network of learning objects according to navigation.

3. VISUALIZING PERFORMANCE

An assignment is a generic term meaning any work that can be graded, such as questions, exercises, tests, exams and so on. Most of the LMS allows for calculating easily useful statistics such as average and standard deviation for a given assignment. It is more difficult to visualize and compare performance of students across several or all assignments, a question raised by teachers. The visualizations presented here cater for this need. It is not rare that different assignments are marked differently. For example assignment 1 may be out of 20 points and assignment 2 out of 50. Sticking to the original scale given by teachers makes a comparison awkward. Therefore in the following visualizations all assignments are scaled to 100. The usual filters mentioned earlier can be used to select particular assignments or tests, or to select particular users. Figure 3 and 4 shows two visualizations to explore performance concentrating on the diagrams.

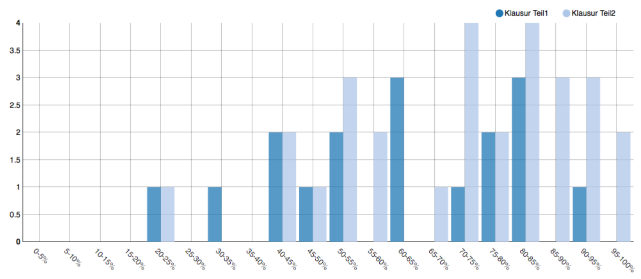


Figure 3. Comparing performance with marks' distribution.

The histogram Figure 3 gives an overview of the distribution of the students according to their marks. We can see that two students are in the highest interval [95 -100] for the second part of the exam, called Klausur-Teil2, while no student has achieved this high performance in the first part of the exam, Klausur-Teil1. Again, the visualization is interactive. Cursor over a bar shows the exact number. The user can activate or deactivate any test or assignment by clicking on the circle near the name. A hollow circle visualizes that an assignment has been deselected. Appearance or disappearance of bars after activation, respectively deactivation, occurs progressively, so that the user can follow the change taking place.

A second visualization shows the box plots for all assignments. In Figure 4 the same assignments as in Figure 3 have been selected. It is possible to grasp not only that the maximum mark is higher in the second part of the exam, but also that over 75% of the students have done better, since the whole box including median is higher.

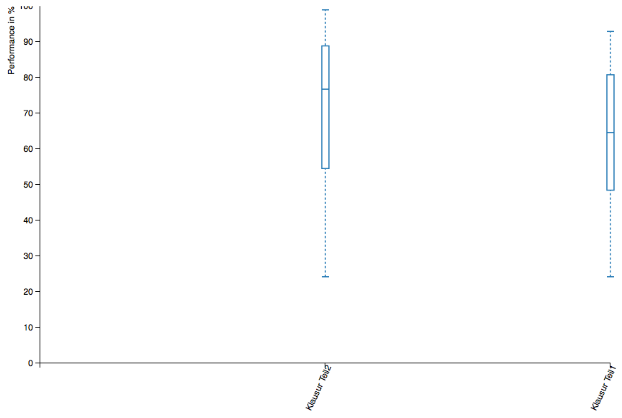


Figure 4. Comparing performance with box plots.

4. CONCLUSION

This visual exploration is the first step for analyzing usage data. When teachers grasp the overall trends in interactions and performance in their course, they should be able to deepen their analysis, seeking answers for questions such as: can students be grouped according to their performance? Or, dually, can assignments be grouped according to how students perform? Future work includes implementing means to help answering similar more involved questions. A challenge is to select the data mining algorithms and their parameters carefully so as to avoid misinterpretation of the results by stakeholders.

5. ACKNOWLEDGMENTS

This work is partially supported by the "Berlin Senatsverwaltung für Wirtschaft, Technologie und Forschung" with funding from the European Social Fund.

6. REFERENCES

[1] Beuster, E., Elkina, M., Fortenbacher, A., Kappe, L., Merceron, A., Schwarzrock, S., Wenzlaff, B. 2012. LeMo-Lernprozessmonitoring auf personalisierenden und nicht personalisierenden Lernplattformen. In *Proceedings of the GML^2 Grundfragen des Multimedialen Lehrens and Lernens Conference* (Berlin, Germany, March 15-16, 2012) . Waxmann Verlag, 63-76. http://www.gml-2012.de/tagungsband/Tagungsband_GML2012_web.pdf

[2] Shneiderman, B. 1996. The eyes have it: A task by data type taxonomy for information visualizations. In: *Proc. of the IEEE Symposium on Visual Languages*, (Maryland Univ., College Park, MD, USA, September 03 – 06, 1996). IEEE, 336-343

Project CASSI: A Social-Graph Based Tool for Classroom Behavior Analysis and Optimization

Robert Olson, Zachary Daily, John Malayny, Robert Szkutak
Department of Computer & Information Sciences
State University of New York at Fredonia
Fredonia, NY 14063
olsonr@fredonia.edu, dail9583@fredonia.edu, mala3598@fredonia.edu, szku7989@fredonia.edu

ABSTRACT

Although educational data mining is a well-established field, it has not yet sought to provide serious, actionable intelligence that can be used by teachers to address bullying in a reasonable amount of time. This paper seeks to propose a system that will streamline the processing and storage of bullying data in social graph form so that it will be available to be mined by expert systems that can help educators in the classroom. In addition, one such expert system will be proposed demonstrating how this data may be used to automate a common classroom management task that may improve students' classroom experiences.

Keywords

Behavior modeling, Implicit social graphs, Classroom behavior optimization, Seating chart generation

1. INTRODUCTION

Although bullying has long been a significant problem, increased awareness has brought the matter to the attention of legislatures. States across the country are passing laws aimed at preventing bullying. Some pieces of legislation, like New York's *Dignity for All Students Act*, include provisions for information sharing and increased data retention [3], creating an environment ripe for innovation in the fight against bullying.

In this paper, we will propose CASSI (Classroom Assisting Social Systems Intelligence), an open-source system aimed at being inexpensive and easily integrated into existing educational practices that will allow for the collection, modeling, and analysis of student behavioral data. Specifically, the collected behavioral data will be used to construct a social graph that represents how dysfunctional the directed relationship between each pair of students is. This social graph can then be used to inform the behavior of a variety of expert systems.

It is hoped that this system, in due time, will be implemented in educational institutions to adhere to both the letter and the spirit of new pieces of anti-bullying legislation. A single central repository for an educational institutions behavioral data would allow for the data to be more easily shared amongst educators and formatted into reports for administrators. This repository of data would also allow for the implementation of expert systems which educators can use in day-to-day classroom management tasks that influence bullying [1]. One expert system will be described that makes use of the data stored in the social graph to improve classroom management by allowing teachers to automatically generate seating charts likely to reduce negative classroom behavior. Other possible expert systems will also briefly be discussed.

2. BACKGROUND

Romero and Ventura[7] provide a detailed overview of early educational data mining projects. The authors describe a wealth of educational data mining projects aimed at improving pedagogy. However, there is a lack of data mining projects intended to improve students' educational experiences through means other than pedagogy. Romero and Ventura also note that early data mining tools do seem to be designed with data mining experts, not educators in mind. Their implication seems to be that, if provided with the right tools, educators would be able to make better use of the information gathered from data mining.

