

ISBN: 978-0-615-37529-8

Preface

The Third International Conference on Data Mining (EDM 2010) was held in Pittsburgh, PA, USA. It follows the second conference at the University of Cordoba, Spain, on July 1–3, 2009 and the first edition of the conference held in Montreal in 2008, and a series of workshops within the AAI, AIED, EC-TEL, ICALT, ITS, and UM conferences. EDM2011 will be held in Eindhoven, Netherlands.

EDM brings together researchers from computer science, education, psychology, psychometrics, and statistics to analyze large data sets to answer educational research questions. The increase in instrumented educational software and databases of student test scores, has created large repositories of data reflecting how students learn. The EDM conference focuses on computational approaches for analyzing those data to address important educational questions. The broad collection of research disciplines ensures cross fertilization of ideas, with the central questions of educational research serving as a unifying focus.

We received a total of 54 full papers and 20 submitted posters from 21 countries. Paper submissions were reviewed by three or four reviewers and 23 of them were accepted as full papers (43% acceptance rate). All papers will appear both on the web, at www.educationaldatamining.org, as well as in the printed proceedings. The conference also included invited talks by Professor Cristina Conati, Computer Science Professor, Computer Science Department and Laboratory for Computational Intelligence at the University of British Columbia, Canada and by Professor Osmar Zaiane Ph.D., Professor, Department of Computing Science, University of Alberta, Canada.

We would like to thank Carnegie Mellon University for their hosting of EDM2010, and thank the Pittsburgh Science of Learning Center DataShop and Carnegie Learning Inc for their generous sponsorship. We would like to thank the program committee members, local committee, web chair, the reviewers and the invited speakers for their enthusiastic help in putting this conference together.

Ryan S.J.d. Baker,
Agathe Merceron,
Philip I. Pavlik Jr. (Eds.)

Conference Organization

Conference Chair: [Ryan S.J.d. Baker](#), Worcester Polytechnic Institute

Program Chairs: [Agathe Merceron](#), Beuth University of Applied Sciences Berlin;

[Philip I. Pavlik Jr.](#), Carnegie Mellon University

Local Organization Chair: John Stamper, Carnegie Mellon University

Web Chair: [Arnon Hershkovitz](#), Tel Aviv University

Program Committee

Esmá Aimeur, University of Montreal, Canada

Ivon Arroyo, University of Massachusetts Amherst, USA

Beth Ayers, Carnegie Mellon University, USA

Ryan Baker, Worcester Polytechnic Institute, USA

Tiffany Barnes, University of North Carolina at Charlotte, USA

Joseph Beck, Worcester Polytechnic Institute, USA

Bettina Berendt, Katholieke Universiteit Leuven, Belgium

Gautam Biswas, Vanderbilt University, USA

Cristophe Choquet, Université du Maine, France

Cristina Conati, University of British Columbia, Canada

Richard Cox, University of Sussex, UK

Michel Desmarais, Ecole Polytechnique de Montreal, Canada

Aude Dufresne, University of Montreal, Canada

Mingyu Feng, SRI International, USA

Art Graesser, University of Memphis, USA

Andreas Harrer, Katholische Universität Eichstätt-Ingolstadt, Germany

Neil Heffernan, Worcester Polytechnic Institute, USA

Arnon Hershkovitz, Tel Aviv University, Israel

Cecily Heiner, University of Utah, USA

Roland Hubscher, Bentley University, USA

Sebastian Iksal, Université du Maine, France

Kenneth Koedinger, Carnegie Mellon University, USA

Vanda Luengo, Université Joseph Fourier Grenoble, France

Tara Madhyastha, University of Washington, USA

Brent Martin, Canterbury University, New Zealand

Noboru Matsuda, Carnegie Mellon University, USA

Manolis Mavrikis, The University of Edinburgh, UK

Gordon McCalla, University of Saskatchewan, Canada

Bruce McLaren, Deutsches Forschungszentrum für Künstliche Intelligenz, Germany

Julia Mingullon Alfonso, Universitat Oberta de Catalunya, Spain

Tanja Mitrovic, Canterbury University, New Zealand

Jack Mostow, Carnegie Mellon University, USA

Rafi Nachmias, Tel Aviv University, Israel

Roger Nkambou, Université du Québec à Montréal (UQAM), Canada

Mykola Pechenizkiy, Eindhoven University of Technology, Netherlands
Steve Ritter, Carnegie Learning, USA
Cristobal Romero, Cordoba University, Spain
Carolyn Rose, Carnegie Mellon University, USA
Steven Tanimoto, University of Washington, USA
Sebastian Ventura, Cordoba University, Spain
Kalina Yacef, University of Sydney, Australia
Osmar Zaiane, University of Alberta, Canada

Steering Committee

Esmā Aïmeur, University of Montreal, Canada
Ryan Baker, Worcester Polytechnic Institute, USA
Tiffany Barnes, University of North Carolina at Charlotte, USA
Joseph E. Beck, Worcester Polytechnic Institute, USA
Michel C. Desmarais, Ecole Polytechnique de Montreal, Canada
Neil Heffernan, Worcester Polytechnic Institute, USA
Cristobal Romero, Cordoba University, Spain
Kalina Yacef, University of Sydney, Australia

External Reviewers

Hua Ai
Acey Boyce
Amanda Chaffin
Min Chi
David G. Cooper
Michael Eagle
Philippe Fournier-Viger
Ilya Goldin
Andrew Hicks
Matthew Johnson
Evgeny Knutov
Behrooz Mostafavi
Andrea Nickel
Zachary Pardos
Ildiko Pelczer
Reihaneh Rabbany
Jose-Raul Romero
Enrique Garcia Salcines
Oliver Scheuer
Benjamin Shih
Ekaterina Vasilyeva
Indre Zliobaite

Invited Speakers for Educational Data Mining 2010

Data-based Student Modeling in Exploratory Learning Environments

Cristina Conati

[Computing Science Department](#) & [Laboratory for Computational Intelligence, University of British Columbia](#) [[homepage](#)]



Abstract: Exploratory Learning Environments (ELE) are designed to help users acquire knowledge by freely experiencing a target domain. In this setting, it is often hard to identify interaction behaviours that are conducive to learning, vs. behaviours that indicate student confusion, making it hard to provide adaptive support to students who do not learn well with ELEs. In this talk, I will present our work on using data-based approaches to identify and recognize relevant behavioral patterns during interaction with ELEs, with the goal of enabling ELEs to monitor how a student works with the environment and provide adaptive support when needed.

Social Network Analysis for the Assessment of Learning

Osmar Zaiane

[Department of Computing Science, University of Alberta](#) [[homepage](#)]



Abstract: Using computer-supported collaborative learning tools, learners interact forming relationships and complex flows of information. In a forum with very few learners it is customary to quickly collect thousands of messages in few months, and these are interrelated in intricate discussion threads. Assessing the participation and interaction between learners can become a daunting task. Social network analysis is a field of study attempting to understand and measure relationships between entities in networked information. Can social network analysis techniques and data mining techniques for information networks help examine and assess online interactions? We examine some work done in this area, particularly the application of community mining, and discuss some open problems pertaining to social network analysis in the e-learning domain.

Table of Contents

Regular papers

Effort-based Tutoring: An Empirical Approach to Intelligent Tutoring -- Ivon Arroyo, Hasmik Mehranian and Beverly P. Woolf	1
An Analysis of the Differences in the Frequency of Students' Disengagement in Urban, Rural, and Suburban High Schools -- Ryan S.J.d. Baker and Sujith M. Gowda	11
On the Faithfulness of Simulated Student Performance Data -- Michel C. Desmarais and Ildiko Pelczer	21
Mining Bodily Patterns of Affective Experience during Learning -- Sidney D'Mello and Art Graesser	31
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)? -- Mingyu Feng and Neil Heffernan	41
Using Neural Imaging and Cognitive Modeling to Infer Mental States while Using an Intelligent Tutoring System -- Jon M. Fincham, John R. Anderson, Shawn Betts and Jennifer Ferris	51
Using multiple Dirichlet distributions to improve parameter plausibility -- Yue Gong, Joseph E. Beck and Neil T. Heffernan	61
Examining Learner Control in a Structured Inquiry Cycle Using Process Mining -- Larry Howard, Julie Johnson and Carin Neitzel	71
Analysis of Productive Learning Behaviors in a Structured Inquiry Cycle Using Hidden Markov Models -- Hogyong Jeong, Gautam Biswas, Julie Johnson and Larry Howard	81
Data Mining for Generating Hints in a Python Tutor -- Anna Katrina Dominguez, Kalina Yacef and James R. Curran	91
Off Topic Conversation in Expert Tutoring: Waste of Time or Learning Opportunity -- Blair Lehman, Whitney Cade and Andrew Olney	101

Sentiment Analysis in Student Experiences of Learning -- Sunghwan Mac Kim and Rafael A. Calvo	111
Online Curriculum Planning Behavior of Teachers -- Keith E. Maull, Manuel Gerardo Saldivar and Tamara Sumner	121
A Data Model to Ease Analysis and Mining of Educational Data -- André Krüger, Agathe Merceron and Benjamin Wolf	131
Identifying Students' Inquiry Planning Using Machine Learning -- Orlando Montalvo, Ryan S.J.d. Baker, Michael A. Sao Pedro, Adam Nakama and Janice D. Gobert	141
Skill Set Profile Clustering: The Empty K-Means Algorithm with Automatic Specification of Starting Cluster Centers -- Rebecca Nugent, Nema Dean and Elizabeth Ayers	151
Navigating the parameter space of Bayesian Knowledge Tracing models: Visualizations of the convergence of the Expectation Maximization algorithm -- Zachary Pardos and Neil Heffernan	161
Mining Rare Association Rules from e-Learning Data -- Cristóbal Romero, José Raúl Romero, Jose María Luna and Sebastián Ventura	171
Using Text Replay Tagging to Produce Detectors of Systematic Experimentation Behavior Patterns -- Michael Sao Pedro, Ryan S.J.d. Baker, Orlando Montalvo, Adam Nakama and Janice D. Gobert	181
Identifying High-Level Student Behavior Using Sequence-based Motif Discovery -- David H. Shanabrook, David G. Cooper, Beverly Park Woolf and Ivon Arroyo	191
Unsupervised Discovery of Student Strategies -- Benjamin Shih, Kenneth R. Koedinger and Richard Scheines	201
Assessing Reviewer's Performance Based on Mining Problem Localization in Peer-Review Data -- Wenting Xiong, Diane Litman and Christian Schunn	211
Using Numeric Optimization To Refine Semantic User Model Integration Of Adaptive Educational Systems -- Michael Yudelson, Peter Brusilovsky, Antonija Mitrovic and Moffat Mathews	221

Young Researcher Track papers

An Annotations Approach to Peer Tutoring -- John Champaign and Robin Cohen	231
Using Educational Data Mining Methods to Study the Impact of Virtual Classroom in E-Learning -- Mohammad Hassan Falakmasir and Jafar Habibi	241
Mining Students' Interaction Data from a System that Support Learning by Reflection -- Rajibussalim	249
Process Mining to Support Students' Collaborative Writing -- Vilaythong Southavilay, Kalina Yacef and Rafael A. Callvo	257

Poster Abstracts

Automatic Rating of User-Generated Math Solutions -- Turadg Aleahmad, Vincent Aleven and Robert Kraut	267
Tracking Students' Inquiry Paths through Student Transition Analysis -- Matt Bachmann, Janice Gobert and Joseph Beck	269
DISCUSS: Enabling Detailed Characterization of Tutorial Interactions Through Dialogue Annotation -- Lee Becker, Wayne H. Ward and Sarel vanVuuren	271
Data Mining of both Right and Wrong Answers from a Mathematics and a Science M/C Test given Collectively to 11,228 Students from India [1] in years 4, 6 and 8 -- James Bernauer and Jay Powell	273
Mining information from tutor data to improve pedagogical content knowledge -- Suchismita Srinivas, Muntaquim Bagadia and Anupriya Gupta	275
Clustering Student Learning Activity Data -- Haiyun Bian	277
Analyzing Learning Styles using Behavioral Indicators in Web based Learning Environments -- Nabila Bousbia, Jean-Marc Labat, Amar Balla and Issam Rebai	279

Using Topic Models to Bridge Coding Schemes of Differing Granularity -- Whitney L. Cade and Andrew Olney	281
A Distillation Approach to Refining Learning Objects -- John Champaign and Robin Cohen	283
A Preliminary Investigation of Hierarchical Hidden Markov Models for Tutorial Planning -- Kristy Elizabeth Boyer, Robert Phillips, Eun Young Ha, Michael D. Wallis, Mladen A. Vouk, and James C. Lester	285
Higher Contributions Correlate with Higher Learning Gains -- Carol Forsyth, Heather Butler, Arthur C. Graesser, Diane Halpern	287
Pinpointing Learning Moments; A finer grain P(J) model -- Adam Goldstein, Ryan S.J.d. Baker and Neil T. Heffernan	289
Predicting Task Completion from Rich but Scarce Data -- José P. González-Brenes and Jack Mostow	291
Hierarchical Structures of Content Items in LMS -- Sharon Hardof-Jaffe, Arnon HersHKovitz, Ronit Azran and Rafi Nachmias	293
Is Students' Activity in LMS Persistent? -- Arnon HersHKovitz and Rafi Nachmias	295
EDM Visualization Tool: Watching Students Learn -- Matthew M. Johnson and Tiffany Barnes	297
Inferring the Differential Student Model in a Probabilistic Domain Using Abduction inference in Bayesian networks -- Nabila Khodeir, Nayer Wanas, Nevin Darwish and Nadia Hegazy	299
Using LiMS (the Learner Interaction Monitoring System) to Track Online Learner Engagement and Evaluate Course Design -- Leah P. Macfadyen and Peter Sorenson	301
Observing Online Curriculum Planning Behavior of Teachers -- Keith E. Maull, Manuel Gerardo Saldivar and Tamara Sumner	303
When Data Exploration and Data Mining meet while Analysing Usage Data of a Course -- André Krüger, Agathe Merceron and Benjamin Wolf	305

AutoJoin: Generalizing an Example into an EDM query -- Jack Mostow and Bao Hong (Lucas) Tan	307
Conceptualizing Procedural Knowledge Targeted at Students with Different Skill Levels -- Martin Možina, Matej Guid, Aleksander Sadikov, Vida Groznik, Jana Krivec, and Ivan Bratko	309
Data Reduction Methods Applied to Understanding Complex Learning Hypotheses -- Philip I. Pavlik Jr.	311
Analysis of a causal modeling approach: a case study with an educational intervention -- Dovan Rai and Joseph E. Beck	313
Peer Production of Online Learning Resources: A Social Network Analysis -- Beijie Xu and Mimi M. Recker	315
Class Association Rules Mining from Students' Test Data -- Cristóbal Romero, Sebastián Ventura, Ekaterina Vasilyeva and Mykola Pechenizkiy	317
Modeling Learning Trajectories with Epistemic Network Analysis: A Simulation-based Investigation of a Novel Analytic Method for Epistemic Games -- Andre A. Rupp, Shauna J. Sweet and Younyoung Choi	319
Multiple Test Forms Construction based on Bees Algorithm -- Pokpong Songmuang and Maomi Ueno	321
Can Order of Access to Learning Resources Predict Success? -- Hema Soundranayagam and Kalina Yacef	323
A Data Driven Approach to the Discovery of Better Cognitive Models -- Kenneth R. Koedinger and John C. Stamper	325
Using a Bayesian Knowledge Base for Hint Selection on Domain Specific Problems -- John C. Stamper, Tiffany Barnes and Marvin Croy	327
A Review of Student Churn in the Light of Theories on Business Relationships -- Jaan Ubi and Innar Liiv	329

Towards EDM Framework for Personalization of Information Services in RPM Systems -- Ekaterina Vasilyeva, Mykola Pechenizkiy, Aleksandra Tesanovic, Evgeny Knutov, Sicco Verwer and Paul De Bra	331
A Case Study: Data Mining Applied to Student Enrollment -- César Vialardi, Jorge Chue, Alfredo Barrientos, Daniel Victoria, Jhonny Estrella, Juan Pablo Peche and Álvaro Ortigosa	333
Representing Student Performance with Partial Credit -- Yutao Wang, Neil T. Heffernan and Joseph E. Beck	335
Where in the World? Demographic Patterns in Access Data -- Mimi M. Recker, Beijie Xu, Sherry Hsi, and Christine Garrard	337
Pundit: Intelligent Recommender of Courses -- Ankit Ranka, Faisal Anwar, Hui Soo Chae	339
Author Index	341

Effort-based Tutoring: An Empirical Approach to Intelligent Tutoring

Ivon Arroyo¹, Hasmik Mehranian², Beverly P. Woolf¹

¹ Department of Computer Science, University of Massachusetts Amherst

² Dept. of Mechanical and Industrial Engineering, University of Massachusetts Amherst

Abstract. We describe pedagogical and student modeling based on past student interactions with a tutoring system. We model student effort with an integrated view of student behaviors (e.g. timing and help requests in addition to modeling success at solving problems). We argue that methods based on this integrated and empirical view of student effort at individual items accurately represent the real way that students use tutoring systems. This integrated view helps to discern factors that affect student behavior beyond cognition (e.g., help misuse due to meta-cognitive or affective flaws). We specify parameters to the pedagogical model in detail.