Dawson[2] recounts a study utilizing social network analysis to draw some conclusions about how a student's position in a social network influences the student's perceptions regarding the sense of community they experience. Although this is a narrow application of social network analysis in the classroom, it provides an excellent justification for using social network analysis as a means of evaluating classroom behavior. While this study doesn't describe a system easily used by educators, one can easily imagine how such a system could be extended to provide a more detailed level of analysis by integrating more observations containing the sort of behavioral data that Hung and Lockard [4] used to create their Behavioral Matrix software.

There has been a recent push for software to address cyber-bullying through the analysis of social networking sites. Nahar, Unankard, Li, and Pang [5] describes a method for using sentiment analysis to develop a graph-based method for identifying cyber bullies and their victims while Sanchez and Kumar [1] describes a method for integrating this style of sentiment analysis with Twitter. Although cyber-bullying is a significant problem, neither of these papers address problems associated with completeness. It is easy for students to restrict educators' access to their social media accounts and such restrictions may skew the decisions of an expert system which integrates social media data with observed classroom data. However, both of these methods appear to accurately recognize the asymmetric nature of bullying discussed by Allen [1].

It also is important to note the distinction between social networks and social networking. As noted in Purtell et al. [6], it is possible to extract implicit or inferred social topographies [8] from other sources other than social media.

3. DATA MODEL AND VISUALIZATION

One of the strengths of CASSI is that much of the data that the system collects is likely to already be collected as a part of routine classroom management practices. At the moment, the only data the system requires educators to record is: the victim, the bully, and a ranking of the bullying incident on a score from 1 to 10. It is expected the system will also record additional book-keeping data such as a time-stamp, the educator filing the incident report, and a detailed description of the even that may be utilized in the future. This data, collected via a web-form when an incident occurs and stored in a local database, is similar to the directed data described by Purtell et al. [6] as suitable for use in the construction implicit social graphs of the type described in Roth et al. [8].

There are some flaws in this data collection mechanism. The incident score necessitates the development of an objective standard against which incident seriousness may be compared to ensure consistency. Another flaw is that bullying often takes place in spaces not observable by educators. This may be addressed in the future through mechanisms such as student-driven incident reporting tools. However, this flaw may also be addressed by educator training aimed at expanding definitions of bulling to include include what Allen [1] refers to as “relational bullying.”

Once the data is collected via web-form, it is processed to form the social graph. This amounts to populating a matrix *S*, indexed by student, of ordered pairs (*T, V)* where *T* for $S_{i,j}$ represents the sum of the seriousness ratings from incidents where student *i* is engaging in bullying behavior targeting student *j*. *V*, meanwhile, represents the number of incidents added to produce *T*.

This two dimensional information can be easily analyzed. For example, the student relationships furthest from the origin should be red flagged as those most in need of immediate intervention. Relationships where *T* / *V* is high while *V* is low may indicate an emerging bully. Finally, the case where the Euclidean distance between points Si,j and Sj,i is low but the distance from both of these points to the origin is high may represent a rivalry in need of serious intervention.

4. CLASSROOM OPTIMIZATION

One expert system that has been developed to make use of the behavioral social graph aims at automating the task of finding behaviorally optimal seating charts for rectangular seating arrangements of an arbitrary size. This task is accomplished by comparing the social graph to a particular arrangement of students. If two students are seated adjacent to one another in the classroom, their relationship information is extracted from the social graph and added to the ranking of the classroom. This reduces the task of finding an optimal classroom down to a simple minimization problem – the lower the ranking, the better the classroom.

Iterating through all possible arrangements of students guarantees that the best seating arrangements is found. However, this is extremely computationally expensive at *O(N!)* for classrooms with asymmetric physical properties that may influence behavior, such as windows that might provide a distraction.

Fortunately, it is also relatively easy to develop a heuristic that exploits the physical properties of the classroom to find good, but not necessarily optimal, arrangements. Specifically, this heuristic exploits the number of adjacencies that each seat has. Corner seats only have three direct adjacencies in a rectangular classroom, for example, making them ideal locations for the students most likely to disrupt others. Using these physical classroom properties and sorting the student in order of the probability of causing a disruption reduces the time to *O(N*).

There was some concern that the formation of networks of friendships along racial, gender, academic performance, or economic class boundaries may cause the seating charts generated this way to, unintentionally, segregate classrooms. This was addressed by scoring the classroom using a weighted average of the social-graph adjacencies within the arrangement of students and the number of homogenous demographic adjacencies within the arrangement of students.

5. CONCLUSION AND FUTURE WORK

Project CASSI is a tool that should allow educators to share behavioral information more easily and serve as the foundation on which useful classroom-management expert systems may be built. However, Project CASSI is in its infancy. Although CASSI has been tested extensively on simulated data, it is most immediately in need of testing with authentic behavioral data.

Once the system has been tested on genuine behavioral data, there are a number of additional expert system modules that may extend its usefulness. In particular, the researchers of Project CASSI expect that the system may be extended to support behaviorally informed scheduling and time-series forecasting. The former task – grouping students into non-disruptive classes – may be accomplished a similar selection algorithm to the one proposed for seating chart generation but operating on combinations of students rather than permutations of students. The later task, time-series forecasting using the time-stamps of the incident report, may be conducted with the ultimate goal of predicting bullying trends and predicting bullying events before they actually occur.

6. REFERNECES

[1] Allen, K. P. (2010). Classroom Management, Bullying, and Teacher Practices. *Professional Educator, 34*(1), 1-15.

[2] Dawson, S. (2008). A study of the relationship between student social networks and sense of community. *Educational Technology & Society, 11*(3), 224-238.

[3] Dignity for All Students Act. 2010. N.Y. Educ. Law §§ 10-18

[4] Hung, W.-C., & Lockard, J. (2007). Using an Advance Organizer Guided Behavior Matrix to Support Teachers' Problem Solving in Behavior Classrom Management. *Journal of Special Education Technology, 22*(1), 21-36.

[5] Nahar, V., Unankard, S., Li, X., & Pang, C. (2012). Sentiment Analysis for Effective Detection. In *Web Technologies and Applications* (Vol. 7235, pp. 767-774). Springer Berlin Heidelberg.

[6] Purtell, T. J., McLean, D., Teh, S. K., Hangal, S., Lam, M. S., & Heer, J. (2011). An Algorithm and Analysis of Social Topologies from Email and Photo Tags. *Workshop on Social Network Mining & Analysis, ACM KDD.* San Diego, CA.

[7] Romero, C., & Ventura, S. (2007). Educational Data Mining: A survey from 1995 to 2005. *Expert Systems with Applications, 33*, 135-146.

[8] Roth, M., Ben-David, A., Deutscher, D., Flysher, G., Horn, I., Leichtberg, A., et al. (2010). Suggesting Friends Using the Implicit Social Graph. *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining* , (pp. 233-242).

[9] Sanchez, H., & Kumar, S. (2011). *Twitter Bullying Detection.* UCSC ISM245 Data Mining course report.

A Moodle Block for Selecting, Visualizing and Mining Students' Usage Data

C.Romero, C.Castro, S.Ventura
Department of Computer Science
University of Cordoba, Spain
cromero@uco.es, i82cavic@uco.es, sventura@uco.es

ABSTRACT

This paper describes a tool that enables instructors to select, visualize and mine students' usage data in Moodle courses. The tool has been developed in PHP language and integrated in Moodle as a block.

Keywords

Student usage data, Moodle block, data visualization, data mining.

1. INTRODUCTION

Nowadays, there are a great number of general free and commercial DM tools and frameworks [2], such as: Weka, RapidMiner, KNIME, R, SAS Entreprise Miner, Oracle Data Mining, etc. These tools can be used for mining datasets from any domain or research area. However, none of these tools is specifically designed for pedagogical/educational purposes and problems. So they are cumbersome for an educator to use since they are designed more for power and flexibility than for simplicity. Due to this fact, an increasing number of specific mining tools have been developed to solve different educational problems [5]. Of all of them, only one small subgroup of tools is specifically oriented to using Moodle data, such as:

- GISMO [1] for visualizing graphically what is happening in Moodle courses.
- Meerkat-ED [4] for analyzing student participation in Moodle discussion forums using social network analysis techniques.
- MMT tool [3] for carrying out data mining processes of Moodle data for newcomers.
- DRAL [6] for discovering relevant e-Activities for Moodle learners.

However, most of these tools are standalone applications that are not integrated into the actual Moodle interface alongside the Moodle resources, activities, modules and blocks. Only GISMO [1] is integrated into the Moodle system, but it only visualizes data and does not perform data mining. In this paper, we describe a specific tool that we have developed as a new Moodle block for visualizing and mining student usage data.

2. TOOL DESCRIPTION

Our tool has been developed in PHP language and integrated into Moodle as a new block (an item which may be added to the left or right or centre column of any page in Moodle). It consists of two main tabs or components:

2.1 Data Selection and Visualization

This tab allows instructors to select and visualize usage information about the students enrolled on a Moodle course (see Figure 1). It provides basic statistics and graphics about the students registered on the course and the resources provided in it. Instructors can select one item (student or resource) manually or the full set of students or resources. There are different types of available statistics or information (grading, historical record, questionnaires forums, resources and an overall summary).

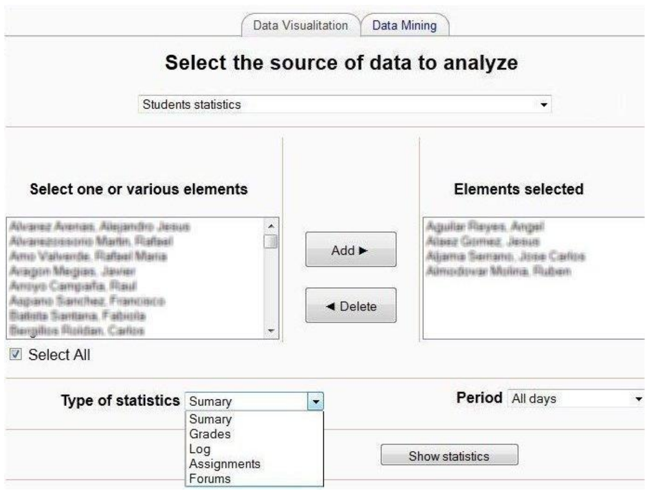


Figure 1: Main window for selecting and visualizing data.

The results can be shown in graphic mode in a pop-up window (see Figure 2) or in table mode in the current window (see Figure 3).

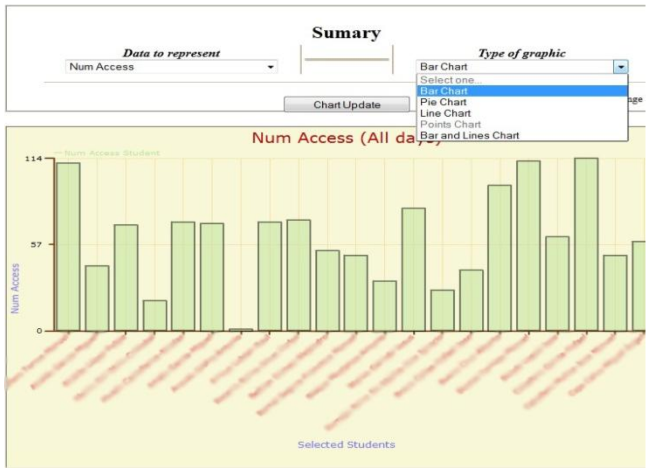
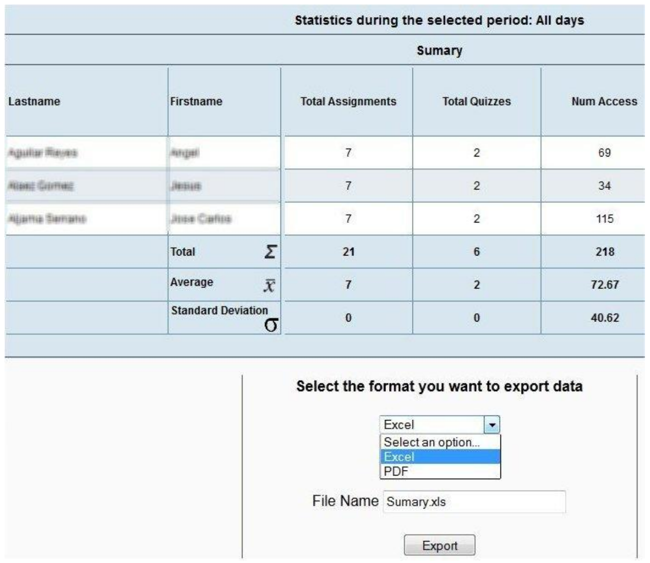


Figure 2: Bar diagram about the number of resources accessed.

In the new pop-up window (Figure 2), the instructor can select the attribute and type of graphics to visualize.



Statistics during the selected period: All days

Sumary

Lastname	Firstname	Total Assignments	Total Quizzes	Num Access
[illegible]	[illegible]	7	2	69
[illegible]	[illegible]	7	2	34
[illegible]	[illegible]	7	2	115
	Total Σ	21	6	218
	Average $\bar{x}$	7	2	72.67
	Standard Deviation σ	0	0	40.62

Select the format you want to export data

Excel: Select an option... / Excel / PDF

File Name: Sumary.xls

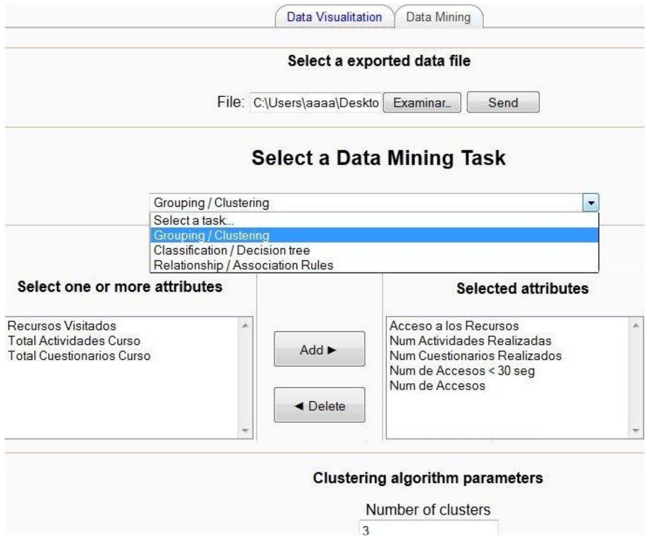
Export

Figure 3: Summarization table of a course.