1 Introduction

The traditional structure of intelligent tutoring systems consists of a student model and a pedagogical model to guide activity selection, generally based on heuristics. The most traditional student models are based on tracing student cognitive knowledge [4]. Knowledge tracing (KT) procedures encode cognitive mastery of the different skills being tutored. KT consists of four parameters fit to each knowledge component (KC), and include: *initial learning*, *learning rate*, *guess* and *slip parameters*. One advantage of these models is that parameters are interpretable, and being just four, they may also be fit from prior student data for each knowledge component in a domain, using expectation maximization or other techniques [5][6]. While this model is simple and has been used in many tutors, the main disadvantage of this method is that it relies on a definition of student performance in terms of number of incorrect attempts –it does not address how students help requests (via glossaries or hint requests) or issues of timing affect performance. This fact has been addressed in later work [2][3][5][6][7] by creating separate models of engagement or more complex Bayesian models that address the impact of help on student knowledge. The problem that still remains is that issues of timing or hints are not addressed in an *integrated* way within these knowledge estimation models, leading to biased estimations of knowledge (e.g., the student answers very fast incorrectly may lead to an estimation of unknowing, however, it really reflects disengagement). One attempt was [7], who modeled student engagement and knowledge at the same time within one model. It helped to keep knowledge more stable instead of an apparently decreasing knowledge, as if students were unlearning while they were actually disengaged. This work encoded math knowledge as single latent variable, though, which is not practical to make decisions in the pedagogical model, so this work is still preliminary.

In summary, while progress has been made in student modeling and intelligent learning environments, there is a need for student models that *integrate* and discern between engagement, student knowledge and other factors such as affect and meta-cognition, or descriptions about how to juggle different models to make optimal pedagogical decisions. This paper provides one approach that discerns among the reasons for student effort (or not) at individual practice items, based on different

dimensions of student behavior. We then thoroughly document a pedagogical model, which is heavily based on empirical estimates of student effort and problem difficulty. We last provide results of a randomized controlled study that shows the adaptive nature of this tutor does improve learning, compared to an “unintelligent” version that makes less smart moves when selecting practice activities. We also provide a methodology to evaluate that the estimates of problem difficulty are accurate. The level of detail in the methodology allows for replication of the tutoring mechanism and estimates of effort on other learning environments, even for systems without a large amount of content available, or ill-defined content (ill-defined domains).

1.1 Wayang Outpost: A Mathematics Tutoring System

Wayang Outpost is a software tutor that helps students learn to solve standardized-test type of questions, in particular for a math test called the Scholastic Aptitude Test, and other state-based exams taken at the end of high school in the USA. This multimedia tutoring system teaches students how to solve geometry, statistics and algebra problems of the type that commonly appear on standardized tests. To answer problems in the Wayang interface, students choose a solution from a list of multiple choice options, providing immediate feedback on students’ entries and offering hints that students can accept or reject. Students are encouraged to ask the tutor for hints that are displayed in a progression from general suggestions to bottom-out solution. In addition to this domain-based help, the tutor currently provides a variety of affective and meta-cognitive feedback, delivered by learning companions designed to act like peers who care about a student's progress and offer support and advice [1][8]. Both decisions about content sequencing and characters response are based on a model of student effort, used to assess the degree of cognitive effort a student invests to develop a problem solution, described in the next sections.

2 Modeling and Acting Upon Student Effort

We start by estimating the expected behavior that a student should have on a problem based on three indicators of effort: 1) number of attempts to solve a problem; 2) number of hints requested for a problem; 3) time required to solve a problem. These are three orthogonal axes that help understand student effort. The only pre-processing done for this data set was to use data corresponding to “valid” student users (instead of test users), and discarding outliers just for the “time” variables.

Figure 1 shows examples of problem solving behavior for nearly 600 students in one problem. This one problem may seem evidently too easy at first glance, as the majority of students made zero or few incorrect attempts, saw no hints, and solved the problem in less than 5 seconds. However, this is not the case. It is common to find problem-student interaction instances where students spend little time and effort. It is also common that students under-use the help in the system. We find it essential to take into account that this is the real way that students use the tutoring system, and we need to take into account what are likely student behaviors when considering how to adjust instruction and the presentation of the material to students. Note that the distributions are not normal, but more similar to Chi-Square distributions.

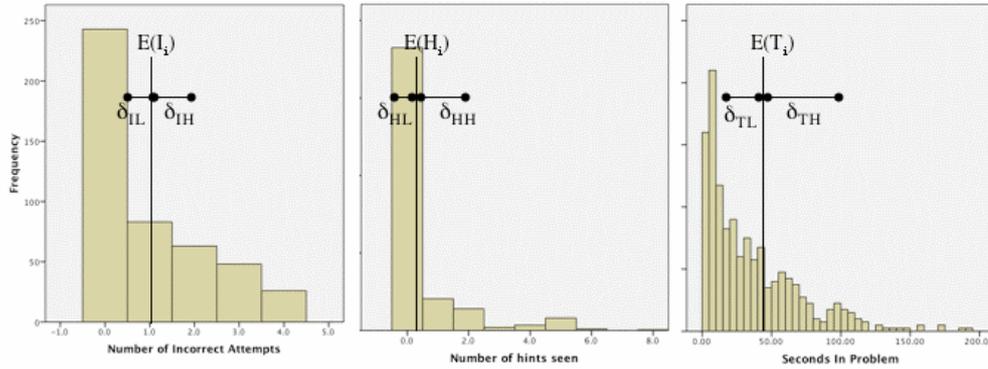


Figure 1. Distribution of attempts, hints and seconds in one problem. Expected and delta values.

The combination of mistakes, hints and time as shown in Figure 1 will allow to estimate higher-level scenarios of mastery or disengagement, see Table 1. For each of the hundreds of problems or practice items in an intelligent tutor, we compute the median (or the sample mean after discarding the top 10 percentile, which was a good approximation in our data and much easier to compute using SQL) and standard deviation for the whole population of students. This median or mean is considered the expected value, i.e. the expected number of incorrect attempts for a problem p_i ($E(I_i)$) where $i=1\dots N$, and N =total practice items in the tutoring system. Expected hints seen is $E(H_i)$ and time required to solve the problem is $E(T_i)$. We also define two delta values for each $E(I_i)$, $E(H_i)$ and $E(T_i)$, a total of six delta values (see Figure 1) for each problem p_i , which represent a fraction of the standard deviation, regulated by two parameters, θ_{LOW} and θ_{HIGH} in the interval $[0,1]$. For example, if $\theta_{LOW}=1/4$ and $\theta_{HIGH}=1/2$, then $\delta_{IL}=\theta_{LOW}SD(I_i)=SD(I_i)/4$ (a fourth of the standard deviation of I_i) and $\delta_{IH}=SD(I_i)\theta_{HIGH}=SD(I_i)/2$, half of the standard deviation of I_i . θ_{LOW} and θ_{HIGH} are the same for all problems in the system. These values help define what is “expected behavior” for a practice item within the tutoring system. Note that the notation for δ values has been simplified (e.g. δ_{IL} should really be $\delta_{i,L}$, as it refers to an individual practice item).

2.1 Pedagogical Decisions based on Student Effort

The large benefit of an effort model based on different orthogonal axes of behavior (hints, time and correctness) is that it can help researchers discern between behaviors related to student engagement (affective) and behaviors related to help misuse (meta-cognitive or affective) in addition to behaviors related to cognitive mastery. Table 1 shows the estimations of most likely scenarios made by the pedagogical model in Wayang Outpost, and the pedagogical decisions made in terms of content difficulty, plus other pedagogical moves related to affective and meta-cognitive feedback. Note that disengagement (e.g. lines 3 and 5) produces a reduction in problem difficulty, based on the assumption that if a student is not working hard enough on the current problem, they probably won’t work hard on a similar or harder problem. However, the key intervention is that Learning Companions deemphasize the importance of immediate success.

Table 1. Empirical-based estimates of effort at the recently completed problem lead to adjusted problem difficulty and other affective and meta-cognitive feedback

Student Model Estimate most likely scenario for student on problem i				Pedagogical Model Moves Cognitive or Affective or Metacognitive	
Mistakes	Hints	Time	Most Likely	Decision	Other Actions
1	$< E(I_i) - \delta_{IL}$	$< E(H_i) - \delta_{HL}$	$< E(T_i) - \delta_{TL}$	Mastery without effort	Increase Problem Difficulty Show learning progress
2	$< E(I_i) - \delta_{IL}$	$< E(H_i) - \delta_{HL}$	$> E(T_i) + \delta_{TH}$	Mastery with high effort	Maintain Problem Difficulty Affective feedback: Praise Effort
3	$< E(I_i) - \delta_{IL}$	$> E(H_i) + \delta_{HH}$	$< E(T_i) - \delta_{TL}$	Hint abuse, low effort	Reduce Problem Difficulty Deemphasize importance of immediate success
4	$< E(I_i) - \delta_{IL}$	$> E(H_i) + \delta_{HH}$	$> E(T_i) + \delta_{TH}$	Towards mastery, effort	Maintain Problem Difficulty Praise effort
5	$> E(I_i) + \delta_{IH}$	$< E(H_i) - \delta_{HL}$	$< E(T_i) - \delta_{TL}$	Quick guessing, low effort	Reduce Problem Difficulty Deemphasize importance of immediate success
6	$> E(I_i) + \delta_{IH}$	$< E(H_i) - \delta_{HL}$	$> E(T_i) + \delta_{TH}$	Hint avoidance and high effort	Reduce Problem Difficulty Offer hints upon incorrect answer in the next problem
7	$> E(I_i) + \delta_{IH}$	$> E(H_i) + \delta_{HH}$	$< E(T_i) - \delta_{TL}$	Quick guess and hint abuse	Reduce Problem Difficulty Deemphasize importance of immediate success
8	$> E(I_i) + \delta_{IH}$	$> E(H_i) + \delta_{HH}$	$> E(T_i) + \delta_{TH}$	Low mastery and High Effort	Reduce Problem Difficulty Emphasize importance of effort and perseverance
9	Otherwise			Expected Behavior	Maintain Problem Difficulty

The retrieval of an increased difficulty item is based on a function $Harder(H[1..n], \gamma)$ that returns a problem of higher difficulty; H is a sorted list of n practice items the student has not yet seen, all harder in difficulty than the one the student has just worked on; $H[1]$ is the item of lowest difficulty, and $H[n]$ is the item of highest difficulty, and γ is a natural number greater than zero. The problem returned by $Harder$ is specified in Eq. 1. For example, $Harder$ with $\gamma=3$ will return the problem at the 33rd percentile of items in list $H[1..m]$.

$$Harder(H[1..m], \gamma) = H \left[\text{ceiling} \left(\frac{m}{\gamma} \right) \right] \quad (1)$$

Similarly, a problem of lesser difficulty is selected with function $Easier(E[1..n])$, where E is a sorted list of problem items, all items are easier than the problem just seen by the student; $E[1]$ is the item of lowest estimated difficulty, and $E[n]$ is the item of highest difficulty. Eq. 2 shows $Easier$ as a function of n and γ . $Easier$ with $\gamma=3$ will return the item that at the 66th percentile of items in list $E[1..n]$.

$$Easier(E[1..n], \gamma) = E \left[\text{ceiling} \left(n - \frac{n}{\gamma} \right) \right] \quad (2)$$

Both *Easier* and *Harder* work upon the assumption that there are easier or harder items to choose from. The next section addresses what happens when $m=0$ or $n=0$.

2.2 Progression through Knowledge Units

In Wayang Outpost, the curriculum is organized in a linear set of topics or knowledge units (KU), which is a classification of problems in sets of items that involve similar skills (e.g. polygon perimeter measurement problems). Pedagogical decisions about content sequencing are made at two levels: within a topic and between topics, skills or knowledge units. This section addresses between topic decisions.

The criteria of “chunking” problems in knowledge units is based on the idea that similar problems should be seen close to each other, to maximize the transfer of what a student has learned, as the concepts are still in working memory to be applied to the next cognitive transfer task. Cognitive effort is then reduced, and the likelihood of applying a recently learned skill to the next task is enhanced.

Each knowledge unit may be defined at a variety of levels, and is composed of a variety of problems involving a set of related skills. For instance, within the “Statistics” topic, a student may be presented with problems about finding the median of a set of numbers, or deciding whether the mean or median were larger, from a picture of a stem and leaf plot. While overlap of skills exists, not all problems within a topic involve the same skills, and their difficulties may vary to a large degree.

Topics are arranged according to pre-requisites (problems presented in KU2 will not include skills introduced in KU3). When a topic begins students are presented an explanation of the kinds of problems that will follow, generally introduced aloud by pedagogical animated creatures. Sometimes this involves an example problem, accompanied by a worked-out solution via multimedia features.

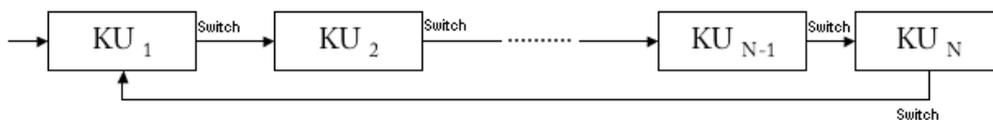


Figure 2. Spiral curriculum in which Knowledge Units are ordered according to pre-requisites

Table 2. Conditions for topic switching in Wayang Outpost

Topic Switch Criterion	Reason	Parameter
2.1 Topic Mastery was reached (e.g. enough “hard” problems answered correctly)	Cognitive	M_{KU}
2.2 Persistent failure to find a problem of desired difficulty	Content limitation	F_{KU}
2.3 Maximum time in Topic condition, or Maximum Number of Problems allowed	Classroom Implementation	T_{KU} N_{KU}

A student progresses through these knowledge units depending on a variety of criteria, specified in Table 2 beyond cognitive mastery. For instance, condition 2.2 shows how a topic switch may be forced based on limitations of content --the system failed F_{KU} times to find a problem of the difficulty it believes the student should get for the topic. If the pedagogical model suggests the student should increase problem difficulty, but there are no harder problems remaining, then a counter for the number of failures for the current topic is increased. Because failures $< F_{KU}$ an easier problem is provided instead. Another possibility is condition 2.3, where the teacher has allocated a specific amount of time for the student to study or review a certain topic.

2.3 Problem Difficulty Estimates

The pedagogical model must be able to estimate problem difficulty in order to assign problems for students in specific scenarios. We identify two faces of problem difficulty in intelligent tutors. From the perspective of a knowledge engineer, problems have *objective difficulty* (e.g., based on number of skills and steps involved in each problem). However, students may perceive each problem differently according to a *student perceived difficulty* (SPD). While objective problem difficulty should be similar to SPD, they are not necessarily the same. Proper estimation of problem difficulty is essential for this pedagogical model, and not possible to do with simple Item Response Theory because tutoring involves more dimensions (help, engagement) than testing (accuracy). We capture SPD from the three independent sources of evidence of students' effort to solve a problem: 1) correctness in term of number of required attempts to solve a problem (random variable C_i); 2) amount of time spent in a problem (random variable T_i); 3) amount of help required or requested to solve the problem correctly (random variable H_i).

We define problem difficulty d_i for a practice activity component i in Eq. 3, as the mean of these three factors: attempts to solve, time and help needed.

$$d_i = \text{mean}(dc_i, dt_i, dh_i) \quad (3)$$

Where dc_i is the difficulty factor in terms of correctness, dt_i is the difficulty factor in terms of time, and dh_i is the difficulty factor in terms of help needed. Alternatively, the three factors might be given a weight, to emphasize them differently.

d_i , dc_i , dt_i and dh_i are normalized values in the interval [0,1] and express SPD. Eq. 4, 5 and 6 show how each of the three difficulty factors are computed.

$$dc_i = \frac{E(I_i)}{\text{Max}_{j=1}^N(E(I_j))} \quad (4)$$

$$dt_i = \frac{E(T_i)}{\text{Max}_{j=1}^N(E(T_j))} \quad (5)$$

$$dh_i = \frac{E(H_i)}{\text{Max}_{j=1}^N(E(H_j))} \quad (6)$$

dc_i (Eq. 4) is the expected value of I_i (number of incorrect attempts while trying to solve a problem p_i) across all students who have seen that problem, divided by the

maximum $E(I_j)$ registered for any problem p_j in the system (N =the total number of problems or practice activities in the system).

Similarly, dt_i (Eq. 5) is the expected value of T_i (time spent on problem p_i) and is also normalized. This expected time is the mean value after removing outliers, or median.

dh_i (Eq. 6) is the expected value of H_i (number of attempts for problem p_i) divided by the maximum $E(H_j)$ registered.

2.4 Accuracy of Item Difficulty Estimations

We computed SPD estimates using a data set of 591 high school students who used Wayang Outpost tutoring software over past years, from 2003 until 2005. The tutors employed a variety of problem selectors during those years, with some percentage of students using a random problem selector.

Validating that student perceived difficulty estimates were reasonable seemed essential. The first reason is that the difficulties play a crucial role in the adaptive behavior of the tutor, and inappropriate difficulties would make the system behave in undesired ways (e.g. providing a harder problem when the student clearly needs an easier one). The second reason is that it is just too likely that the student perceived difficulty estimates are biased, because student behavior is contingent to the problem selector in place at the moment the data on problem performance was collected. Unless the raw data comes from a random selection of problems, student behavior and thus the data collected will be biased in some direction. This will make problems look easier or harder than they truly are.

We devised a variety of methods to assess the correctness of our estimation of perceived student difficulty, and implemented three of them. All of these are based on the following axiom: “*Pairs of Similar Problems Should have Similar Problem Difficulty Estimates*”. In other words, if two problems are very similar, the perceived differences in their difficulty should approach zero. We subsequently drew a subset of 60 mathematics problems (p_1 to p_{60}) from our tutoring system. These sixty problems are special because may be divided into 30 pairs of problems, where each p_i , with $i=1 \dots 30$, is extremely similar to p_{30+i} . In this domain of geometry problems, similar problems involved similar showing graphics with slightly different angles, or measurements. For example, same problems with a rotated figure (and different operands). Similar problems involve the application of the same skills the same amount of times. We call these *highly similar pairs* and now describe four criteria used to verify that these pairs are similar in their difficulty estimates.

2.4.1 Criteria 1: Correlations. We tested that such pairs had similar difficulty estimates with a simple Pearson correlation, which is the most familiar measure of dependence between two quantities. It is obtained by dividing the covariance of the two variables (d_{p_i} and $d_{p_{30+i}}$) by the product of their standard deviations. A Pearson correlation determined that pairs of problems were significantly correlated ($N=30$, $p<0.000$, $R=.823$), thus this test is passed.

2.4.2 Criteria 2: Mean Squared Error. Another criteria used was that the difference in objective difficulty between highly similar problems should be smaller than the difference in difficulty between either of these problems and any other problem in the

system that is not as similar –other problems will involve different skills, or different total amount of applications of the same skills. While it may be coincidental that a problem foreign to the pair might have a very similar difficulty to either problem in the pair, this should not be the general case.

The distance between the difficulty of a problem p_i and its highly similar problem pair p_{30+i} should be smaller than the mean distance between one of the problems in the pair and the remaining problems in the set. A more common jargon when talking about differences due to error is the mean squared error. Eq. 7 rephrases the above in terms of squared differences, where $N=total\ number\ of\ pairs=30$.

$$\left(d_{p_i} - d_{p_{30+i}}\right)^2 < \frac{\sum_{j=1, j \neq i}^N (d_{p_i} - d_{p_j})^2}{N} \quad (7)$$

If we can show that this inequality holds in general for problems, we have some evidence that our system is doing a reasonable job at estimating difficulties. We computed the 30 square differences $\left(d_{p_i} - d_{p_{30+i}}\right)^2$, and their corresponding mean squared differences as specified in Eq. 7. The result was that the inequality holds for 29 of the thirty cases, which is a 97% success rate. A paired-samples t-test for the two inequality terms in Eq. 6 revealed that these two sides of Eq. 7 are significantly different $t(29)=7.35, p<.000$. The second test is then passed.

2.4.3 Criteria 3: Human Expertise. While pairs of highly similar problems should have similar student perceived difficulty levels, they don't necessarily need to have exactly the same difficulty (i.e. the difference in their difficulty levels will not be exactly zero. In other words, $\left(d_{p_i} - d_{p_{30+i}}\right)^2 = \epsilon_i$, where ϵ_i is a small number. While it would be hard to determine the true value of epsilon for each problem pair, an expert human eye (e.g. a teacher or tutor) could probably make good predictions about whether $d_{p_i} > d_{p_{30+i}}$ or whether $d_{p_i} < d_{p_{30+i}}$. This kind of expert knowledge can help us establish that the latter problem should be harder for a student to solve than the former one. Other restrictions may have to do with operand size, involvement of decimals or negative numbers, or a small extra step. We managed to establish such restrictions for 21 of the 30 pairs of problems we considered, the other 9 were just too similar to each other. Such restrictions (true positives or true negatives) were correctly guessed in 14 of the 21 cases (67%), and a Chi-Square test revealed this is significantly better than chance (Pearson Chi-Square=5.25, $p=.022$). Thus, the third test is passed.

2.4.4 Criteria 4: Convergence. Ideally, the difference between highly similar pairs of problems would converge to ϵ_i as more data arrives to the logs, even if different problem selectors are in place at different moments. This test is still ongoing.

2.5 Evaluation of Effectiveness of Effort-Based Pedagogical Model

While we may be satisfied that difficulty of items are reasonably estimated, we need also to show that the adaptive mechanism underlying the pedagogical model makes a difference to student learning. A study was carried out in the 2003-2004 academic year with 60 students to evaluate the effectiveness of the adaptive sequencing of problems, compared to a random selection of problems within a topic (no learning companions or affective feedback).

Table 3. Pretest and Posttest Scores in Math Test

		Pretest score SAT questions (correct/attempt)	Posttest score SAT (correct/attempt)	Pretest correct SAT	Posttest correct SAT
Adaptive/random					
	Random problem selection	Mean .3868	.4220	2.7576	3.3030
		N 33	33	33	33
	Std. Deviation .25160	.19730	1.87133	1.51007	
Adaptive problem selection	Mean	.4216	.5664	3.5217	4.9565
	N	23	23	23	23
	Std. Deviation	.23227	.19108	2.06419	1.77042
Total	Mean	.4011	.4813	3.0714	3.9821
	N	56	56	56	56
	Std. Deviation	.24230	.20590	1.97122	1.80395

Both the experimental and the control conditions implemented topic switching based on one parameter only, N_{KU} , so that the “topic switch” criterion was set to a fixed maximum number of problems per topic. This was established so that all students were exposed to the same number of problems in each topic. M_{KU} , F_{KU} and T_{KU} were then ignored. The main difference between conditions was the problem selection mechanism *within* the topic. For the experimental condition, it adjusted problem difficulty as described in previous sections, with the following parameters: $\gamma=2$; $\theta_{LOW}=0$; $\theta_{HIGH}=0$; this made the changes in problem difficulty quite marked. Control condition students received random problems within each topic. Students were randomly assigned to either the Effort-based Adaptive Problem

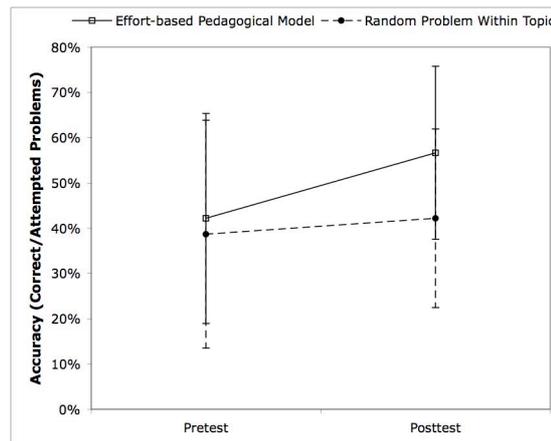


Figure 3. Pre to Posttest Improvement with Effort-based Pedagogical Model compared to a Random Problem Selector Within the Topic.

Selection condition, or the Random Problem Selection Condition. Students used the Wayang Tutoring System for 4 class periods, completing a 10-item math test before starting and a similar posttest the last day. The test consisted of items drawn from the SAT (Scholastic Aptitude Test) and released by the College Board. The two tests were counterbalanced –half of students received pretest A, and half pretest B, and the tests were reversed for students at posttest time.

We measured the total number of correct items achieved in the test, and the accuracy at items (correct/test items attempted) as a measure of performance, see Table 3. We obtained full pre and posttest data for 56 students, 23 in the experimental adaptive condition, and 33 in the control condition. Table 3 shows the mean and standard deviation of pretest and posttest scores for the pretest and the posttest. Mean achievement in the posttest increased and standard deviations reduced for both groups. However, mean improvement was higher for the experimental adaptive

problem selection group (Figure 3). This difference is significant (ANCOVA for posttest score with pretest score as a covariate, group effect $F(55,1)=8.4, p=.006$). The group receiving adaptive effort-based pedagogical decisions about problem difficulty improved more than did the group receiving random problem selection control condition. We conclude that adaptive problem selection is better than random.

3 Summary

This paper presented a novel approach to the development of smart learning environments, based on empirical measures of student effort at individual items. It described a pedagogical model that uses empirical estimates of problem difficulty, specifying parameters that regulate behavior within knowledge units ($\gamma, \theta_{\text{LOW}}$ and θ_{HIGH}) and between knowledge units ($M_{KU}, F_{KU}, T_{KU}, N_{KU}$). Knowledge Units may be defined at different levels of abstraction, thus addressing restrictions of content. This allows for replication in other ILEs, even in ill-defined domains or in small ILEs that are trying to encode smart decisions about practice items or activity selection.

We have described criteria for evaluating that estimates of problem difficulty are not too biased to the problem selector in place at the time of data collection. Last, we have shown that this effort-based pedagogical model leads to improved learning compared to uninformed random decisions within a topic or knowledge unit.

References

- [1] Arroyo, I., Woolf, B.P., Royer, J.M., Tai, M.: Affective Gendered Learning Companions. *Proceedings of AIED conference*, 2009. p. 41-48. IOS Press.
- [2] Baker, R.S.; Corbett, A.T.; Koedinger, K. Detecting Student Misuse of Intelligent Tutoring Systems. *Proceedings of ITS Conference*. LNCS 3220, 2004, p. 54-76
- [3] Beck, J.E. Engagement tracing: using response times to model student disengagement. *Proceedings of AIED conference*, 2005. p. 88-95. IOS Press
- [4] Corbett, A.T., Anderson, J.R., Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 1995, 4, p.253-278.
- [5] Johns, J. and Woolf, B.P. A Dynamic Mixture Model to Detect Student Motivation and Proficiency. *Proceedings of AAAI Conference*, 2006, 1, p. 163-168.
- [6] Ferguson, K.; Arroyo, I., Mahadevan, S., Woolf, B.P., Barto, A. Improving Intelligent Tutoring Systems: Using Expectation Maximization To Learn Student Skill Levels. *Proceedings of ITS conference*. LNCS 4053, 2006. p. 453-462.
- [7] Mayo, M., Mitrovic, A. Optimising ITS Behaviour with Bayesian Networks and Decision Theory. *International Journal of Artificial Intelligence in Education*, 2001, 12, p. 124-153.
- [8] Woolf, B.P. , Arroyo, I., Woolf, B., Muldner, K., Burleson, W., Cooper, D., Dolan, B., L., The Effect of Motivational Learning Companions on Low-Achieving and Learning Disability Students. *Proceedings of ITS conference*, 2010. Springer.

An Analysis of the Differences in the Frequency of Students' Disengagement in Urban, Rural, and Suburban High Schools

Ryan S.J.d. Baker, Sujith M. Gowda
rsbaker@wpi.edu, sujithmg@wpi.edu

Department of Social Science and Policy Studies, Worcester Polytechnic Institute

Abstract. We study how student behaviors associated with disengagement differ between different school settings. Towards this, we investigate the variation in the frequency of off-task behavior, gaming the system, and carelessness in an urban school, a rural school, and a suburban school in the United States of America. This analysis is conducted by applying automated detectors of these behaviors to data from students using the same Cognitive Tutor educational software for high school Geometry, across an entire school year. We find that students in the urban school go off-task and are careless significantly more than students in the rural and suburban schools. Differences between schools in terms of gaming the system are less stable. These findings suggest that some of the differences in achievement by school type may stem from differences in engagement and problem behaviors.

1 Introduction

In recent years, intelligent tutoring systems have left the research laboratory, expanded beyond the research classroom, and have started to see large-scale use worldwide. To give two examples, Cognitive Tutors for high school Algebra and Geometry [cf. 14, 15] are now used by hundreds of thousands of students each year in the United States, and constraint-based tutors for SQL and database normalization are now used by tens of thousands of students each year worldwide [20]. Aplusix, another system that has seen particularly wide use, is now in use in at least 6 countries [cf. 21]. As the reach of intelligent tutors increases, they become available to an increasingly diverse population of students. Tutors are now used by students of very different economic and ethnical backgrounds, by students in suburbs, cities, and rural areas, by wealthy, middle-class, and poor students, by students from ethnic majority groups and students from minority groups [cf. 8, 14, 15, 23].

As the use of intelligent tutors spreads to a more diverse selection of populations, we gain the potential to use intelligent tutors as a research tool in yet another domain of educational research – comparisons of student learning and behavior between learners in schools with different demographic profiles. There is increasing evidence that students in different learning settings have radically different learning outcomes [cf. 11], but there is insufficient understanding about what factors mediate those learning outcomes. There are many differences between schools in different settings, including differences in teacher expertise [17], in the physical conditions of schools, and in students' backgrounds.

An additional possibility is that the differences in learning in schools may be influenced, at least in part, by disengagement. Given the known relationships between disengaged

behaviors and learning (see [13] for an early review of this literature in traditional classrooms; see [2, 6, 10, 12, 22] for studies on this topic in classrooms using educational software), there is valid reason to think that disengaged behavior may mediate the differences in learning between different school settings. However, it has not yet been established whether there are significant differences in disengaged behavior between types of schools.

In this paper, we focus on this topic, studying how three types of student behavior associated with disengagement differ across different school settings. In specific, we investigate the variation in gaming the system, off-task behavior, and carelessness between urban, rural, and suburban classrooms in the United States of America, using “discovery with models” methods on data from students using the same Cognitive Tutor across an entire school year. Although there have been many studies on off-task behavior within rural, urban, and suburban settings ([13] offers an early review of this literature), we are not aware of any systematic comparisons of off-task behavior *between* types of schools. Similarly, we are not aware of any studies on gaming the system that span urban, rural, and suburban schools, and are not aware of systematic attempts to measure carelessness in any type of school prior to the model advanced in [3].

Studying these issues in the context of Cognitive Tutors several advantages. First, Cognitive Tutors have been deployed to a wide variety of settings, and learning gains have been demonstrated in urban, rural, and suburban schools [7, 8, 15, 23]. Second, Cognitive Tutors collect extensive logs of every student answer or help request within the software [16]. For Cognitive Tutors, sufficient log data to study these research questions is currently available in the Pittsburgh Science of Learning Center (PSLC) DataShop [16], in a format compatible with existing detectors of disengaged behaviors [2, 3, 4, 5].

In this paper, we utilize data from the PSLC DataShop in order to compare USA schools with 3 very different profiles (urban, suburban, and rural), according to 3 metrics (percent of time off-task, percent of time spent gaming the system, and average slip probability). We do so in the context of an entire year of use of the exact same Cognitive Tutor for Geometry.

2 Methods

Data from 3 schools was obtained through the PSLC DataShop [16] in order to compare students’ behavior within urban, rural, and suburban settings, across an entire school year. While a sample of 3 schools is clearly not enough data to be able to draw conclusions about all schools from these categories, it enables us to statistically compare the behavioral profile of 3 schools that are representative of each of these categories. This study also provides a methodological template for future expanded investigations of these issues, which can be conducted as data becomes available from a broader range of schools.

The three schools studied are each public high schools in Southwestern Pennsylvania. As shown in Table 1, each school has a distinctive demographic profile; the urban school is overwhelmingly African-American, whereas the suburban and rural school are

Table 1. School Population Demographics

	Urban school	Suburban school	Rural school
% African-American	100%	<1%	2%
% White	0%	98%	97%
% Hispanic	0%	<1%	<1%
Free or Reduced-Price Lunch	99%	4%	Not Reported
% Proficient on state math exam	20%	77%	Not Reported
Median household income in community	\$26,621	\$60,307	\$32,206
% Children under poverty line	30.8%	2.5%	18.4%

overwhelmingly White Non-Hispanic (the ethnic majority group in the region). There is evidence of considerable poverty in the urban school's population, with a low median household income for the United States, and a high percentage of children under the poverty line. Similarly, the rural school has a low median household income for the United States, and a substantial percentage of children under the poverty line. The suburb has considerably less poverty, with a median household income over double the urban school's median household income, and only 2.5% of children under the poverty line.

Data from each school was obtained through the PSLC DataShop [16]. In each case, high school students took their Geometry courses using the same Cognitive Tutor for Geometry [7, 14, 15], and all data was collected in the PSLC DataShop. Data was collected for the entire school year, from August 2005 to May 2006. 434 students in the rural school used the software, 88 students in the suburban school used the software, and 34 students in the urban school used the software. The three schools each assigned the software to students who were neither in special needs nor gifted classes – the difference in the number of students using the software is solely based on school size, and how many teachers chose to use the software (in particular, the rural school is a large regional school, as is increasingly common in the USA in rural areas, whereas the urban and suburban schools serve smaller populations).

In all schools, the software was used by groups of students in a computer laboratory, working individually at separate computers, at their own pace. Students in the rural school used the software an average of 9 hours, students in the suburban school used the software an average of 35 hours, and students in the urban school used the software an average of 51 hours. Hence it appears that the teachers in each school chose to have their students use the software in different amounts. This difference represents a selection bias in our data, but it is a difficult confound to resolve; for instance, restricting analysis to students who used the software above a time cutoff introduces a different selection bias. In particular, the difference in usage is a natural one, reflecting genuine implementation in each type of school.

To address this selection bias stemming from teacher choice, we analyze the data in two ways – using all data (the more ecologically valid choice), and using a time-slice consisting of the 3rd-8th hours (minutes 120-480) of each student’s usage (this time-slice will not be as representative of the usage in each school, but avoids this confound). The 3rd-8th hours were selected, because the initial 2 hours likely represent interface learning, and therefore may not be representative of overall tutor use (and interface learning is likely to be dependent on prior experience with educational software, which is likely to be greater for wealthier students). In general, implementational differences between schools are likely to exist in year-long comparisons. In the long-term, this problem can probably be best addressed by conducting analyses of this nature across large numbers of schools, in order to average across implementational differences orthogonal to the type of school (though some implementational differences may be characteristic to certain types of schools – for instance, urban schools might use specific pieces of educational software more heavily due to having lower resources to provide a wide range of educational software in their classrooms).

Each action in each data set was labeled using detectors of gaming the system, off-task behavior, and carelessness. The gaming detector used was trained using data from students using a Cognitive Tutor for Algebra [5], using an age-similar population and an approach validated to generalize between students and between Cognitive Tutor lessons [4]. The off-task detector used was trained using data from students using a Cognitive Tutor for Middle School Mathematics. The off-task detector was validated to generalize to new students, and to function accurately in several Cognitive Tutor lessons [2]. Although the age range was moderately older in this study than in the original training data, off-task behavior is similar in nature within these populations – it involves ceasing to use the software for a significant period of time without seeking help (which can be detected in the log files by the behavior occurring before and after an idle pause). Carelessness was detected using the slip detector from [3], which was trained on data from Cognitive Tutor Geometry. This use of contextual slip is in line with theoretical work by Clements [9], who argues that making errors despite knowing the skills needed for successful performance should be considered evidence of carelessness. It is important, however, to note that contextual slip could potentially also be an indicator of shallow knowledge that does not apply to all items in the tutor, even if they are labeled as involving the same skill.