The table information (see Figure 3) shows data in columns together with the total, average and standard deviation. Finally, the table can be saved/exported to an Excel file (for mining purposes) or to a PDF file (for reporting purposes).

2.2 Data Mining

This tab enables the instructor to do data mining starting from a previously saved Excel data file. Currently, it allows three different types of data mining methods/tasks to be performed: classification, association and clustering by using C4.5, Apriori and K-means algorithms, respectively (see Figure 4).



Data Visualitation | Data Mining

Select a exported data file

File: C:\Users\aaaa\Deskto | Examinar... | Send

Select a Data Mining Task

Grouping / Clustering: Select a task... / Grouping / Clustering / Classification / Decision tree / Relationship / Association Rules

Select one or more attributes:
Recursos Visitados
Total Actividades Curso
Total Cuestionarios Curso

Add ▶ | ◀ Delete

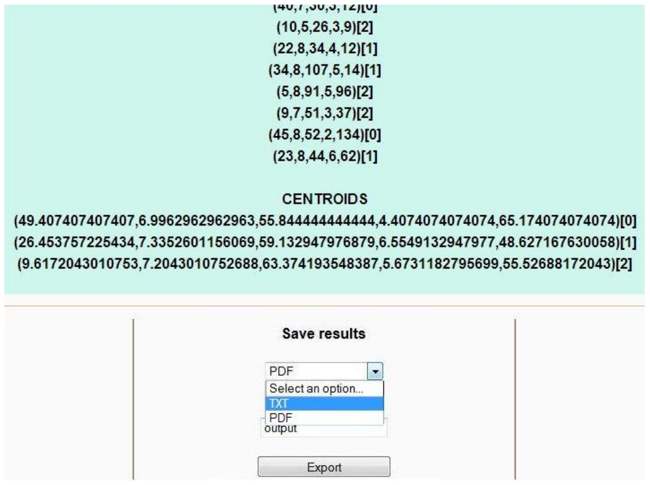
Selected attributes:
Acceso a los Recursos
Num Actividades Realizadas
Num Cuestionarios Realizados
Num de Accesos < 30 seg
Num de Accesos

Clustering algorithm parameters

Number of clusters: 3

Figure 4: Main window for mining data.

The instructor also has to select both the attributes to use from the data file and the parameter values of the algorithm. Once the algorithm has been executed, the model obtained/discovered is shown and can be saved as a PDF or plain text file. For example, Figure 5 shows the model or result (the instances together with the assigned cluster, and the centroids information) obtained after executing the clustering algorithm.



```
[illegible]
(10,5,26,3,9)[2]
(22,8,34,4,12)[1]
(34,8,107,5,14)[1]
(5,8,91,5,96)[2]
(9,7,51,3,37)[2]
(45,8,52,2,134)[0]
(23,8,44,6,62)[1]

CENTROIDS
(49.407407407407,6.9962962962963,55.844444444444,4.4074074074074,65.174074074074)[0]
(26.453757225434,7.3352601156069,59.132947976879,6.5549132947977,48.627167630058)[1]
(9.6172043010753,7.2043010752688,63.374193548387,5.6731182795699,55.52688172043)[2]
```

Save results

PDF: Select an option... / TXT / PDF

output

Export

Figure 5: Result window of clustering algorithm.

3. CONCLUSIONS

In the future, we intend to add various new data mining algorithms in order to provide for more advanced algorithms of each type. Also, we would like to add a specific pre-processing step/tab that lets the instructor modify the selected data before running the data mining.

4. ACKNOWLEDGMENTS

This work was supported by the Regional Government of Andalusia and the Spanish Ministry of Science and Technology projects, P08-TIC-3720 and TIN-2011-22408, respectively, FEDER funds.

5. REFERENCES

[1] Mazza, R., Milani, C. GISMO: a Graphical Interactive Student Monitoring Tool for Course Management Systems, In International Conference on Technology Enhanced Learning, Milan, 1-8. 2004.

[2] Mikut, R., Reischl, M. Data Mining Tools. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 1,5, 431–443. 2011.

[3] Pedraza-Perez, R. Romero, C., Ventura, S. A Java Desktop Tool for Mining Moodle Data. In International Conference on Educational Data Mining, Eindhoven, 319-320. 2011.

[4] Rabbany, R., Takaffoli, M., Zaïane, O. Analyzing Participation of Students in Online Courses Using Social Network Analysis Techniques. In International Conference on Educational Data Mining, Eindhoven, 21-30. 2011.

[5] Romero, C., Ventura, S. Data Mining in Eduation. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, In Press. 2013.

[6] Zafra, A., Romero, C., Ventura, S. DRAL: A Tool for Discovering Relevant e-Activities for Learners. Knowledge and Information Systems. In Press. 2013.

SEMILAR: A Semantic Similarity Toolkit For Assessing Students' Natural Language Inputs

Vasile Rus
Department of Computer Science
The University of Memphis
Memphis, TN 38152
vrus@memphis.edu

Rajendra Banjade
Department of Computer Science
The University of Memphis
Memphis, TN 28152
rbanjade@memphis.edu

Mihai Lintean
Department of Computer Science
The University of Memphis
Memphis, TN 38152
mclinten@memphis.edu

Nobal Niraula
Department of Computer Science
The University of Memphis
Memphis, TN 38152
nbnraula@memphis.edu

Dan Stefanescu
Department of Computer Science
The University of Memphis
Memphis, TN 38152
dnstfnscu@memphis.edu

ABSTRACT

We present in this demo SEMILAR, a SEMantic similarity toolkit. SEMILAR includes offers in one software environment several broad categories of semantic similarity methods: vectorial methods including Latent Semantic Analysis, probabilistic methods such as Latent Dirichlet Allocation, greedy lexical matching methods, optimal lexico-syntactic matching methods based on word-to-word similarities and syntactic dependencies with negation handling, kernel based methods, and some others. We will demonstrate during this demo presentation the efficacy of using SEMILAR to investigate and tune assessment algorithms for evaluating students' natural language input based on data from the DeepTutor computer tutor.

Keywords

Natural language student inputs, assessment, conversational tutors.