3 Results

In discussing results, we will first discuss our analyses conducted across the full year of tutor data, and then discuss the same analyses conducted across only a time-slice including the 3rd to 8th hours of tutor usage.

3.1 Analyses Across Full Data Set

Across the full data set, representing data collected during the entire school year, the pattern of off-task behavior was highly different between the three schools. Students in the suburban school were off-task an average of 15.4% of the time (past research in traditional classrooms has averaged 15-20% of time off-task [cf. 18, 19]). Students in the

Table 2. Average incidence of each indicator per school. Parentheses give standard deviation.

	Urban school	Suburban school	Rural school
% Off-Task	34.1% (18.0%)	15.4% (20.7%)	20.4% (13.3%)
% Gaming the System	7.4% (2.2%)	6.9% (3.1%)	6.6% (1.7%)
% Slip Probability	0.50 (0.07)	0.32 (0.11)	0.27 (0.13)

rural school were off-task an average of 20.4% of the time. Students in the urban school were off-task an average of 34.1% of the time. Hence, students at the urban school were off-task 67% more than students at the rural school (a 1.0 SD difference), and over double as much as students at the suburban school (a 0.9 SD difference). The overall difference in off-task behavior between schools was statistically significant between schools, $F(2,553)= 18.80$, $p<0.01$. The model predicting time off-task by school predicted 6.4% of the variance in time off-task. The pairwise differences between schools were all statistically significant, using Tukey’s HSD to control for multiple comparisons.

The frequency of gaming the system had smaller differences between the three schools, although there were still significant differences. Students in the suburban school gamed 6.9% of the time, students in the rural school gamed 6.6% of the time, and students in the urban school gamed 7.4% of the time. In other words, students in the urban school gamed only 13% more than students in the rural school (a 0.47 SD difference), and 9% more than students in the suburban school (a 0.16 SD difference). The overall difference in gaming the system between schools was statistically significant, $F(2,553)= 3.12$, $p=0.05$. The model predicting time spent gaming the system by school predicted 1.1% of the variance in gaming, considerably less than is predicted by individual differences between students or by the differences between tutor lessons [1]. According to Tukey’s HSD, the rural school had significantly less gaming than the urban school, but the other differences in gaming were not statistically significant.

The pattern of carelessness was highly different between the three schools. Students in the suburban school had a probability of 0.32 of slipping despite knowing a skill, students in the rural school had a probability of 0.27 of slipping despite knowing a skill, and students in the urban school had a probability of 0.50 of slipping despite knowing a skill. The overall difference in slipping between schools was statistically significant, $F(2,553)= 54.50$, $p<0.001$. The model predicting slip probability by school predicted 16.5% of the variance in slip probability. The pairwise differences between schools were all statistically significant, using Tukey’s HSD to control for multiple comparisons.

3.2 Analyses Across Data From Hours 3-8

Within the restricted time-slice of data from hours 3-8, the differences in the frequency of off-task behavior were qualitatively similar to the analysis across the full data set, although the difference between the urban school and the other schools was smaller.

Table 3. Average incidence of each indicator per school. Parentheses give standard deviation.

	Urban school	Suburban school	Rural school
% Off-Task	25.7% (22.8%)	16.5% (27.5%)	21.0% (16.5%)
% Gaming the System	4.7% (1.9%)	5.9% (7.3%)	6.4% (2.2%)
% Slip Probability	0.53 (0.08)	0.44 (0.17)	0.33 (0.18)

Students in the suburban school were off-task an average of 16.5% of the time, very similar to the 15.4% reported across all data. Students in the rural school were off-task an average of 21.0% of the time, very similar to the 20.4% reported across all data. However, students in the urban school were off-task an average of 25.8% of the time, substantially lower than the 34.1% reported across all data, a statistically significant difference, $t(33)=-2.55$, $p=0.02$, for a two-tailed paired t-test. This result suggests that off-task behavior increased during the year in the urban school.

Nonetheless, even during this earlier time-slice, off-task behavior was higher in the urban school than the suburban school. The overall difference in off-task behavior between schools was statistically significant, $F(2,484)= 3.01$, $p=0.05$. The model predicting time off-task by school predicted 1.2% of the variance in time off-task. According to Tukey’s HSD, the urban school had significantly more off-task behavior than the suburban school, but the rural school was not significantly different from either of the other two schools.

The pattern of gaming the system was highly different within the restricted time-slice of data from hours 3-8, as compared to the entire data set: Gaming the system was much rarer in the urban school. Students in the urban school gamed 4.7% of the time in the restricted time-slice, compared to 7.4% of the time in the full data set, a significant difference, $t(33)=8.14$, $p<0.001$, for a two-tailed paired t-test. Gaming was also less common in this time-slice in the other two schools, but to a much lower degree. Students in the suburban school gamed 5.9% of the time, compared to 6.9% of the time in the full data set, which was not quite statistically significant, $t(71)=1.51$, $p=0.13$, for a two-tailed paired t-test. Students in the rural school gamed 6.4% of the time, compared to 6.6% of the time in the full data set.

The overall difference in gaming the system between schools was statistically significant, $F(2,484)= 4.09$, $p=0.02$. The model predicting time spent gaming the system by school predicted 1.7% of the variance in gaming, considerably less than is predicted by individual differences between students or by the differences between tutor lessons [1]. According to Tukey’s HSD, the rural school had significantly more gaming than the urban school – the exact opposite of the result across the entire data set – but the other differences in gaming were not statistically significant.

The pattern of carelessness retained the same ordering within the restricted time-slice of data from hours 3-8, and the entire data set, but the degree of carelessness was significantly higher for all three groups of students, $t(71)=6.32$, $p<0.001$ in the suburban school, $t(374)=10.08$, $p<0.001$ in the rural school, and $t(33)=2.04$, $p=0.05$ in the urban school. In all cases, a two-tailed paired t-test was used.

The overall difference in slipping between schools was statistically significant, $F(2,478)=29.78$, $p<0.001$. The model predicting slip probability by school predicted 11.1% of the variance in slip probability. The pairwise differences between schools were again all statistically significant, using Tukey's HSD to control for multiple comparisons.

4 Discussion and Conclusions

In this paper, we have analyzed the prevalence of three student behaviors associated with disengagement in urban, suburban, and rural classrooms in the USA: off-task behavior, gaming the system, and carelessness. These students used the exact same learning software for high school Geometry across the same school year. However, the students used the software for different amounts of time in each school, a common phenomena in real-world use of educational software, where usage decisions are made by teachers and school administrators, rather than researchers and curriculum developers. To address this difference, we compared between these schools in two fashions. First, we compared all data to get a fully ecologically valid comparison. Second, we compared within a time-slice consisting of each student's 3rd to the 8th hours of usage in each school, in order to control for confounds stemming from differences in implementation between schools.

The two versions of the analysis agreed that the urban school had more off-task behavior and carelessness than the suburban school and rural school. In terms of these behaviors, students in the rural and suburban schools were more similar to each other than either school was to the urban school. One interesting note is that carelessness dropped significantly more over the course of the school year in the suburban and rural schools than in the urban school, suggesting that some influence or factor caused the suburban and rural students to become more diligent during the school year, but that this influence or factor was significantly less relevant in the urban school.

As both the rural school and the urban school had significant poverty, it appears that some aspect of these schools other than simply socio-economic status explains the higher frequency of off-task behavior and carelessness in the urban school. There are several potential hypotheses what other aspects may explain these behavioral differences, including differences in teacher expertise (which is often lower in urban schools [17]), differences in schools' facilities, equipment (e.g. computers), and physical environment, and differences in students' cultural backgrounds. Determining whether one of these factors explains the differences in off-task behavior and carelessness will be an important topic for future research.

A contradictory finding between analyses was found for gaming the system. Across the whole data set and entire school year, gaming the system was most frequent in the urban school. However, within the 3rd to 8th hours of tutor usage, gaming the system was least frequent in the urban school. This suggests that students in the urban school gamed more, later in the year. This may just be an artifact of the lessons encountered, as tutor lesson predicts a substantial portion of the variance in gaming behavior in Cognitive Tutors [1]. However, it may also be that the novelty of Cognitive Tutors reduces gaming initially in American students. There is evidence for this possibility in the finding that gaming was lower in all schools during the 3rd-8th hours. The difference in gaming behavior between

the early time-slice and the overall data set was more pronounced in the urban school, but this may be due to lower familiarity with educational technology in general, a finding worth investigating further.

In future years, we plan to replicate these analyses with a larger number of schools in each of these settings, using the research presented here as a methodological template for that later research. Automated machine-learned detectors provide an essential tool for analysis of this sort, in this author's opinion a better tool than existing alternatives. For example, it is not tractable to use observational, text replay annotation, or video methods at this sort of scale. [5] presents a use of text replay methods to analyze a single behavior among 58 students over an entire school year; though text replay methods are significantly faster than live observation or video coding methods, the coding needed for this analysis took over 200 hours. Utilizing text replays to annotate the 3 school sample used in this paper would have taken over 2000 hours, assuming a rate of observation equal to that in [5]. Video coding and field observation would have taken even longer.

That said, it is worth noting that automated detectors have important challenges not present when using human labels. It is important to validate the generalizability of detectors across students, schools, and learning materials, a task which has been only partially completed for the detectors used in this paper, and which has received insufficient attention in the literature in general. Construct validity is also a key issue in the use of machine-learned detectors, and is more a risk in detectors that are based on theoretically determined training labels (e.g. the model of carelessness), compared to detectors based on human judgments shown to have good inter-rater reliability (e.g. the detectors of off-task behavior and gaming the system). It is worth noting that automated detectors produced with a common alternative to machine learning, knowledge engineering, are likely to be prone to the same challenges to generalizability and construct validity as machine-learned detectors. Current practice with knowledge engineering often does not check detectors against human labels or across contexts, a potentially significant risk to using these models in discovery with models analyses.

As research applying detectors across contexts goes forward, it has significant potential to support progress in studying the impact of school context. By further study of which school contexts – and what attributes of those contexts – are associated with greater frequencies of disengaged behavior, we may be able to better understand the differences in learning between different learning settings. This may in turn support education researchers and practitioners in designing curricula, learning software, and interventions tailored to different schools – a potentially key step towards developing educational software that is equally effective for all students, whether they are in urban schools, rural schools, suburban schools, or elsewhere.

Acknowledgements

This research was supported by NSF grant REC-043779 to “IERI: Learning-Oriented Dialogs in Cognitive Tutors: Toward a Scalable Solution to Performance Orientation”, and by the Pittsburgh Science of Learning Center (National Science Foundation) via grant “Toward a Decade of PSLC Research”, award number SBE-0836012. We would

also like to thank Adriana de Carvalho and Ma. Mercedes T. Rodrigo for valuable comments and suggestions, and the members of the Pittsburgh Science of Learning Center DataShop, Alida Skogsholm in particular, for their support and assistance.

References

- [1] 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. *Complete On-Line Proceedings of the Workshop on Data Mining for User Modeling at the 11th International Conference on User Modeling 2007*, 76--80.
- [2] Baker, R.S.J.d. Modeling and Understanding Students' Off-Task Behavior in Intelligent Tutoring Systems. *Proceedings of ACM Computer-Human Interaction*, 2007, 1059--1068
- [3] Baker, R.S.J.d., Corbett, A.T., Aleven, V. More Accurate Student Modeling Through Contextual Estimation of Slip and Guess Probabilities in Bayesian Knowledge Tracing. *Proceedings of the 9th International Conference on Intelligent Tutoring Systems*, 2008, 406-415.
- [4] Baker, R.S.J.d., Corbett, A.T., Roll, I., Koedinger, K.R.. Developing a Generalizable Detector of When Students Game the System. *User Modeling and User-Adapted Interaction*, 18 (3), 2008, 287--314.
- [5] Baker, R.S.J.d., de Carvalho, A. M. J. A.: Labeling Student Behavior Faster and More Precisely with Text Replays. *Proceedings of the 1st International Conference on Educational Data Mining*, 2008, 38-47.
- [6] Beck, J. Engagement tracing: using response times to model student disengagement. *Proceedings of the 12th International Conference on Artificial Intelligence in Education (AIED 2005)*, 88-95.
- [7] Butcher, K., & Aleven, V. Integrating visual and verbal knowledge during classroom learning with computer tutors. In D.S. McNamara & J.G. Trafton (Eds.), *Proceedings of the 29th Annual Cognitive Science Society*, 2007, 137-142.
- [8] Carnegie Learning, Inc. *Results from The Colony, TX* (Cognitive Tutor Research Report TX-00-02), 2001. Pittsburgh, PA: Carnegie Learning, Inc.
- [9] Clements, M.A. Careless Errors Made by Sixth-Grade Children on Written Mathematical Tasks. *Journal for Research in Mathematics Education*, 13 (2), 1982, 136-144.
- [10] Cocea, M., Hershkovitz, A., Baker, R.S.J.d.: The Impact of Off-task and Gaming Behaviors on Learning: Immediate or Aggregate? *Proceedings of the 14th International Conference on Artificial Intelligence in Education*, 2009, 507--514

- [11] Fan, X., Chen, M. J.: Academic achievement of rural school students: A multi-year comparison with their peers in suburban and urban schools. *Journal of Research in Rural Education*, 15, 1999, 31--46.
- [12] Gobel, P.: Student off-task behavior and motivation in the CALL classroom. *International Journal of Pedagogies and Learning*, 4 (4), 2008, 4-18.
- [13] Karweit, N., Slavin, R.E.: Time-On-Task: Issues of Timing, Sampling, and Definition. *Journal of Experimental Psychology*, 74 (6), 1982, 844--851.
- [14] Koedinger, K. R., Anderson, J.R., Hadley, W., Mark, M. Intelligent tutoring goes to school in the big city. *Proceedings of the 7th International Conference on Artificial Intelligence and Education*, 1995, 421--428.
- [15] Koedinger, K. R., Corbett, A. T. Cognitive tutors: Technology bringing learning sciences to the classroom. R. K. Sawyer (Ed.), *The Cambridge handbook of the learning sciences*. New York, NY: Cambridge University Press, 2006.
- [16] Koedinger, K.R., Baker, R.S.J.d., Cunningham, K., Skogsholm, A., Leber, B., Stamper, J. A Data Repository for the EDM community: The PSLC DataShop. To appear in Romero, C., Ventura, S., Pechenizkiy, M., Baker, R.S.J.d. (Eds.) *Handbook of Educational Data Mining*, in press. Boca Raton, FL: CRC Press.
- [17] Lankford, H., S. Loeb, J. Wyckoff. Teacher Sorting and the Plight of Urban Schools: A Descriptive Analysis. *Educational Evaluation and Policy Analysis*, 24 (1), 2002, 37—62.
- [18] Lee, S.W., Kelly, K.E., Nyre, J.E. Preliminary Report on the Relation of Students' On-Task Behavior with Completion of School Work. *Psychological Reports*, 84, 1999, 267--272.
- [19] Lloyd, J.W., Loper, A.B.: Measurement and Evaluation of Task-Related Learning Behavior: Attention to Task and Metacognition. *School Psychology Review*, 15 (3), 1986, 336—345.
- [20] Mitrovic, A. An Intelligent SQL Tutor on the Web. *International Journal of Artificial Intelligence in Education*, 13 (2), 2003, 173--197.
- [21] Nicaud J.F., Bittar M., Chaachoua H., Inamdar P., Maffei L. Experiments With Aplusix In Four Countries. *International Journal for Technology in Mathematics Education*, 13 (1), 2006.
- [22] Rowe, J., McQuiggan, S., Robison, J., Lester, J.. Off-Task Behavior in Narrative-Centered Learning Environments. *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Education*, 2009, 99-106.
- [23] Sarkis, H. *Cognitive Tutor Algebra 1: Program Evaluation: Miami-Dade County Public Schools*, 2004. Lighthouse Point, FL: The Reliability Group.

On the Faithfulness of Simulated Student Performance Data

Michel C. Desmarais and Ildiko Pelczer
Polytechnique Montréal
michel.desmarais@polymtl.ca, ildiko.pelczer@gmail.com

Abstract. The validation of models for skills assessment is often conducted by using simulated students because their skills mastery can be predefined. Student performance data is generated according to the predefined skills and models are trained over this data. The accuracy of model skill predictions can thereafter be verified by comparing the predefined skills with the predicted ones. We investigate the faithfulness of different methods for generating simulated data by comparing the predictive performance of a Bayesian student model over real vs. simulated data for which the parameters are set to reflect those of the real data as closely as possible. A similar performance suggests that the simulated data is more faithful to the real data than for a dissimilar performance. The results of our simulations show that the latent trait model (IRT) is a relatively good candidate to simulate student performance data, and that simple methods that solely replicate mean and standard deviation distributions can fail drastically to reflect the characteristics of real data.

1 Introduction

Student cognitive diagnosis is commonly defined as estimating the probability of mastery of a set of skills by a given student. However, skills mastery cannot be directly measured. Instead, it is measured by observing performance results over a task, such as the successes or failures over a set of question items or exercises.

How can the accuracy of a cognitive diagnosis model be validated without direct measures of skill mastery? There are at least three, non exclusive means around this issue :

1. *Obtain indirect and independent measures of skill mastery.* Many studies rely on an independent source to estimate skill mastery and match the model prediction with this independent source. For example, Vomlel [9] asked experts to determine if a student mastered a set of skill in fraction algebra based on their answers to a test. The test data was used for training a Bayesian Network model and the prediction of the model was matched against the experts' judgment.

2. *Match predictions over observed items only.* Another approach consists in using solely the predicted outcome of observable items that can be directly matched to real data. No attempt is made at estimating skill mastery, and instead the approach relies on the assumption that hidden skills are correctly assessed if observed performance is accurately predicted.
3. *Generate simulated data.* The approach we investigate here consists in generating student performance data according to a predefined model for which skill mastery is defined for each student. This approach is commonly used in psychometric research where latent response models are validated against simulated data (see for eg. [4]). The approach has also been used for cognitive modeling within a number of studies and over different models such as the DINA [1] and a the Bayesian Network approach [7], to name but a few examples.