1. INTRODUCTION

In dialogue-based Intelligent Tutoring Systems (ITS; Rus, D'Mello, Hu, & Graesser, in press; Evens & Michael, 2005), it is important to understand students' natural language responses. Accurate assessment of students' responses enables the building of accurate student models for both cognition and affect. An accurate student model in turn affects the quality of tutor's feedback (Rus & Lintean, 2012). In general, accurate student models lead to improved macro- and micro-adaptivity in ITSs which is needed for effective tutoring (Rus, D'Mello, Hu, & Graesser, in press).

There are at least two different types of natural language assessments in conversational ITSs. First, there is need for advanced natural language algorithms to interpret the meaning of students' natural language contributions at each turn in the dialogue. The student responses in the middle of the dialogue tend to be short, i.e. the length of a sentence or less. There is also a need to assess the more comprehensive, essay-type answers that students are required to provide immediately after being prompted to solve a problem. These essay-type answers can be a paragraph long or even longer depending on the task and target domain.

One approach to assessing students' responses is to compute how similar the responses are to benchmark solutions provided by experts (Rus & Graesser, 2006). That is, *semantic similarity* is the underlying principle for computing the meaning of student contributions in many of today's state-of-the-art conversational ITSs and in other mainstream natural language processing applications such as Question Answering or Paraphrase Identification. The alternative approach to natural language understanding, called true understanding, is impractical as it requires world knowledge which is an intractable problem in Artificial Intelligence.

As already mentioned, in the semantic similarity approach a student contribution is assessed in terms of its similarity to an expert answer. The expert answer is deemed correct. Therefore, a student contribution is deemed correct if it is semantically similar to the expert answer (and incorrect otherwise).

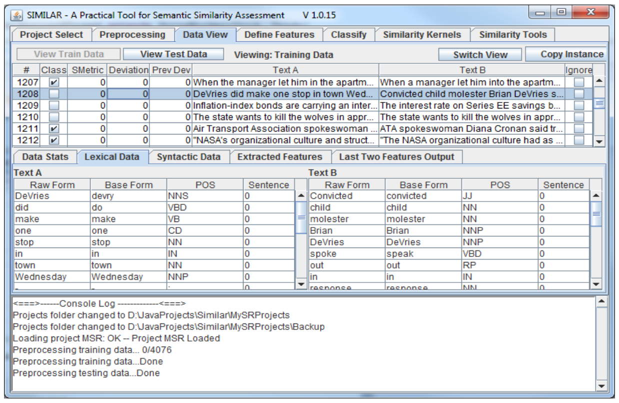
Below, we show an example of a real student response from an ITS and the corresponding expert-answer as authored by an expert.

Student Response: *An object that has a zero force acting on it will have zero acceleration.*

Expert Answer: *If an object moves with a constant velocity, the net force on the object is zero.*

The student response above is deemed correct as it is semantically similar to the expert answer. In general, the student response is deemed incorrect if it is not semantically similar enough to the expert response. More nuanced assessments can be made (e.g., partially correct or partially correct and partially incorrect at the same time).

Researchers have been proposing various methods to assess the semantic similarity of texts, in particular sentences (Corley and Mihalcea, 2005; Fernando & Stevenson, 2008; Rus, Lintean, Graesser, and McNamara 2009). However, there is no software library or toolkit that would allow for a fair comparison and investigation of the various methods. Furthermore, there is no one-stop-shop kind of environment to explore semantic similarity methods at various levels of granularity: word-to-word, sentence-to-sentence, paragraph-to-paragraph, or document-to-document similarity. Furthermore, mixed combinations of similarity could be imagined such as examining how similar a summary paragraph is to a document (useful in summarization).



#	Class	SMetric	Deviation	Prev Dev	Text A	Text B	Ignore
1207	✓	0	0	0	When the manager let him in the apartm...	When a manager let him into the apartm...	☐
1208	☐	0	0	0	DeVries did make one stop in town Wed...	Convicted child molester Brian DeVries s...	☐
1209	☐	0	0	0	Inflation-index bonds are carrying an inter...	The interest rate on Series EE savings b...	☐
1210	☐	0	0	0	The state wants to kill the wolves in appr...	The state wants to kill the wolves in appr...	☐
1211	✓	0	0	0	Air Transport Association spokeswoman ...	ATA spokeswoman Diana Cronan said tr...	☐
1212	✓	0	0	0	"NASA's organizational culture and struct...	"The NASA organizational culture had as ...	☐

Text A

Raw Form	Base Form	POS	Sentence
DeVries	devry	NNS	0
did	do	VBD	0
make	make	VB	0
one	one	CD	0
stop	stop	NN	0
in	in	IN	0
town	town	NN	0
Wednesday	Wednesday	NNP	0

Text B

Raw Form	Base Form	POS	Sentence
Convicted	convicted	JJ	0
child	child	NN	0
molester	molester	NN	0
Brian	Brian	NNP	0
DeVries	DeVries	NNP	0
spoke	speak	VBD	0
out	out	RP	0
in	in	IN	0
response	response	NN	0

```
<===>-------Console Log --------------<===>
Projects folder changed to D:\JavaProjects\Similar\MySRProjects
Projects folder changed to D:\JavaProjects\Similar\MySRProjects\Backup
Loading project MSR: OK -- Project MSR Loaded
Preprocessing training data... 0/4076
Preprocessing training data...Done
Preprocessing testing data...Done
```

Figure 1. Snaphsot of SEMILAR (Data View Pane).

Given the importance of assessing students' natural language inputs for building accurate student models, there is an acute need for such a software environment that would allow for a systematic and fair comparison of the various semantic similarity methods to assess students' natural language inputs.

The proposed SEMILAR (SEMantic simILARity) toolkit address this need by offering a java library as well as a GUI-based Java application that integrates a myriad of semantic similarity methods for tuning and optimizing the parameters of such methods for the student assessment task in conversational ITSs.

2. SEMILAR: THE SEMANTIC SIMILARITY TOOLKIT

The authors of the SEMILAR toolkit have been involved in assessing the semantic similarity of texts for more than a decade.

During this time, they have conducted a careful requirements analysis for an integrated software toolkit to be used for semantic similarity assessment. The result of this effort is the prototype presented here.

The SEMILAR toolkit includes the following components: project management; data view/browsing/visualization; textual preprocessing (e.g., tokenization, lemmatization/stemming, collocation identification, part-of-speech tagging, phrase or dependency parsing, etc.), semantic similarity methods, classification components (naïve Bayes, Decision Trees, Support Vector Machines, and Neural Network), kernel-based methods (sequence kernels, word sequence kernels, and tree kernels; as of this writing, we are still implementing several other tree kernel methods); debugging and testing facilities for model selection; and annotation components (allows domain expert to manually annotate texts with semantic relations using GUI-based facilities). For space reasons, we will only detail next the core component that includes the text-to-text similarity algorithms available in SEMILAR.

We briefly present core methods available as of this writing:

- a greedy method based on word-to-word similarity measures
- an optimal matching solution based on word-to-word similarity measures. The optimal lexical matching is based on the optimal assignment problem, a fundamental combinatorial optimization problem which consists of finding a maximum weight matching in a weighted bipartite graph;
- a lexical overlap component combined with syntactic overlap and negation handling to compute an unidirectional subsumption score between two sentences, T (Text) and H (Hypothesis), typically used in textual entailment which as a text-to-text semantic relation;
- a method in which similarities among all pairs of words are taken into account for computing the similarity of two texts. A similarity matrix operator W that contains word-to-word similarities between any two words is used;
- a weighted-LSA (wLSA) method for semantic similarity based on Latent Semantic Analysis. The similarity of two texts A and B can be computed using the cosine (normalized dot product) of their LSA vectors. Alternatively, the individual word vectors can be combined through weighted sums. A combination of 3 local weights and 3 global weights are available.
- A set of similarity measures based on the unsupervised method Latent Dirichlet Allocation. LDA is a probabilistic generative model in which documents are viewed as distributions over a set of topics (θ_d text d's distribution over topics) and topics are distributions over words (φ_t – topic t's distribution over words).
- The Quadratic Assignment Problem (QAP) method aims at finding an optimal assignment from words in text A to words in text B, based on individual word-to-word similarity, while simultaneously maximizing the match between the syntactic dependencies of the matching words. The Koopmans-Beckmann formulation of the QAP problem best fits the purpose of semantic similarity. The QAP method provides best accuracy results (=77.6%) that rival the best reported results so far (Madnani, Tetreault & Chodorow, 2012).

3. ACKNOWLEDGMENTS

This research was supported in part by Institute for Education Sciences under awards R305A100875.

4. REFERENCES

[1] Evens, M., and Michael, J. 2005. One-on-one Tutoring by Humans and Machines. Mahwah, NJ: Lawrence Erlbaum Associates.

[2] Graesser, A. C., Rus., V., D'Mello, S., K., & Jackson, G. T. (2008). AutoTutor: Learning through natural language dialogue that adapts to the cognitive and affective states of the learner. In D. H. Robinson & G. Schraw (Eds.), Current perspectives on cognition, learning and instruction: Recent innovations in educational technology that facilitate student learning (pp. 95–125). Information Age Publishing.

[3] Rus, V. & Lintean, M. (2012). A Comparison of Greedy and Optimal Assessment of Natural Language Student Input Using Word-to-Word Similarity Metrics, Proceedings of the Seventh Workshop on Innovative Use of Natural Language Processing for Building Educational Applica-tions, NAACL-HLT 2012, Montreal, Canada, June 7-8, 2012.

Gathering Emotional Data from Multiple Sources

Sergio Salmeron-Majadas
aDeNu Research Group. UNED
Calle Juan del Rosal, 16. Madrid
28040. Spain
+34 91 398 93 88
ssalmeron@bec.uned.es

Olga C. Santos
aDeNu Research Group. UNED
Calle Juan del Rosal, 16. Madrid
28040. Spain
+34 91 398 93 88
ocsantos@dia.uned.es

Jesus G. Boticario
aDeNu Research Group. UNED
Calle Juan del Rosal, 16. Madrid
28040. Spain
+34 91 398 91 97
jgb@dia.uned.es

Raúl Cabestrero
UNED
Calle Juan del Rosal, 10. Madrid
28040. Spain
+34 91 398 62 40
rcabestrero@psi.uned.es

Pilar Quirós
UNED
Calle Juan del Rosal, 10. Madrid
28040. Spain
+34 91 398 82 48
pquiros@psi.uned.es

Mar Saneiro
aDeNu Research Group. UNED
Calle Juan del Rosal, 16. Madrid
28040. Spain
+34 91 398 91 98
marsanerio@dia.uned.es

ABSTRACT

Collecting and processing data in order to detect and recognize emotions has become a research hot topic in educational scenarios. We have followed a multimodal approach to collect and process data from different sources to support emotion detection and recognition. To illustrate the approach, in this demo, participants will be shown what emotional data can be gathered while solving Math problems.

Keywords

Affective Computing, Data Mining, Sensor Data, Emotion Detection, Mathematics

1. INTRODUCTION

Currently there is a growing interest in offering emotional support to learners in e-learning platforms through an expanded set of adaptive features. A key issue is to determine learners' affective state, which is related to their cognitive and metacognitive process [4], preferable with low cost sensors [2]. Affective states in our approach are to be defined from mining in a jointly manner subjective, physiological and behavioral data gathered from diverse emotional information sources while the learner interacts on the given e-learning environment. This approach offers possible improvements on emotion detection, which as suggested in the literature may come out from the combination of different data sources simultaneously [5]. Math problem solving scenarios have provided opportunities to investigate this new approach, as from them different emotions may be elicited [7].

2. OUR APPROACH

As for emotion detection, our approach is based on the use of data mining techniques. As shown in Figure 1, we follow a multimodal gathering approach based on the combination of the following data sources obtained while the learner carries out learning interactions to solve Mathematical tasks in the e-learning platform and stored in the corresponding user model. To start with, bio-feedback data provide appropriate measures to detect typical physiological reactions that come along with emotions. Although they should not be used for categorizing discrete emotions on its own, they provide useful indicators of the participants' arousal level associated with the ongoing affective state over the learning process. Signals used to this end are: heart rate, breath frequency, galvanic skin response and skin temperature. To evaluate phasic variations on collected signals upon a tonic state, recordings of each learner pre-baseline are done to provide reference values for subsequent analysis.

Another key source for gathering affective information is the non-verbal behavior (e.g. gestures, facial expression, body movements). Facial expressions of participants are recorded by Windows Kinect face features extraction. Kinect for Windows device provides an API able to detect a user's face model based in 100 points. The processing of these data is to identify the learner's head position, inclination and expressions. In addition, a webcam (with microphone) is used to record other sources of information not necessarily located in the participant's face expression, such as verbal expression and speech tone.

Some additional user interactions are also gathered. In particular, keyboard and mouse data sources are recorded to find out behavioral correlates of the emotional intensity. To collect all the events triggered by mouse and keyboard, a key logger and mouse tracker has been developed in Java (with no GUI, so it cannot interfere with the user interactions) using the library provided by kSquared.de. A video of participant's desktop is also recorded to keep track of the session.

As this approach is based on a wide range of information sources, synchronization is a key issue. Due to the number of devices used, several computers may be needed to collect the required information. Thus, the synchronization of the systems involved (given that some of the recorded interactions can last less than a second) is needed. Through this, synchronization data are merged and data mining can be applied.

Information about learners' personality is also considered when processing the data, given the narrow relation between personality traits and affective states with learning styles and strategies [3]. To get this information, some self-reports are used. To gather personality traits, learners fill the Big Five Inventory (BFI), that reveals the main five structural dimensions of personality and the General Self-Efficacy Scale (GSE) that assesses participants self-beliefs in coping with a variety of demands in life. Moreover, as supervised learning techniques are considered due to their well-known benefits for emotions detection [8], some labeling is needed. With this aim, participants are told to rate proposed activities in the valence (from negative to positive) and arousal (from calm to excited) dimensions using the Self-Assessment Manikin (SAM) scale. Other meaningful sources to gather learners' subjective emotions are obtained by self-described emotional reports (participants are asked to write their emotions in natural language) and the 20 emotional states of the Positive and Negative Affect Schedule (PANAS). The learners' performance regarding the Mathematical tasks is another information source that should be taken into account from their learning outcomes.