The obvious advantage of having predefined skills with simulated data is, however, plagued by the issue that the underlying skill model may not reflect the reality. The models can be over simplistic, or they can misrepresent the relationships between skills and performance, and among skills themselves.

We investigate this issue by using four models of skills to generate simulated student data. We look at how close are the performances of a student model trained over real and simulated data, while ensuring that the simulated data reflects as closely as possible the characteristics of the real data. The student model for this study is a Bayesian approach to cognitive modeling, POKS [5].

The first data generation model is one of the simplest possible and it serves as a baseline. The probability of *item outcome* (generally defined as a success or a failure to a test item question, or to an exercise) is a function of the expected values from marginal probabilities of item success rate and student scores. The second data generation model relies on a Q-Matrix that defines the links between items and skills. The matrix is used to assign skill outcome probabilities, from which a data sample can be generated. A third approach is based on a standard approach in Monte Carlo simulations in which sample data is generated by a technique that preserves the correlations among variables (among items in our case). The fourth approach is based on latent trait modeling (IRT—Item Response Theory) [2].

A number of studies on generating simulated student data have been conducted for the latent trait (IRT) approach [10][3][6], but they were all done within the IRT framework, using the same underlying latent trait models both for simulating the data and for measuring the predictive accuracy of the student model constructed from this data. On the contrary, the current study uses a Bayesian approach as the student model and a makes comparison of widely different approaches in addition to IRT.

We explain each of the simulated data generation approach in greater details below before moving to the experiments and the results.

2 Expected Outcome Based on Marginal Probabilities

The simplest model for generating simulated data is based on the expected item outcome according to marginal probabilities, as represented by the student general skill level and the item difficulty. This model presumes of no conceptual or skill structure behind the items set. Each item is considered independent of the other and the outcome solely depends on the item difficulty and the ability of the student.

Within this framework, the generation of sample test outcome can be conceptualized as a random sampling process using the expected probabilities. Assuming two vectors of probabilities: (1) S , that represents the skills mastery level of students, and (2), Q , that represents the (inverse) difficulty of items, then, the outer product of the two vectors is a matrix $\mathbf{X} = Q \times S$ where each element, m_{ij} , represents the expected probability of student i mastering item j . In the current study, we forced the sampling process to exactly replicate the distribution of scores, S , by sampling a predefined number of successes for each examinee.

Since the probability of an item x_{ij} being considered a success is solely dependent on the marginal probabilities, Q_i and S_j , we will refer to this model as the Marginal Probabilities sampling.

3 Q-Matrix Sampling

The second model we explore is based on a Q-matrix [8] which defines the links between items and skills. For example, assuming we have I items and K skills, and a response matrix of N students, then the Q-matrix and the response matrix are defined as:

$$\mathbf{Q} = \begin{bmatrix} q_{1,1} & \cdots & q_{1,K} \\ \vdots & \ddots & \vdots \\ q_{I,1} & \cdots & q_{I,K} \end{bmatrix}, \mathbf{X} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,I} \\ \vdots & \ddots & \vdots \\ x_{N,1} & \cdots & x_{N,I} \end{bmatrix}$$

For example, if an item x_1 involves only skills k_2 and k_3 , then $q_{1,2}$ and $q_{1,3}$ will be set to 1, and all other entries for that item, $q_{1,\bullet}$ will be set to 0.

The skill mastery of a set of students can be computed as the dot product of the two matrices: $\mathbf{X} \cdot \mathbf{Q}$.

The generation of sample data from this Q-matrix consists in defining the probability of an item outcome as a function of the level of mastery of the set of skills it involves. By defining skill mastery in the range $[0,1]$ (for which case the Q-matrix corresponds to a *capability matrix* as defined in [1]), then, the probability of a successful outcome to an item x_i is defined as the smallest of the mastery value of each skill involved for x_i . This is a heuristic estimate that reflects the requirement that all skills must be involved in order to correctly answer x_i .

Akin to the process described for *marginal probability sampling*, we can ensure that the scores distribution perfectly matches the real by fixing the number of item successes per

examinee. Sampling thus proceeds in a similar manner to the *marginal probabilities sampling* model, with the difference that instead of marginal probabilities, the item probabilities are derived from concept mastery. In turn, concept mastery is derived, in our experiment, from the student concept mastery distribution of the sample data and the capability matrix.

4 Covariance Matrix

Another mean of generating simulated student performance data is based on the idea of preserving the covariance (correlation) among items. This method is commonly used in Monte Carlo simulations. In the context of student test data, the method would stipulate that question items are interrelated and that a representative sample of simulated test data preserves the structure of correlation among items. This assumption is not unreasonable as we would, for example, expect that items of similar difficulty and that draw from the same skill set to show correlated student response patterns.

The generation of sample data based on item covariance relies on the Cholesky decomposition of the item covariance matrix. Assuming \mathbf{L} is the upper triangular matrix of the Cholesky decomposition of the item covariance matrix, a first step is to generate a sample of correlated variables as:

$$\mathbf{S} = \mathbf{NL}$$

where \mathbf{N} is an $N \times I$ matrix (number of students by number of items) of normally distributed independent random values having a mean of 0 and a standard deviation of 1. The sample data \mathbf{S} will be an $N \times I$ matrix for which the item covariance matrix will approach the real data item covariance. It will have an expected mean of 0. The second step is to fit the distribution of this data's item success rate to the real data by adding the vector of real data item means to each row of \mathbf{S} and, finally, to transform values to binary item outcome, setting values above 0.5 to 1 and 0 otherwise.

5 Latent Trait Models (IRT)

The last method of generating simulated student performance data relies on Item Response Theory, also known as *latent trait modeling*. As mentioned above, some authors have studied the faithfulness of this approach to replicate real data [10][3][6]. We refer the reader to [3] for a more elaborate description of this approach¹

We use a 2 parameter logistic IRT model for generating the simulated data. According to this model, the probability of a successful outcome by an examinee s to an item i is defined as:

$$P(X_i | \theta_s) = \frac{1}{1 + e^{-a_i(\theta_s - b_i)}}$$

¹Available from the ERIC Web Portal <http://eric.ed.gov/> under ref. ED414297 (accessed April 23, 2010).

where θ_s is the student's ability level, and where a_i and b_i are respectively the discrimination and difficulty levels of item i . The values for these three variables are directly estimated from the real data sample and therefore it is possible to replicate simulated data that reflects the real data. Estimates of the discrimination parameter is obtained with the R `ltm` package² and values for item difficulty and examinee ability are directly obtained through the logit transformation of the item average success rate and examinee percentage score. We also limit discrimination to values to the interval [0,4] and difficulty values to [-4,4], as is commonly done for IRT with small samples.

6 Experiments

We mentioned in the introduction that the issue with simulated student performance data is to determine how far the simulated data is representative of the complexity of the real student performance. To address this question, we train the POKS student model [5] over real and simulated data sets and compare its predictive performance across each condition. The simulated data sets are generated to closely resemble the real data according to the underlying model. The four models described above are used for simulated data: (1) **MP sampling**, marginal probability sampling (section 2), (2) **QM sampling**, Q-matrix sampling (section 3), (3) **Covariance**, sampling based on preserving item covariance using the Cholesky decomposition (section 4), and finally (4) **IRT**, sampling based on the latent trait modeling (section 5).

6.1 Adaptive Testing Simulation

The results of the different simulated data models are compared in the context of simulated adaptive testing with the POKS model. The process of adaptive testing consists in choosing the most informative item to present to the student and to infer the outcome of other items based on the pattern of previous item outcomes.

The performance is measured as the percent-correct predicted item outcome. Items that have been asked represent observed evidence and are considered correct by definition. Thus, performance after all items have been observed always converges to 100%. At the beginning, when no items are observed, item outcome is based on average item success rate: if an item has a success rate above 50%, it is considered mastered, and not mastered otherwise. As new items are observed, the POKS model computes the probability of mastery of each item based on the pattern of previous item outcome, and the predictions are compared to the actual data to compute the percent correct performance.

In this experiment, a cross-validation process is used for the College mathematics data set and a leave-one-out process is used for the Unix data set because of the small number of records.

²cran.r-project.org/web/packages/ltm/ltm.pdf

6.2 Data Sets

The characteristics of the real data sets from which the simulated data is generated can be very influential in this investigation and therefore we provide some details about them here. The experiment is conducted over two data sets:

1. *Unix*. The *Unix* data set contains 34 questions items that have all been answered by 48 respondents. The average success rate is 53% and it contains a large array of skills and difficulty, with test scores varying from 1/48 to 45/48, and item success rate varying from 1/34 to 34/34.

Skills decomposition of this data is done over 9 topics ("sys-admin", "awk", "basic" "directories", "file permissions", "input-output redirection", "printing", "regular expressions" "shell language"). These topics contain from 3 to 7 items and only one topic is associated with an item. In other words, the row sums of the Q-matrix is always 1.

2. *College Mathematics*. The *Math* data set is composed of 59 items, which were administered to 250 freshmen students at Polytechnique Montreal. Each item was analyzed by two domain experts who determined if it involved one of the following topics : (1) Algebra, (2) Geometry, (3) Trigonometry, (4) Matrices and Vectors, (5) Differential equations and (6) Integrals. Mean student score is 57%, ranging from 9/59 to 55/59.

Contrary to the *Unix* data set, most items are linked from two to four topics (only 17 are single topic, 32 are linked to two topics, 9 to three topics, and 1 to four topics).

The simulated performance data is generated to reflect as closely as possible the characteristics of the two real data sets. The similarity of the simulated data can be compared to the real one by looking at the correlation between success rates of students and items. Table 1 reports a number of similarity measures that represent the averages for 10 simulated data sets (numbers in parenthesis represent the standard deviations):

- *Mean* and *Sim. mean*: The percentage of correct responses over the whole data set. This number is to be compared to 53% for *Unix* and 57% for *Math*. The data generation process for the *QM sampling* and *MP sampling* methods were devised to match exactly this parameter.
- *Cor. exami.*: Pearson correlation between the simulated and real respondent test scores.
- *Cor. items*: Pearson correlation between the simulated and real average item scores.
- *Cor. concepts*: Pearson correlation between the simulated and real average concept mastery scores of students. Concept mastery for the students is computed on the basis of the dot product $\mathbf{X} \cdot \mathbf{Q}$ (see section 3), but with a normalization that ensures the scores range is between [0,1]. This normalization corresponds to the notion of a *capability matrix* (see [1]).

Table 1: Similarity of simulated data with real data

Unix	Mean	Sim. mean	Cor. items	Cor. exami.	Cor. concepts	% diff.
QM	.53	.53 (.00)	.93 (.01)	.80 (.03)	.81 (.02)	26 (1)
MP	.53	.53 (.01)	.55 (.08)	.64 (.10)	.28 (.05)	43 (2)
IRT	.53	.57 (.01)	.98 (.00)	.98 (.01)	.88 (.00)	15 (1)
Covariance	.53	.53 (.04)	.97 (.01)	.03 (.13)	.40 (.06)	38 (1)
Math	Mean	Sim. mean	Cor. items	Cor. exami.	Cor. concepts	% diff.
QM	.57	.57 (.00)	.84 (.01)	.03 (.05)	.57 (.02)	44 (0)
MP	.57	.57 (.00)	.55 (.04)	.78 (.04)	.20 (.03)	47 (0)
IRT	.57	.62 (.00)	.83 (.01)	.89 (.01)	.44 (.01)	40 (0)
Covariance	.57	.56 (.02)	.98 (.00)	.07 (.10)	.11 (.05)	42 (0)

- *% diff.*. Percentage of items with different outcome.

The patterns of similarity vary considerably across the different sampling methods, but the most consistent one is the IRT method, in particular for the *Unix* data set, with correlations of 0.98 for both item success rate and examinee scores. Whilst these correlations are very high, we find that 15% of items differ from the real to the generated samples. We will see from the data in table 2 that this 15% difference can considerably degrade the predictive performance if the items are chosen at random.

7 Results

The CAT simulations experiment results are reported in Figure 1. The graphs depict the predictive performance of POKS over the two data sets. The percent correct number of item outcome prediction (accuracy) is reported over the different experimental conditions. Both graphs start a 0% observations, where the accuracy corresponds to guesses based on item average success rate. They end at 100% of questions observed for each data set, where the accuracy converges to 1 because observed item outcome are considered correctly “predicted”. For indicative purpose, a straight line is drawn that starts at the initial guess of the real data, $(0, y_0)$, and ends at $(1,1)$. It corresponds to the theoretical baseline accuracy of random guesses over non-observed items and provides an idea of the prediction gain obtained with the student model (note that only the real data line is drawn). Standard errors over simulation runs are not shown on the graphs to avoid cluttering, but they are at most around 7% and have no significant affect on the general patterns observed. The different curves correspond to the four methods respectively described in section 2 to section 5 (see section 6 for label correspondance).

Table 2 provides a single score for the predictive performance, termed here the *accuracy gain*. This score represents the gain from guessing the outcome based on the initial probabilities of items and its range is $[0,1]$. It provides a simple means of comparing the

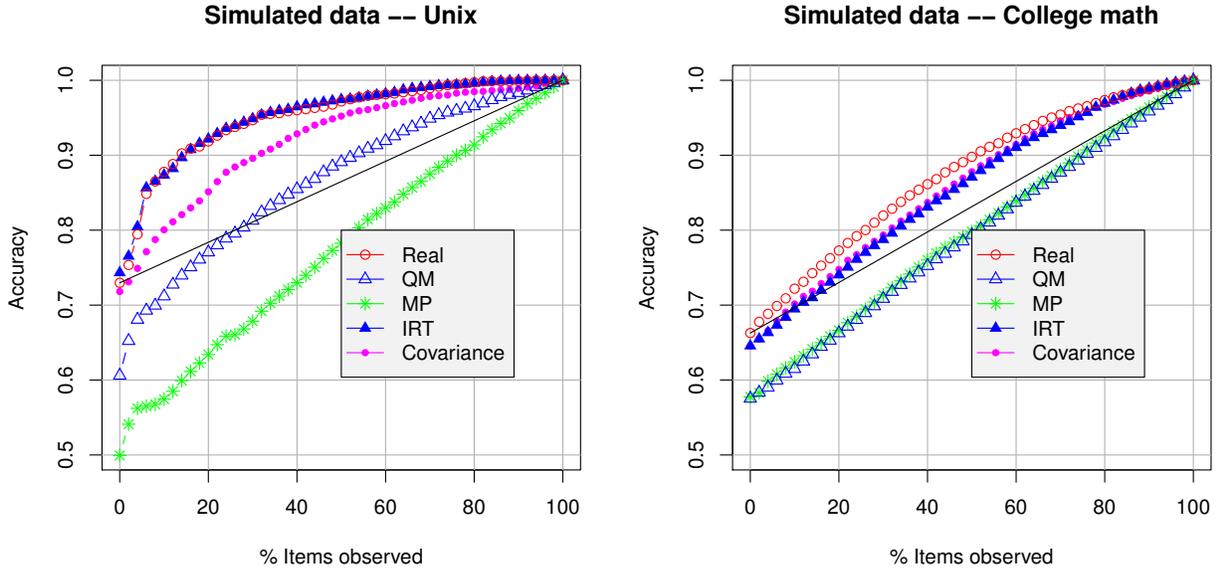


Figure 1: Results predictive accuracy simulation experiments with real student performance data compared with different models of simulated student data.

Table 2: Global accuracy gain over baseline

	Real	QM	MP	IRT	Covariance	15%
Unix	0.77 (0.0*)	0.43 (0.07)	0.14 (0.06)	0.80 (0.01)	0.58 (0.06)	0.29 (0.04)
Math	0.40 (0.02)	0.04 (0.01)	0.08 (0.01)	0.34 (0.01)	0.37 (0.01)	0.20 (0.03)

**Deterministic leave-one-out simulation*

overall predictive performances across the simulations and corresponds to the error reduction averaged over all intervals. It is computed as:

$$\text{accuracy gain} = \frac{1}{N} \sum_i^N \frac{y_i - \hat{y}_i}{1 - \hat{y}_i}$$

where N is the total number of intervals (we arbitrarily use 50), y_i is the accuracy at interval i (the x value) and \hat{y}_i is the baseline accuracy at that same interval as represented by the straight diagonal of the figures (there exists one diagonal per curve but only the one for the real curve is represented in the figures).

For indicative purposes and in addition to the four methods, table 2 also reports a score corresponding to randomly changing the values of item outcome for 15% of the items, which is the proportion of items differing from the IRT simulated data to the real data.

In the case of the Unix data, the results indicate that the *IRT* method is able to generate data over which the POKS model is similar the performance, with accuracy gains of 0.77

for real data agains 0.80 for *IRT*. The *Covariance* method comes second with a performance of 0.58 instead of 0.77.

In the case of the Math data, the general predictive performance of all methods is substantially lower than for the Unix data. The *Covariance* and *IRT* methods both yield performance relatively close to the real data, but this time the *Covariance* method is closer to the real data performance.

8 Discussion

This investigation is limited to two real world data sets and to predictions based on a single student model, namely POKS. As such, further investigations are necessary to draw stronger conclusions. Nevertheless, we can still hint at some conclusions. First, the simpler methods of generating data, based on marginal probabilities and on concept mastery, yield simulated data that do not appropriately reflect the underlying structure of the real student performance data. However, the *IRT* method, based on the 2 parameter model (difficulty and discrimination), does appear to reflect the characteristics of real data, but not systematically for all data sets, as a non neglectible difference can be observed in the case of the Math data set. Furthermore, the *Covariance* method actually generates data for which the predictive accuracy is slightly closer to real data then the *IRT* method is. It also is close overall to the real data, standing at 0.37 accuracy gain compared to 0.40 for real data.