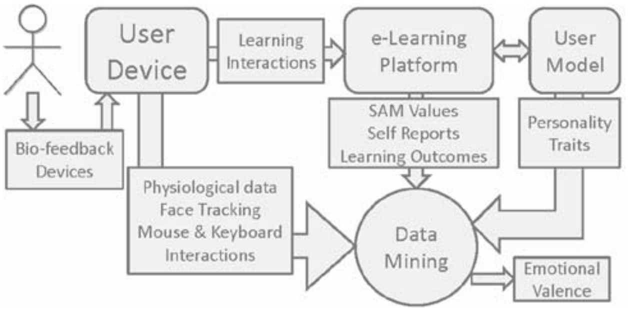


Figure 1. Data gathering flow

Once data have been gathered, and before data mining processes are carried out, each kind of data is to be pre-processed in a particular way depending on their nature. Physiological data are being pre-processed to find significant variations on the signals with regard to the pre-baseline. Relations between signal variation patterns found and the self-reported affective labeling given by participants provide valuable information. A similar approach is being followed to the interaction data, relating interaction changes (in keyboard typing or mouse clicking behavior) with the subjective emotional reported values. Text mining techniques have been applied to extract valence values depending on the terms typed by the user when writing the emotional reports. Data recorded with Kinect are being pre-processed to extract emotional patterns according to gestures and movements from detected facial points. The result of these processes is to be used (in the same way as the emotional feedback and subjective information collected from the user commented above) to emotionally label data gathered from sensors in supervised learning systems. The current state of this work will be presented in a demo where participants are to carry out mathematical exercises while some of the above information will be collected and shown, and the data mining process followed discuss with the participant.

3. ONGOING WORKS

This approach is supported by the MAMIPEC project, where we are exploring how to combine different information sources from different signals to offer an accessible and personalized learning experience to the learner, which accounts for their affective state and aims to provide, accordingly, affective based recommendations. To progress on this goal, a large-scale experiment was carried out where 72 participants (excluding 10 additional pilots to test the settings) performed different types of individual activities, which consisted in a Math problem solving experience implemented in dotLRN e-learning platform. Text mining techniques are being applied over the emotional report in order to extract valence information. Using the text mining scores (filtering out those reports with a difference of less than 3 words when subtracting positive items from negative items when computing the frequency of affective words from the MPQA database) and the keyboard interactions, our best results were roughly a 70% success rate when predicting positive or negative valence compared to the experts' labeling [6]. New experiments using this approach with a problem solving ITS [1] have been proposed in cooperation with Valencia University.

4. ACKNOWLEDGMENTS

Authors would like to thank the Spanish Government for funding MAMIPEC project (TIN2011-29221-C03-01).

5. REFERENCES

[1] Arnau, D., Arevalillo-Herráez, M., Puig, L., González-Calero, J.A. 2013. Fundamentals of the design and the operation of an intelligent tutoring system for the learning of the arithmetical and algebraic way of solving word problems. Computers & Education. 63,119-130.

[2] Arroyo, I., Cooper, D., Burleson, W., Woolf, B.P., Muldner, K., Christopherson, R. 2009. Emotion Sensors Go To School. Proceedings of the 14th International Conference on Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling, 17-24.

[3] Bidjerano, T., Dai, D.Y. 2007. The relationship between the big-five model of personality and self-regulated learning strategies. Learning and Individual Differences, 17 (1) 69–81.

[4] Blanchard, E.G., Volfson, B., Hong, Y.J., Lajoie, S.P. 2009. Affective Artificial Intelligence in education: From detection to adaptation. Proceeding of the 2009 conference on Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling, 81–88.

[5] D'Mello, S. K., Kory, J. 2012. Consistent but Modest: A Meta-Analysis on Unimodal and Multimodal Affect Detection Accuracies from 30 Studies. In L. P. Morency et al (Eds) Proceedings of the 14th ACM International Conference on Multimodal Interaction, 31-38.

[6] Santos, O.C., Salmeron-Majadas, S., Boticario, J.G. 2013. Emotions detection from math exercises by combining several data sources. In proceedings of the 16th International Conference on Artificial Intelligence in Education, 742–745.

[7] Shen, L., Wang, M. and Shen, R. 2009. Affective e-learning: Using "emotional" data to improve learning in pervasive learning environment. Educational Technology & Society. 12 (2), 176–189.

[8] Zeng, Z., Pantic, M., Roisman, G.I.,Huang, T.S. 2009. A survey of affect recognition methods: Audio, visual, and spontaneous expressions. IEEE Transactions on Pattern Analysis and Machine Intelligence, 31 (1), 39–58.

A Tool for Speech Act Classification Using Interactive Machine Learning

Borhan Samei
The University of Memphis
365 Innovation Dr.
Memphis, TN, USA
bsamei@memphis.edu

Fazel Keshtkar
The University of Memphis
365 Innovation Dr.
Memphis, TN, USA
fkshtkar@memphis.edu

Arthur C. Graesser
The University of Memphis
365 Innovation Dr.
Memphis, TN, USA
a-graesser@memphis.edu

Vasile Rus
The University of Memphis
365 Innovation Dr.
Memphis, TN, USA
vrus@memphis.edu

ABSTRACT

In this Demo, we introduce a tool that provides a GUI interface to a previously designed to Speech Act Classifier. The tool also provides features to manually annotate data by human and evaluate and improve the automated classifier. We describe the interface and evaluate our model with results from two human judges and Computer.

Keywords

Speech act, interactive machine learning, Speech act classifier

1. INTRODUCTION

Speech act classification is the task of classifying a sentence, utterance or any other discourse contribution, into a speech act category which is selected from a set of predefined categories. Each category represents a particular social discourse function. “What is your name?” for example, is classified as a Question. There are other examples of speech act categories, such as Statement, Greeting, etc.

Our tool is designed to offer annotation facilities in order to improve a previously developed automated speech act classification (SAC; the SAC is available online at www.cs.memphis.edu/~vrus/SAC/) [2]. Furthermore, we provide a GUI-based interface to the speech act classifier [2]. We use an interactive machine learning model for this task that allows for manual classification by human judges which is used to improve the accuracy of our machine learning model. The tool is created and written in Java. The SAC relies on decision tree (J48) that has proved to provide based performance on training data from human annotated utterances [2].

The decision tree is a machine learning approach that requires a feature set to be designed. The feature set is an important part of machine learning algorithms. Moldovan, Rus, and Graesser [2] designed a feature set and used it in order to automatically classify chat utterances.