This investigation focused on models for generating data which allow their parameters to replicate real data characteristics, namely items difficulty, student skill levels, concept mastery as defined by the Q-matrix, and item covariance. Not all models allow this replication as readily as for these approaches. The DINA model used in [1] contains parameters that cannot be readily estimated from data, such as performance slips. Validating the faithfulness of such models is a desirable endeavour that would require means to estimate such parameters and constitutes an interesting research avenue. Indirectly, such investigations are in fact a means to validate if a model can actually reflect the characteristics of real data and, thus, they can be considered as an assessment of the external validity of a student model.

Turning back to the fundamental question of whether we can rely on simulated data to validate a student model, the simulations in this study suggest that simulated data from the 2 parameter *IRT* model can appropriately reflect some data set characteristics, but not with equal faithfulness for all data sets. It suggests that the validation of a model based on the indirect and independent measures of skill mastery may be indispensable to ensure a proper validation, as we outlined in the introduction. Alternatively, we could argue that the approach which consists in validating predictive performance over observable items only is just as indispensable. If we assume that the accuracy of a model for predicting item outcome is directly and monolithically linked to the accuracy of non observable parameters estimates of a model, then item outcome represents a good indirect measure of skills and concept mastery.

References

- [1] Ayers, E., Nugent, R., and Dean, N. A comparison of student skill knowledge estimates. In *2nd International Conference on Educational Data mining, Cordoba, Spain* (2009), pp. 1–10.
- [2] Baker, F. B., and Kim, S.-H. *Item Response Theory, Parameter Estimation Techniques*. Marcel Dekker Inc., New York, NY, 2004.
- [3] Davey, T., Nering, M. L., and Thompson, T. Realistic simulation of item response data. Tech. rep., ACT Research Report Series 97-4, July 1997.
- [4] de la Torre, J., and Douglas, J. Higher-order latent trait models for cognitive diagnosis. *Psychometrika* 69 (September 2004), 333–353. 10.1007/BF02295640.
- [5] Desmarais, M. C., Maluf, A., and Liu, J. User-expertise modeling with empirically derived probabilistic implication networks. *User Modeling and User-Adapted Interaction* 5, 3-4 (1996), 283–315.
- [6] Harwell, M. R., Stone, C. A., Hsu, T.-C., and Kirisci, L. Monte carlo studies in item response theory. *Applied Psychological Measurement* 20, 2 (1996), 101–125.
- [7] Millán, E., and Pérez-de-la-Cruz, J. L. A Bayesian diagnostic algorithm for student modeling and its evaluation. *User Modeling and User-Adapted Interaction* 12, 2–3 (2002), 281–330,.
- [8] Tatsuoka, K. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement* 20 (1983), 345–354.
- [9] Vomlel, J. Bayesian networks in educational testing. *International Journal of Uncertainty, Fuzziness and Knowledge Based Systems* 12, Supplementary Issue 1 (2004), 83–100.
- [10] Yiand, Q., and Nering, M. L. Simulating nonmodel-fitting responses in a CAT environment. Tech. rep., ACT Research Report Series, December 1998.

Mining Bodily Patterns of Affective Experience during Learning

Sidney D'Mello and Art Graesser
[sdmello, a-graesser]@memphis.edu
Institute for Intelligent Systems, University of Memphis

Abstract. We investigated 28 learners' postural patterns associated with naturally occurring episodes of boredom, flow/engagement, confusion, frustration, and delight during a tutoring session with AutoTutor, a dialogue-based intelligent tutoring system. Training and validation data were collected in a learning session with AutoTutor, after which the learners' affective states (i.e., emotions) were rated by the learner, a peer, and two trained judges. An automated body pressure measurement system was used to capture the pressure exerted by the learner on the seat and back of a chair during the tutoring session. We extracted 16 posture-related features that focused on the pressure exerted along with the magnitude and direction of changes in pressure during emotional experiences. Binary logistic regression models yielded medium sized effects in discriminating the affective states from neutral. An analysis of the parameters of the models indicated that the affective states were manifested by three unique postural configurations and a general increase in movement (when compared to neutral).

1 Introduction

Intelligent Tutoring Systems (ITSs) have emerged as valuable tools to promote active learning by capitalizing on the benefits of one-on-one tutoring in an automated fashion. Although ITSs have typically focused on the learner's cognitive states, they can be far more than mere cognitive machines. ITSs can be endowed with the ability to recognize and respond to learners' affective states. Consequently, the last decade has witnessed a burst of research activities aimed at developing ITSs that are responsive to learners' affective states in addition to their cognitive states [1-6]. Much of the research has focused on developing systems to detect learner affect automatically because an ITS cannot respond to learners' affective states if it cannot detect their affective states.

State-of-the-art affect detection systems have overlooked posture as a serious contender when compared to facial expressions and acoustic-prosodic features (see reviews by [7, 8]), so an analysis of posture merits a closer examination. There apparently are some benefits to using posture as a means to diagnose the affective states of a learner. One compelling reason is that human bodies are large and have multiple degrees of freedom, thereby providing them with the capability of assuming a myriad of unique configurations [9]. These static positions can be concurrently combined and temporarily aligned with a multitude of movements, all of which makes posture a potentially ideal affective communicative channel [10, 11]. Perhaps the greatest advantage to posture-based affect detection is that gross body motions are ordinarily unconscious, unintentional, and thereby not susceptible to social editing, at least compared with facial expressions, speech intonation, and some gestures [12].

There has been some recent research aimed at developing posture-based affect detection systems [13-15]. The results of these studies have indicated that posture is an important

channel to detect the affective states that occur during learning sessions. Although these studies have provided an indication of *how accurate* posture-based affect detectors are, left unanswered is the important question of *how does* the body convey affect through articulations of posture and modulations of movement. For example, we have previously reported that supervised classifiers can discriminate confusion, frustration, boredom, engagement/flow, and delight from neutral with an average accuracy of 74% (baseline = 50%) [15], but we are unaware of the manner in which these states are expressed through the body (e.g. forward leans, arms akimbo, general fidgeting).

Hence, taking a step back from affect detection, the present paper focuses on applying data mining techniques to uncover relationships between body position, arousal, and affect. Learner affect and posture data were obtained from an earlier study [16] where 28 students were tutored on computer literacy topics with AutoTutor, an ITS with conversational dialogues [17]. Learner affect was measured via a retrospective affect judgment protocol in which the learners' affective states were rated by the learner, a peer, and two trained judges (described in section 3). An automated body pressure measurement system was used to capture the pressure exerted by the learner on the seat and back of a chair during the tutoring session. Bodily patterns of affective experience were mined with binary logistic regressions. We begin with a description of the theoretical framework that links body posture and affect.

2 Theoretical Framework Linking Posture and Affect

Several researchers have proposed that affective experience can be productively analyzed by initially considering the underlying dimensions of valence (pleasant/unpleasant) and arousal (active/sleepy) [18, 19]. After this initial classification of emotions is achieved, the emotion is analyzed further with respect to the task, environment, and social world (i.e., an *appraisal* is performed). We adopt the first-stage valence-arousal framework in uncovering the expressive nature of body language in communicating affect.

One challenge to adopting the valence-arousal framework for learning environments is that it does not neatly fit as well as the analyses of the basic emotions (anger, fear, sadness, enjoyment, disgust, and surprise) which naturally align on a valence scale [20]. A simple valence-based categorization is more challenging for the affective states that are prominent during learning. For example, while prolonged, hopeless confusion can be considered to be a negative emotion, it is not necessarily the case that brief experiences of confusion share the same negative connotation. In fact, confusion is a state that has been positively correlated with deep thinking and learning [21, 22], and some learners, such as academic risk takers, like to be challenged with tasks that create short-term confusion, followed by a resolution [23]. Although it is difficult to fathom that learners derive pleasure from being confused, there are theories of learning that suggest complex relations between affect and cognition that extend beyond a simple valence dimension.

Therefore, we relied on an *attentive-arousal* framework [24] to interpret relationships between posture and the affective states of the learner. One can think of heightened pressure in the seat as resonating with a tendency to position one's body towards the source of stimulation (i.e., high attentiveness since the learner is positioning his or her

body towards the learning interface, or a short distance between the nose and the screen). On the other hand, an increase in pressure on the back of the chair suggests that the learner is leaning back and detaching himself or herself from the stimulus (low attentiveness). Arousal was operationally defined by the rate of change of pressure exerted on the back and the seat of the pressure sensitive chair (described below) and is similar to the degree of movement [18].

3 Data Collection

The participants were 28 undergraduate students from a mid-south university who participated for extra course credit. The study was divided into two phases. The first phase was a standard pretest–intervention–posttest design. The participants completed a pretest with multiple-choice questions, then interacted with the AutoTutor system for 32 minutes on one of three randomly assigned topics in computer literacy (Hardware, Internet, Operating Systems), and then completed a posttest. During the tutoring session, the system recorded a video of the participant’s face, their posture patterns with the Body Pressure Measurement System (BPMS), and a video of their computer screen.

The BPMS system, developed by Tekscan™ consisted of a thin-film pressure pad (or mat) that can be mounted on a variety of surfaces. The pad was paper thin with a rectangular grid of sensing elements. Each sensing element provided a pressure output in mmHg. Our setup had one sensing pad placed on the seat of a Steelcase™ Leap Chair and another placed on the back of the chair (see Figure 1 A).

The output of the BPMS system consisted of 38×41 matrix (rows \times columns) with each cell in the matrix monitoring the amount of pressure as reported by the corresponding sensing element (see Figure 1B). Therefore, in accordance with our setup, at each sampling instance (1/4 second), matrices corresponding to the pressure in the back and the seat of the chair were recorded for offline analyses.

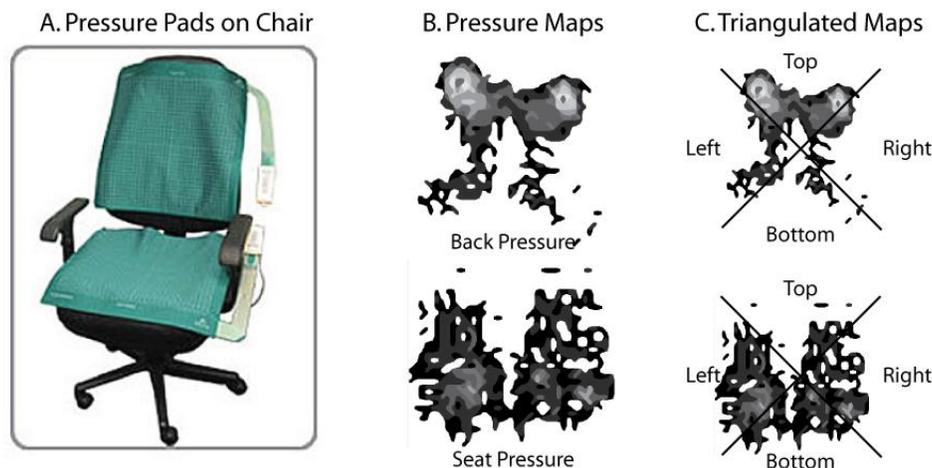


Figure 1: Body Pressure Measurement System

The second phase of the study involved affect judgments by the learner, a peer, and two trained judges. The affect judging session proceeded by displaying video streams of both

the learner's screen and face, which were captured during the AutoTutor session. Judges were instructed to make judgments on what affective states were present at any moment during the tutoring session by manually pausing the videos (spontaneous judgments). They were also instructed to make judgments at each 20-second interval where the video automatically stopped (fixed judgments); these fixed points are particularly useful to compute interrater reliability among judges (see [16]). A list of the affective states and definitions was provided for all judges. The states were boredom, confusion, flow, frustration, delight, neutral and surprise. The selection of emotions was based on previous studies of AutoTutor and other learning environments [25].

Judgments were provided by the learners themselves (self judgments), by untrained peers, and by two trained judges. These trained judges had been trained on how to detect facial action units according to Ekman's Facial Action Coding System [20] and on characteristics of AutoTutor's dialogue (i.e., contextual cues).

4 Results and Discussion

4.1 Computing Posture Features

Several features were computed by analyzing the pressure maps of the 28 participants recorded in the study. We computed six pressure-related features and two features related to the pressure coverage for both the back and the seat, yielding 16 features in all. Each of the features was computed by examining the pressure map (called the *current frame*) during an emotional episode (i.e., when an emotion judgment was made). The pressure related features include the *average pressure*, which measures the average pressure exerted on the chair and the *top pressure* which was the pressure on the topmost segment of the pad (see Figure 1C). The *prior change* and *post change* measure the difference between the average pressure in the current frame and the frame three seconds earlier and later respectively. The *reference change* measures the difference between the average pressure in the current frame and the frame for the last known affective rating. Finally, the *average pressure change* measures the mean change in the average pressure across a predefined window, typically 4 seconds, that spans two seconds before and two seconds after an emotion judgment. The two coverage features examined the proportion of non-negative sensing units (*average coverage*) on each pad along with the mean change of this feature across a 4-second window (*average coverage change*). Please see [15] for detailed description of the features.

Each feature vector was associated with an emotion category on the basis of each of the four human judges' affect ratings. More specifically, each emotion judgment was temporally bound to each posture based feature vector. This data collection procedure yielded four ground truth models of the learner's affect (self, peer, two trained judges), so we were able to construct four labeled data sets. When aggregated across each 32-minute session for each of the 28 participants, we obtained 2967, 3012, 3816, and 3723 labeled data points for the self, peer, trained judge 1, and trained judge 2, respectively.

4.2 Affect-Neutral Discrimination from Posture Features

We conducted a series of binary logistic regression analyses in order to systematically explore relationships between the posture features and the various affective states. The analyses served two purposes. First, we were interested in determining the extent to which the five affective states of interest could be predicted from the various posture features. Second, the statistically significant predictors (positive or negative) of the logistic regression models were used to isolate commonalities in posture configurations that accompany affective experience through the body.

We conducted five binary logistic regression analyses for each of the four data sets (self, peer, 2 trained judges). Each logistic regression analysis was conducted to segregate each of the five affective states from neutral. Therefore, the criterion (dependent) variable for each logistic regression analysis was the affective state (1 or 0 if present or absent, respectively) whereas the predictor variables were the set of posture features.

Statistically significant relationships ($p < .05$) were discovered for all of the models. When R^2 values were averaged across judges and affective states, the posture features explained 11% of the variance in discriminating each affective state from neutral. For the affective states of boredom, delight, flow, and frustration, the posture features explained 13%, 12%, 14%, and 11% of the variance respectively. The weakest model was obtained for confusion with the posture features explaining only 6% of the variance.

However, when one considers the best model across judges (e.g., self for boredom and confusion, peer for flow and frustration, and judge 2 for delight), 19%, 10%, 16%, 19%, and 14% of the variance was explained for boredom, confusion, delight, flow, and frustration, respectively. More succinctly, 16% of the variance was explained on average. This result is close to a medium sized effect [26], and indicates that posture is indeed a viable channel for affect detection.

Table 1. Summary of binary logistic regression analyses with posture features as predictors of each affective state from neutral for data sets based on affective judgments of the four judges

Affective State	Affect Judge							
	Self		Peer		Judge 1		Judge 2	
	χ^2	R^{2b}	χ^2	R^{2b}	χ^2	R^{2b}	χ^2	R^{2b}
Boredom	196.87	.19	139.43	.12	91.44	.09	148.61	.10
Confusion	105.71	.10	87.05	.08	53.35	.03	44.01	.03
Delight	42.24	.09	29.82	.09	94.13	.12	10.22	.16
Flow	142.96	.13	235.40	.19	118.92	.09	142.90	.13
Frustration	101.94	.12	102.61	.14	73.33	.08	71.63	.11

^bComputed in accordance with the Nagelkerke R^2 which is a pseudo R^2 [27].

Next, we consulted the numerical directions (i.e. signs, + and -) of the statistically significant coefficients of the logistic regression models in order to explore relationships between body position, movement, and affect. Although all 16 features were used as predictors for the logistic regression analyses, we focus on the average pressure and the change in pressure; these are the most interpretable features and can be theoretically aligned within the attentive-arousal framework.

The logistic regression analyses were used to discriminate between each affective state versus neutral, hence, a statistically significant predictor implies that the feature is heightened (significant positive predictor) or suppressed (significant negative predictor) during the emotional experience when compared to neutral. For example the *back average pressure change* feature was a significant positive predictor for the boredom-neutral logistic regression, so the episodes of boredom were accompanied by an increase in movement on the back when compared to neutral.

Table 2. Significant predictors for the multiple regression models for emotions in each data set

Sensor Feature	Boredom				Confusion				Delight				Flow				Frustration					
	SF	PR	J1	J2	SF	PR	J1	J2	SF	PR	J1	J2	SF	PR	J1	J2	SF	PR	J1	J2		
Back	Pressure			+	+			-	-			-	-			-	-			-		
	Change			-								+	+			+	+			-		
Seat	Pressure			-	-	-					+	+	+	+		+	+			-	+	+
	Change			+	+	+			+	+			+			-				+	+	

Notes. SF: Self Judgments, PR: Peer Judgments, J1: Trained Judge1, J2: Trained Judge2
+ or - indicates that the feature is a positive or negative predictor in the logistic regression model at $p < .05$ significance level. Empty cells indicate that a feature was not a statistically significant predictor for the respective emotion.

A number of relationships surface when one considers the significant predictors of the affective states in which at least two judges agreed. The requirement that two judges agree on the significance and direction of each predictor is motivated by a desire to establish a degree of convergent validity in exploring the posture-affect relationships. By requiring that the features need to be significant predictors of affect for at least half of the judges models ensures that, to some extent, they generalize across judges.

Boredom. Our results suggested that during episodes of boredom, the learners leaned back and presumably disengaged from the learning environment (low attentiveness indicated by an increase in pressure on back and a significant decrease in pressure on the seat). Experiences of boredom were also accompanied by an increase in the rate of change of pressure exerted on the seat. Therefore, heightened arousal was associated with the boredom experience, presumably as learners mentally disengage from the tutor and divert their cognitive resources to fidget around and alleviate their ennui.

Some may view the heightened arousal accompanying boredom to conflict with the preconceived notion of boredom in which a learner stretches out, lays back, and simply disengages. Our results suggest that the learner lays back, disengages, but is aroused. Furthermore, this pattern of increased arousal accompanying disengagement (or boredom) replicates a previous study by Mota and Picard [14]. They monitored activity related posture features and discovered that children fidget when they were bored while performing a learning task on a computer.

Delight and Flow. In contrast to boredom, learners experiencing the positive emotions of delight and flow demonstrate increased attentiveness towards the learning environment by leaning forward. Learners experiencing these emotions also demonstrate heightened arousal on the back of the chair – at least when compared to the neutral state.

Confusion and Frustration. Similar to delight and flow, learners experiencing confusion and frustration tend to lean forward. However, it appears that during episodes of delight and flow learners lean forward at a steeper inclination than with confusion and frustration. We arrived at this conclusion because the increase in the pressure exerted on the seat of the chair was accompanied by a commensurate decrease in pressure exerted on the back of the chair for delight and flow. On the other hand, confusion was accompanied by a decrease in pressure exerted on the back of the chair without any accompanying increase on the seat. Similarly for frustration the increase in pressure on the seat was devoid of a notable (statistically significant) decrease on the back. We suspect that the pattern of body position with confusion and frustration indicates that learners are in an upright position when they experience these states, as opposed to the forward lean that seems to accompany experiences of delight and flow.

Arousal. In addition to boredom, it also appears that experiences of delight, flow, confusion, and frustration are accompanied by significant arousal, either on the back or the seat of the pressure sensitive chair. The arousal that is affiliated with confusion and frustration occurs on the back while there is an increase in movement on the seat when the emotions of delight and flow are experienced. While the significance of the location of the arousal (back or seat) during experiences of these is unclear, what is important is that all affective experiences (including boredom) were accompanied by heightened arousal when compared to neutral. In summary, the experience of each affective state is accompanied by a significant increase in arousal, at least when compared to the neutral baseline.

General Discussion

We discovered relationships between body position, degree of movement, and learners' affective states. Our findings are in line with an attentive-arousal or engagement-arousal framework (see section 2). With respect to the attentiveness dimension, it appears that there are three bodily configurations that are associated with the affective states. These include heightened *attentiveness*, which is manifested by a forward lean when the positive emotions of delight and flow are experienced. On the other hand, bored learners tend to lean back, presumably in a state of *disengagement* (boredom).

States such as confusion and frustration occur when learners confront contradictions, anomalous events, obstacles to goals, salient contrasts, perturbations, surprises, equivalent alternatives, and other stimuli or experiences that fail to match expectations [28]. Learners are in a state of cognitive disequilibrium, with more heightened physiological arousal, and more intense thought. Our results suggest that the bodily corollary to the mental state of cognitive disequilibrium is an *alert* position, where the learner sits upright and pays attention.

The single major finding with respect to the arousal dimension is that each of the affective experiences is accompanied by a significantly higher arousal when compared to a neutral baseline (i.e. no emotion). This finding has important theoretical implications because some of our colleagues view some of these emotions (i.e., flow, confusion, etc.) as cognitive states, whereas other researchers would classify them as either emotions or affect states. We have traditionally agreed with the latter group because we hypothesize that the single major discriminator of an affective state over a cognitive state is that the affective state is accompanied by enhanced physiological arousal (compared with neutral). Our results indicate that in most cases there is a significant increase in bodily movements (bodily arousal) during the experience of emotional episodes indicating that both cognitive and affective processes are at play. Therefore, it might be the case that the term cognitive-affective state is the most defensible position for mental states such as confusion, flow, and frustration.

We acknowledge that the aforementioned relationships between body posture and affect ignore individual differences in affect expression. In ideal circumstances, from a statistical point of view, the landscape of postural configurations would be evenly distributed among the 28 different students. However, this claim is implausible and it is therefore important to contrast contributions of individual learners versus generalizable posture features in predicting affect. Our results focused on broad patterns observed across all learners and should be interpreted with a modicum of caution. We are currently addressing these concerns by building models that attempt to separate variance explained by individual student characteristics versus variance explained by posture features above and beyond individual differences.

References

- [1] A. Kapoor and R. Picard, "Multimodal affect recognition in learning environments," in *Proceedings of the 13th annual ACM international conference on Multimedia*, Hilton, Singapore, 2005.
- [2] C. Conati and H. Maclaren, "Empirically building and evaluating a probabilistic model of user affect," *User Modeling and User-Adapted Interaction*, vol. 19, pp. 267-303, 2009.
- [3] K. Forbes-Riley, M. Rotaru, and D. Litman, "The relative impact of student affect on performance models in a spoken dialogue tutoring system," *User Modeling and User-Adapted Interaction*, vol. 18, pp. 11-43, Feb 2008.

- [4] S. McQuiggan, B. Mott, and J. Lester, "Modeling self-efficacy in intelligent tutoring systems: An inductive approach," *User Modeling and User-Adapted Interaction*, vol. 18, pp. 81-123, Feb 2008.
- [5] S. D'Mello, R. Picard, and A. Graesser, "Towards an affect-sensitive AutoTutor," *Intelligent Systems, IEEE*, vol. 22, pp. 53-61, 2007.
- [6] I. Arroyo, B. Woolf, D. Cooper, W. Burleson, K. Muldner, and R. Christopherson, "Emotion Sensors Go To School," in *Proceedings of 14th International Conference on Artificial Intelligence In Education*, V. Dimitrova, R. Mizoguchi, B. Du Boulay, and A. Graesser, Eds. Amsterdam: IOS Press, 2009, pp. 17-24.
- [7] Z. Zeng, M. Pantic, G. Roisman, and T. Huang, "A survey of affect recognition methods: Audio, visual, and spontaneous expressions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, pp. 39-58, Jan 2009.
- [8] M. Pantic and L. Rothkrantz, "Toward an affect-sensitive multimodal human-computer interaction," *Proceedings of the IEEE*, vol. 91, pp. 1370-1390, Sep 2003.
- [9] N. Bernstein, *The co-ordination and regulation of movement*. London: Pergamon Press, 1967.
- [10] M. Coulson, "Attributing emotion to static body postures: Recognition accuracy, confusions, and viewpoint dependence," *Journal of Nonverbal Behavior*, vol. 28, pp. 117-139, Sum 2004.
- [11] J. Montepare, E. Koff, D. Zaitchik, and M. Albert, "The use of body movements and gestures as cues to emotions in younger and older adults," *Journal of Nonverbal Behavior*, vol. 23, pp. 133-152, Sum 1999.
- [12] P. Ekman and W. Friesen, "Nonverbal Leakage and Clues to Deception," *Psychiatry*, vol. 32, pp. 88-&, 1969.
- [13] G. Castellano, M. Mortillaro, A. Camurri, G. Volpe, and K. Scherer, "Automated Analysis of Body Movement in Emotionally Expressive Piano Performances," *Music Perception*, vol. 26, pp. 103-119, Dec 2008.
- [14] S. Mota and R. Picard, "Automated Posture Analysis for Detecting Learner's Interest Level," in *Computer Vision and Pattern Recognition Workshop, 2003. CVPRW '03. Conference on*. vol. 5, 2003, pp. 49-49.
- [15] S. D'Mello and A. Graesser, "Automatic detection of learners' affect from gross body language," vol. 23, pp. 123 - 150, 2009.
- [16] A. Graesser, B. McDaniel, P. Chipman, A. Witherspoon, S. D'Mello, and B. Gholson, "Detection of emotions during learning with AutoTutor," in *28th Annual*

- Conference of the Cognitive Science Society, Vancouver, Canada, 2006, pp. 285-290.*
- [17] A. Graesser, S. L. Lu, G. Jackson, H. Mitchell, M. Ventura, A. Olney, and M. Louwerse, "AutoTutor: A tutor with dialogue in natural language," *Behavioral Research Methods, Instruments, and Computers*, vol. 36, pp. 180-193, 2004.
- [18] J. Russell, "Core affect and the psychological construction of emotion," *Psychological Review*, vol. 110, pp. 145-172, 2003.
- [19] L. Barrett, B. Mesquita, K. Ochsner, and J. Gross, "The experience of emotion," *Annual Review of Psychology*, vol. 58, pp. 373-403, 2007.
- [20] P. Ekman and W. Friesen, *The Facial Action Coding System: A technique for the measurement of facial movement*. Palo Alto: Consulting Psychologists Press, 1978.
- [21] S. Craig, A. Graesser, J. Sullins, and J. Gholson, "Affect and learning: An exploratory look into the role of affect in learning," *Journal of Educational Media*, vol. 29, pp. 241-250, 2004.
- [22] A. Graesser, P. Chipman, B. King, B. McDaniel, and S. D'Mello, "Emotions and learning with AutoTutor," in *13th International Conference on Artificial Intelligence in Education* R. Luckin, K. Koedinger, and J. Greer, Eds. Amsterdam: IOS Press, 2007, pp. 569-571.
- [23] D. Meyer and J. Turner, "Re-conceptualizing emotion and motivation to learn in classroom contexts," *Educational Psychology Review*, vol. 18, pp. 377-390, Dec 2006.
- [24] P. Bull, *Posture and Gesture*. Oxford Pergamon Press, 1987.
- [25] R. Baker, S. D'Mello, M. Rodrigo, and A. Graesser, "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*, vol. 68 (4), pp. 223-241, 2010.
- [26] J. Cohen, "A power primer," *Psychological Bulletin*, vol. 112, pp. 155-159, Jul 1992.
- [27] N. J. D. Nagelkerke, "A Note On A General Definition Of The Coefficient Of Determination," *Biometrika*, vol. 78, pp. 691-692, 1991.
- [28] A. Graesser, S. Lu, B. Olde, E. Cooper-Pye, and S. Whitten, "Question asking and eye tracking during cognitive disequilibrium: Comprehending illustrated texts on devices when the devices break down," *Memory and Cognition*, vol. 33, pp. 1235-1247, 2005.

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)?

Mingyu Feng¹, Neil Heffernan²
{mingyu.feng@sri.com, nth@wpi.edu}

¹ SRI International, Menlo Park, CA 94025

² Worcester Polytechnic Institute, Worcester, MA 01609

Abstract. Dynamic assessment (DA) has been advocated as an interactive approach to conduct assessments to students in the learning systems as it can differentiate student proficiency at a finer grained level. Sternberg and others have been pursuing an alternative to IQ tests. They proposed to give students tests to see how much assistance it takes a student to learn a topic; and to use as a measure of their learning gain. They referred to this as dynamic assessment. It was suggested that this assisting-while-testing procedure could be done well by computer. To researchers in the intelligent tutoring system community, it comes as no surprise that measuring how much assistance a student needs to complete a task successfully is probably a good indicator of this lack of knowledge. However, a cautionary note is that conducting DA takes more time than simply administering regular test items to students. In this paper, we report a study analyzing 40-minutes data of 1,392 students from two school years using educational data mining techniques. We compare two conditions: one contains only practice items without intervention while the other condition allows students to seek for help when they encounter difficulties. The result suggests that for the purpose of assessing student performance, it is more efficient for students to take DA than just having practice items.

Keywords: Dynamic assessment, assessment in learning system.

1 Introduction

In the past twenty years, much attention from the Intelligent Tutoring System (ITS) community has been paid to improve the quality of student learning while the topic of improving the quality of assessment has not been emphasized as much. However, student assessment is very important. In the US, state tests mandated by “No Child Left Behind” are causing many schools to give extra tests to see if they can group students together to get special help. Of course, giving tests for this practices is not meant to help students learn, but is mainly focused on being able to tell teachers and principals about who needs help on what. It would be great if intelligent tutoring systems could be used to do the tests, so that no time from instruction is “stolen” to do extra assessments. Many psychometricians would argue that let students learn while being tested will make the assessment harder since you are trying to measure a moving target. Can an ITS, if given the same amount of time, be a better assessor of students (while also of course providing the benefit of helping students learn during that time period. Is it possible to have our cake (better assessment) and eat it too (also let student learn)?

As an intelligent tutoring system adapts the educational interaction to the specific needs of the individual student, student modeling is an essential component in an ITS as well. The learning effectiveness depends heavily on the understanding of student knowledge, difficulties, and misconceptions. Yet, assessing students automatically,

continuously and accurately without interfering with student learning is an appealing but also a challenging task.

Dynamic assessment (DA, or sometimes called dynamic testing, Grigorenko & Sternberg, 1998) has been advocated as an interactive approach to conducting assessments to students in the learning systems as it can differentiate student proficiency at the finer grained level. Different from traditional assessment, DA uses the amount and nature of the assistance that students receive which is normally not available in traditional practice test situations as a way to judge the extent of student knowledge limitations. Even before the computer supported systems become popular, much work has been done on developing “testing metrics” for dynamic testing (Grigorenko & Sternberg, 1998; Sternberg & Grigorenko, 2001, 2002) to supplement accuracy data (wrong/right scores) from a single setting. Researchers have been interested in trying to get more assessment value by comparing traditional assessment (static testing; students getting an item marked wrong or even getting partial credit) with a measure that shows how much help they needed. Grigorenko and Sternberg (1998) reviewed relevant literature on this topic and expressed enthusiasm for the idea. Sternberg & Grigorenko (2001, 2002) argued that dynamic tests not only serve to enhance students’ learning of cognitive skills, but also provide more accurate measures of ability to learn than traditional static tests. Campione and colleagues (Bryant, Brown & Campione, 1983; Campione & Brown, 1985) took a graduated prompting procedure to compare traditional testing paradigms against a dynamic testing paradigm. In the dynamic testing paradigm, learners are offered increasingly more explicit prewritten hints in response to incorrect responses. In this study they wanted to predict learning gains between pretest and posttest. They found that student learning gains were not as well correlated ($R = 0.45$) with static ability score as with their “dynamic testing” ($R = 0.60$) score. They also suggested that this dynamic method could be effectively done by computer, but never pushed toward to conduct such studies using a computer system.

Intelligent Tutoring Systems are perfect test beds for DA as they naturally lead students into a tutoring process to help students with the difficulties they have encountered. Traditional paper and pencil or even some online assessment usually focuses on students’ responses to test items and whether they are answered correctly or incorrectly. It ignores all other student behaviors during the test (e.g., response time). However, the most unique information from DA is information about the learner’s responsiveness to intervention (Fuches et al. 2007) in the tutoring system. There have been a few studies that pay attention to such unique information. For instance, recently Fuches and colleagues (Fuches et al., 2008) employed DA in predicting third graders' development of mathematical problem solving. We (Feng, Heffernan & Koedinger, 2006, 2009) have also taken advantage of a computer-based tutoring system (ASSISTments, www.assistment.org, Razzaq et al., 2005), to collect extensive information while students interact with the system. Our results showed that the assistance model that includes no assessment result on the main problems leads to significantly better predictions than the lean model that is based on the assessment results alone. This relative success of the assistance model over the lean model highlights the power of the assistance measures, which suggests not only is it possible to get reliable information during “teaching on the test”, but also data from the teaching process actually improves reliability.

Although DA has been shown to be effective predicting student performance, yet there is a cautionary note about DA since students are allowed to request assistance: it generally takes longer for students to finish a test using the DA approach than using a traditional test. For instance, in Feng et al. (2009) we reported that we could do a better job predicting student state test score using DA than a contrast case, the traditional testing situation. However, there is a caveat that the DA condition has included more time than the contrast case, which seems unfair for the contrast case. Although this sort of contrast leaves out the instructional benefit (e.g., Razzaq & Heffernan, 2006, 2007; Feng, Heffernan, Beck & Koedinger, 2008) of the tutoring system and, moreover, may not be well received by teachers and students, whether or not the system using DA would yield a better prediction of state scores or learning is still worth of further research. In this paper, we report a study that aims to answer this question.

2 Methods

2.1 ASSISTments, the test bed

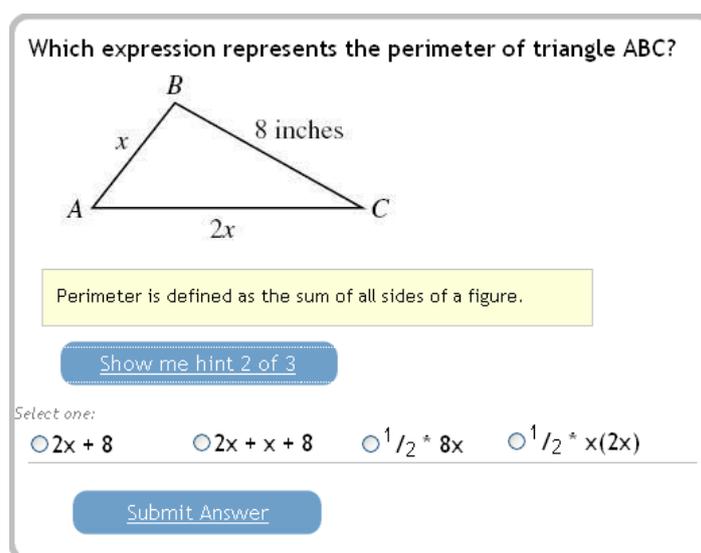


Fig.1. A screenshot showing student requested a hint for one scaffolding question in ASSISTments

Traditionally, the areas of testing (i.e. psychometrics) and instruction (i.e., math educational research and instructional technology research) were separated fields of research with their own goals. The ASSISTments system is an attempt to blend the positive features of both computer-based tutoring and benchmark testing. The online system presents math problems to students of approximately 13 to 16 years old in middle school or high school to solve. If a student gets an item (the **main** item) right, they will get a new item. If a student has trouble solving a problem, the system provides instructional assistance to lead the student through by breaking the problem into a few **scaffolding** steps (typically 3~5 per problem), or displaying **hint** messages on the screen (usually 2~4 per question), upon student request as shown in Fig.1. Although the system is web-based hence accessible in principle anywhere/anytime, students typically interact with the system during one class period in the schools'

computer labs every three or four weeks. As students interact with the system, time-stamped student answers and student actions are logged into the background database. The hypothesis is that ASSISTments can do a better job of assessing student knowledge limitations than practice tests or other online testing approaches by using the DA approach based on the data collected online.