They used eight speech act categories which are shown in Table 1. According to analyzes on a variety of corpora, such as chat and multiparty games we can converge on a set of speech act categories that are both theoretically justified and can be used by trained judges [3]. For the feature set we tokenize the chat utterances based on basic regular expressions and for each utterance five features are extracted: the first three words, the last word, and the length of the sentence in words. Many other feature sets have been experimented with but the five features just mentioned proved to lead to highest performance in conjunction with decision trees and naïve Bayes methods [2]. The model has been trained and experimented with on data sets from intelligent tutoring systems as well as chat data [2]. Our model is a J48 model built on the training the data using Weka toolkit.

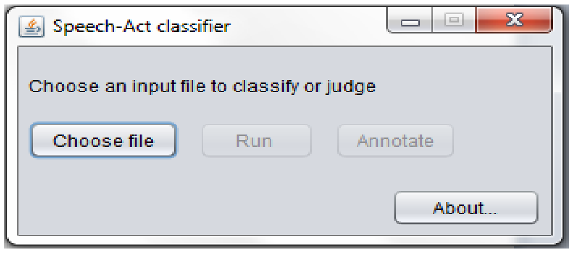
The model and training data can be updated and improved independently. This tool can be used with several training data based on the domain. For example, if we are looking at the dialogs in a movie we can use a different training data model based on this domain.

Table 1. Set of speech act categories and an example of each category.

Speech act category	Example
ExpressiveEvaluation	Your stakeholders will be grateful!
Greeting	Hello!
MetaStatements	oh yeah, last thing.
Statement	a physical representation of data.
Question	What should we do?
Reaction	Thank you
Request	Please check your inbox
Other	ed is tough, no doubt.

2. THE INTERFACE

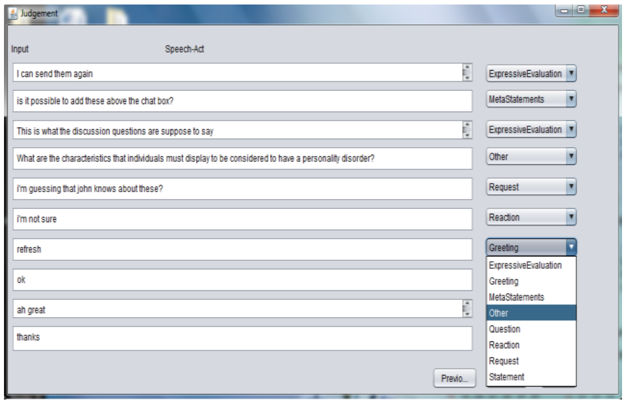
We have designed a Graphical User Interface (GUI) for the tool. It can be used on any machine, since it is implemented in Java. Figure 1 shows a snapshot of the starting interface of the tool. Use is able to “Run” or “Annotate” the input data (see Figure 1).



Choose an input file to classify or judge

Figure 1. A snapshot of the starting window of the tool's interface.

By clicking on “Run” the tool will start to classify the input file which contains a collection of utterances. After classifying the utterances, the output will be saved as an excel file. The users can also click on “Annotate” to annotate the data manually. By clicking on “Annotate” a new GUI will appear (Figure 2) which contains the utterances. The user will see 10 utterances in each step and for each utterance there is a drop down list of categories from which the user selects one. After annotating by user can go to next utterances or save the current annotation any time and do the rest after getting back.



Input	Speech-Act
I can send them again	ExpressiveEvaluation
is it possible to add these above the chat box?	MetaStatements
This is what the discussion questions are suppose to say	ExpressiveEvaluation
What are the characteristics that individuals must display to be considered to have a personality disorder?	Other
i'm guessing that john knows about these?	Request
i'm not sure	Reaction
refresh	Greeting
ok	
ah great	
thanks	

Figure 2. A snapshot of the manual annotation tool.

3. RESULTS

A collection of chat utterances are used as the data set to evaluate the algorithm. The system is trained by a collection of datasets derived from Auto-Mentor frame board dataset. The test data is chosen from a collection of new chat data. We have chosen a hundred chat utterances from the data and tried to maintain a normal distribution on the speech act categories so each category has 10 to12 utterances in our test data. The test data is also annotated by two human judges.

The system runs the algorithm on the test data and for each utterance we show top three speech-act categories based on their probability distribution in the decision tree. Top three categories are the ones with the highest probability and we represent these 3 categories as Comp1, Comp2, and Comp3. Figure 3 represents the agreement of Human judges with Comp1, Comp2, and Comp3.

As our model is based on interactive machine learning we tend to compare automated classification to human judges in order to improve the model and retrain with new enhanced training data. To do this, we have calculated agreement among humans and computer. For each utterance we have five output categories, the top three assigned by our model, and the annotations by human judges. These five outputs are compared to investigate their agreement and evaluate the current model. We have looked at agreement both by Speech-Act categories and overall among the dataset.

Table 2 shows the overall agreement among the classifiers. Our human judges agree on 70.00% of the utterances. Agreement of our model and human judges is about 50%. The agreement of judges with Comp2 and Comp3 are less than 5% of the human agree with the second and third category computed by our model. As it was mentioned earlier Comp2 and Comp3 are actually the two categories that our model would propose based in their probability in the decision tree. This result helps improve our model by increasing the probability of the right categories based on human judge annotations, in the decision tree. We will also look at the agreement by.

Table 2. Agreement among judges and classifiers.

Judges	Agreement
Human1 – Human2	70%
Human1 – Comp1	50%
Human2 – Comp1	47%
Human1 – Comp2	4%
Human2 – Comp2	3%
Human1 – Comp3	2%
Human2 – Comp3	3%

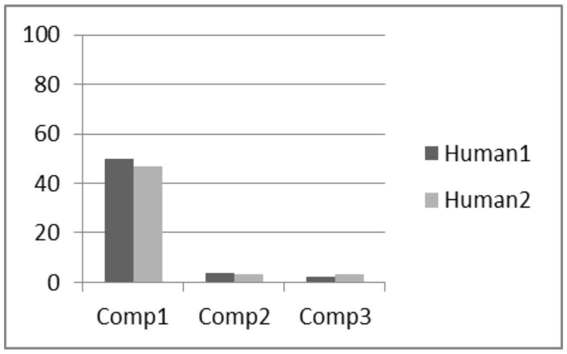


Figure 3. Overall agreement of human judges with automated classifier.

4. CONCLUSION

As the results show, our model is working close to human judges. The tool can be used to improve the model by taking both human and computer annotations and enhance the training data. The main goal of this tool is not only to automate the classification task, but also provide more features to improve the classifier. Both automated and manual annotations are easy to use by the interface and this can be used in several applications and domains. The training data and J48 models can be externally changed for different domains.

5. References

[1] Bagley, E., 2011 , Stop Talking and Type: Mentoring in a Virtual and Face-to-Face Environmental Education Environment. Ph.D. thesis, University of Wisconsin-Madison

[2] Moldovan, C., Rus, V., Graesser, A.C. 2011, Automated speech act classification for on-
line chat. In: Proceeding of FLAIRS 2011

[3] R.G. DAndrade, M.W., 1985, Speech act theory in quantitative research on inter-personal behavior. In: Discourse Processes. 8 (2), 229-259

Author Index

ISBN: 978-0-9839525-2-7