2.2 Approach

Fundamentally, in order to find out whether DA was worth the time, we would want to run a study comparing the assessment value of the following two different conditions:

- **Static assessment condition (A):** students were presented with one static (as opposed to dynamic) test item and were requested to submit an answer. Once they had done that, more static items followed.
- **Dynamic assessment condition (B):** students were presented with one static test item followed by a DA portion where they could request help.

Then the question was: Is condition B better, or at least as good considering the learning effect, at assessing students after we control the time?

We could have conducted a randomized controlled experiment with the two conditions. But, since the logging system of ASSISTments had collected data with the information needed by DA, we chose to compare predictions made based on log data from 40 minutes of time across *simulated* conditions that were similar but not exactly the same as above:

- **Simulated static assessment condition (A')**: 40 minutes of student work selected from existing log data on only main items
- **Dynamic assessment condition (B')**: 40 minutes of work selected from existing log data on both main items and the scaffolding steps and hints

Such a simulation study using educational data mining techniques not only saved time from setting up and carrying out classroom experiments, but also allowed us to compare the same student's work in two different conditions, which naturally rules out the subject effect. There would be no threat to validity of the comparison as both A' and B' allow learning on the test so there was a general trend up that you would expect¹. So we will not devote much attention to the learning value of these conditions. We will refer interested readers to our previous publications where there were evidences showing that the system led to better student learning (e.g. Razzaq & Heffernan, 2006, 2007; Feng, Heffernan, Beck & Koedinger, 2008).

We chose to use student's end of year state accountability test score as the measure of student achievement, and we used data from conditions A' and B' to predict state test scores and compare the predictive accuracy of the two conditions.

¹ Students in the static condition A' potentially could have spent more time in the system considering the tutoring portion following main items.

2.3 Data

The first raw data set we considered came from the 2004 – 2005 school year, the first full year in which the ASSISTment system was used in classes in 2 middle schools in Massachusetts. 912 8th grade students' logs were maintained in the system over the time period from September to May. Among these students, we were able to obtain complete data for 628. The data set contained online interaction data from the ASSISTment system and the results of 8th grade state tests taken in May 2005. Students whose state test scores were not available and those who had done less than 40 minutes of work were excluded.

The second raw data set we used was from the 2005-2006 school year. We collected a full data set for 764 students from Worcester Public Schools, including the online data from ASSISTments and their 8th grade state test raw scores. We applied the same filter to exclude students who had not done enough work. In both years, the items involved in the data sets were given to students in a random order.

For each of the two raw data sets, we prepared two data sets for analysis, one for simulated static assessment condition (A') and one for dynamic assessment condition (B'). The data for condition A' included student response data during the first 40 minutes of work on only main problems; all responses and other actions during the DA portion were ignored. On the contrary, the data for condition B included all the responses for main questions and scaffoldings, as well as hint requests. For instance, consider the following scenario:

Chris spent one minute trying to answer a main question in ASSISTments but failed, and was forced into the tutoring session. Chris then spent four minutes working through the three scaffolding questions. Chris answered one scaffolding question correctly and requested hints for the other two.

This scenario counted as 1 minute of static work among the 40 minutes of data we prepared for condition A' with a response to the main question being recorded as zero. Yet it counted as 5 minutes of dynamic work in the data for condition B', including 1 correct response to scaffolding, 2 incorrect responses to scaffolding and 2 hint requests.

2.4 Metrics

We followed our work in Feng, Heffernan & Koedinger (2006, 2009) of developing online metrics for dynamic testing that measures student accuracy, speed, attempts, and help-seeking behaviors. Simply, the metrics we picked were

- **Main_Percent_Correct**² – students' percent correct on main questions, which we often referred to as the “static metric”.
- **Main_Count** - the number of main items students completed. This measures students' attendance and how on-task they were. This measure also reflects students' knowledge since better students have a higher potential to finish more items in the same amount of time. This is especially true for condition B'

² Student requesting for help before attempting a problem are considered as making an incorrect response to the main question.

where students' work on scaffolding also counted as part of the 40 minute work. While in condition A', low performing kids could go through many items but give wrong answers since their time consumed during the tutoring session is disregarded.

- *Scaffold_Percent_Correct* - students' percent correct on scaffolding questions. In addition to original items, students' performance on scaffolding questions was also a reasonable reflection of their knowledge. For instance, two students who get the same original item wrong may, in fact, have different knowledge levels and this may be reflected in that one may do better on scaffolding questions than the other.
- *Avg_Hint_Request* - the average number of hint requests per question.
- *Avg_Attempt* - the average number of attempts students made for each question.
- *Avg_Question_Time* - on average, how long it takes for a student to answer a question, whether original or scaffolding, measured in seconds.

The last five metrics are DA style metrics and were not measured in traditional tests. They indicate the amount of assistance students needed to finish problems and the amount of time they needed to finish the questions. Our hypothesis is that the last three metrics will be negatively correlated with students' performance. Thereby, the more hints they request, the more attempts they make on a question and the longer they need to go through a question, the worse their performance.

Among the above six metrics, condition A' used only the first one as predictor to simulate paper practice tests by scoring students either correct or incorrect on each main problem while condition B' used all the metrics.

2.5 Modeling

We ran stepwise linear regression³ to use the metrics described above to predict student state test scores. The same process was repeated on the second year's data. For all the models, the dependent variable is the state test score but the independent variables differ. Specifically, for condition A', the independent variable of the simple linear regression model is *Main_Percent_Correct*; while for condition B', it changed to be a collection of metrics: *Main_Percent_Correct*, *Main_Count*, *Scaffold_Percent_Correct*, *Avg_Hint_Request*, *Avg_Attempt*, *Avg_Question_Time*.

2.6 Results

First, we noticed that in both years, students finished more test items in the 40 minutes in static condition than in dynamic condition, which is not surprising considering the tutoring portion in the DA condition. Particularly, in year 2004-2005, the average number of main items finished was 22 in the simulated static assessment condition while it was only 11 in the dynamic condition; in year 2005-2006, the number was 31 in the static condition but it was only 13 in the dynamic condition.

³ probability of F-to-enter $\leq .05$, probability of F-to-remove $\geq .10$

Then, we examined the parameters and associated coefficients in the linear regression models of both conditions.

Table 1. Parameters of simple regression models for simulated static assessment condition (A')

Condition A'	Parameter	Coefficient
2004-2005	Intercept	16.383
	Main_Percent_Correct	24.690
2005-2006	Intercept	13.993
	Main_Percent_Correct	40.479

As shown in Table 2, the first three parameters entered the models were the same in both years (with the order changed a little bit). Scaffold_Percent_Correct was the most significant predictor in the first year while in the second year, it changed to be Main_Percent_Correct. Also, in the later year 2005-2006, Avg_Attempt was considered as a significant predictor while in the first year it was Avg_Hint. Yet, it was consistent with our hypothesis that more attempts or more hints on a question will end up with a lower estimated score.

Table 2. Parameters entered regression models of dynamic condition (B')

Condition B' (2004-2005)	Parameter	Coefficient
0	Intercept	17.090
1	Scaffold_Percent_Correct	16.311
2	Main_Percent_Correct	7.107
3	Main_Count	0.179
4	Avg_Hint	-2.580
Condition B' (2005-2006)	Parameter	Coefficient
0	Intercept	16.061
1	Main_Percent_Correct	21.331
2	Scaffold_Percent_Correct	16.242
3	Main_Count	0.172
4	Avg_Attempt	-2.543

Now that we had looked at the parameters in the regression models, we would examine which condition does a better job predicting state test score. The R square's of all models were summarized in Table 3. Additionally, because the models in different conditions always had different numbers of parameters, we also chose to use Bayesian Information Criterion (BIC) to compare the generalization quality of the models. We applied the formula for linear regression models introduced by Raftery (1995, p135), which was different from what is typical used for calculating BIC but most convenient for linear regression models:

$$BIC = n*\ln(1-R^2) + p*\ln(n)$$

where

n: the sample size (for the 2004-2005 data case, *n* = 628; for the 2005-2006 data, *n*=764)

ln: natural logarithm

p : the number of independent variables included in each model (not including intercept)

Table 3. Summary of models

	R²		BIC	
	2004-2005	2005-2006	2004-2005	2005-2006
Simulated static condition	0.174	0.377	-114	-354
Dynamic condition	0.240	0.426	-147	-398

As we can see from Table 3, in both years, the R square of the model from the dynamic condition was always higher than that of the simulated static condition. Raftery (1995) discussed a Bayesian model selection procedure, in which the author proposed the heuristic of a BIC difference of 10 was about the same as getting a p-value of 0.05. And the lower BIC indicated a better fitted model. Thereby, we can see, in both years, the dynamic assessment condition did a significantly better job at predicting state test scores than the control condition which is static.

2.7 Validation

Before jumping into the conclusion saying dynamic assessment is more efficient than just giving practice test items, we performed 5-fold cross validation on the 2004-2005 data set. For the testing data, we calculated mean absolute difference (MAD) as a measure of prediction accuracy, which was computed as the average of the absolute difference between students' real state test scores and the predicted scores across all students included in the testing set.

Table 4. Results of cross validation

Fold	MAD					Avg.
	1	2	3	4	5	
Simulated static condition	9.44	9.05	8.67	9.01	9.13	9.01
Dynamic condition	9.02	8.65	8.74	9.04	8.57	8.7
p-value (95%) from two-sided paired t-test comparing absolute difference of two conditions	0.35	0.36	0.88	0.94	0.13	0.10

As illustrated in Table 4, out of the 5 folds, DA condition ended up with a lower MAD in 3 folds. On average, DA condition did a better job predicting state test scores in the testing set: The difference between MADs of the DA condition and simulated static condition was bigger in these 3 DA-winning folds, and it was much smaller in the other 2 folds (folds 3 &4). Even though, the results from two-sided paired t-test indicated none of the difference was statistically significant.

Then we took a closer look to see whether the trained regression models of the DA condition were consistent across the 5 folds validation. We found out that the trained models were fairly stable. The four variables as shown in Table 2 (2004-2005 portion), Scaffold_Percent_Correct, Main_Percent_Correct, Main_Count, Avg_Hint entered all five trained models while no other variables have been selected. Scaffold_Percent_Correct was always the most significant predictor across all folds while the entering order of the other variables varied during the stepwise variable selection process. The associated coefficients of the selected variables differed across folds with variance ranging between 0.0 (Main_Count) and 2.4 (Scaffold_Percent_Correct).

As the last step, we took the average of coefficients from the five trained models and applied the model on the full data set of year 2004-2005. The average model from the simulated static condition and the DA condition produced MAD of 9.01 and 8.7 respectively. The paired t-test suggested that there was a marginally significant difference ($p=0.10$)

All in all, based on the results, we conclude that dynamic assessment is more efficient than just giving practice test items. So, not only that students are learning during DA but also DA can produce at least as accurate assessment of student math performance as traditional practice test, even limited by using the same amount of testing time.

This is surprising as students in the dynamic assessment do few problems and yet we get better assessment results. Of course, DA has another major advantage in that kids are learning during the test and therefore are not wasting their time just testing, while the practice tests are not likely to lead to much learning.

3 Conclusion

Dynamic assessment (DA) has been advocated as an interactive approach to conducting assessments to students in the learning systems as it can differentiate student proficiency at the finer grained level. In this paper, we compare dynamic assessment against a tough contrast case where students are doing assessment all the time in order to evaluate efficiency and accuracy of dynamic assessment in a tutoring system.

Contribution: The contribution of this paper lies in that it eliminates the cautionary note about dynamic assessment that says DA will always need a longer time to do as well at assessing students, which further validates the usage of tutoring systems for assessment. ITS researchers have showed the effectiveness of such systems at promoting learning (e.g. Koedinger et al., 1997). This paper adds to that fact and presents a nice result suggesting that maybe, students should take their tests in an ITS as well!

General implication: Combining with our previous findings (Feng, Heffernan & Koedinger, 2006, 2009), this paper tells us that not only we can better assess students while teaching them, but also the assessment can be done efficiently. Our results are important because they provide evidence that reliable and efficient assessment and instructional assistance can be effectively blended. At the *Race to The Top Assessment Competition* public input meetings, experts advocated for computer-based state assessments and argued the tests should be taken more than once a year (U.S. Dept of Ed., 2009). The general implication from this series research suggests that such computer-based, continuous assessment systems are possible to build and that they can be quite accurate and efficient at helping schools get information on their students while allowing student learning at the same time.

Acknowledgements

We would like to acknowledge funding from the Institute of Education Sciences of US Department of Education, the National Science Foundation, the Office of Naval

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

References

1. Brown, A. L., Bryant, N.R., & Campione, J. C. (1983). Preschool children's learning and transfer of matrices problems: Potential for improvement. Paper presented at the Society for Research in Child Development meetings, Detroit.
2. Campione, J.C., Brown, A.L., & Bryant, N.R. (1985). Individual differences in learning and memory. In R.J. Sternberg (Ed.). *Human abilities: An information-processing approach*, 103–126. New York: W.H. Freeman.
3. Campione, J.C. & Brown, A.L. (1985). Dynamic assessment: One approach and some initial data. Technical Report No. 361. Cambridge, MA: Illinois University, Urbana. Center for the Study of Reading. ED269735
4. 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*. 19(3), 2009.
5. Feng, M., Heffernan, N., Beck, J, & Koedinger, K. (2008). Can we predict which groups of questions students will learn from? In Beck & Baker (Eds.). *Proceedings of the 1st International Conference on Education Data Mining*. Montreal, 2008.
6. Feng, M., Heffernan, N. T., & Koedinger, K. R. (2006). Addressing the testing challenge with a web based E-assessment system that tutors as it assesses. *Proceedings of the 15th Annual World Wide Web Conference*. ACM Press: New York.
7. Fuchs, L.S., Compton, D.L., Fuchs, D., Hollenbeck, K.N., Craddock, C.F., & Hamlett, C.L (2008). Dynamic assessment of algebraic learning in predicting third graders' development of mathematical problem solving. *Journal of Educational Psychology*, 100(4), 829-250.
8. Fuchs, D., Fuchs, L.S., Compton, D.L., Bouton, B., Caffrey, E., & Hill, L. (2007). Dynamic assessment as responsiveness to intervention. *Teaching Exceptional Children*, 39 (5), 58-63.
9. Grigorenko, E. L. and Sternberg, R. J. (1998). Dynamic testing. *Psychological Bulletin*, 124, 75–111.
10. 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.
11. Raftery, A. E. (1995). Bayesian model selection in social research. In *Sociological Methodology*, 25, 111-163.
12. 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., & Rasmussen, K.P. (2005). The Assistent Project: Blending Assessment and Assisting. In C.K. Looi, G.McCalla, B. Bredeweg, & J. Breuker (Eds.) *Proceedings of the 12th Artificial Intelligence in Education*. Amsterdam: ISO Press. pp. 555–562.
13. U.S. Dept of Ed. (2009). Transcripts from U.S. Department of Education, Race to the Top Assessment Competition public and expert input meetings. Retrieved Jan, 2010 from <http://www.ed.gov/programs/racetothetop-assessment/boston-transcript-2.pdf>.
14. Razzaq, L., Heffernan, N.T. (2006). Scaffolding vs. hints in the Assistent System. In Ikeda, Ashley & Chan (Eds.). *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*. Springer-Verlag: Berlin. pp. 635-644. 2006.
15. Razzaq, L., Heffernan, N.T., Lindeman, R.W. (2007). What level of tutor interaction is best?. In Luckin & Koedinger (Eds.). *Proceedings of the 13th Conference on Artificial Intelligence in Education*. Amsterdam, Netherlands: IOS Press.
16. Sternburg, R.J., & Grigorenko, E.L. (2001). All testing is dynamic testing. *Issues in Education*, 7, 137-170.
17. Sternburg, R.J., & Grigorenko, E.L. (2002). *Dynamic testing: The nature and measurement of learning potential*. Cambridge, England: Cambridge University Press.

Using Neural Imaging and Cognitive Modeling to Infer Mental States while Using an Intelligent Tutoring System

Jon M. Fincham, John R. Anderson, Shawn Betts and Jennifer L. Ferris

{fincham, ja+}@cmu.edu

Psychology Department, Carnegie Mellon University

Abstract. Functional magnetic resonance imaging (fMRI) data were collected while students worked with a tutoring system that taught an algebra isomorph. A cognitive model predicted the distribution of solution times from measures of problem complexity. Separately, a linear discriminant analysis used fMRI data to predict whether or not students were engaged in problem solving. A hidden Markov algorithm merged these two sources of information to predict the mental states of students during problem-solving episodes. The algorithm was trained on data from one day of interaction and tested with data from a later day. In terms of predicting what state a student was in during any 2 second period, the algorithm achieved 87% accuracy on the training data and 83% accuracy on the test data. Further, the prediction accuracy using combined cognitive model and fMRI signal showed superadditivity of accuracies when using either cognitive model or fMRI signal alone.

1 Introduction

This paper reports an approach of integrating cognitive modeling and neural imaging data to facilitate student modeling in intelligent tutoring systems. Intelligent tutoring systems have proven to be effective in improving mathematical problem solving (14, 20). Their basic mode of operation is to track students while they solve problems and offer instruction based on this tracking. These tutors individualize instruction by two processes called model tracing and knowledge tracing. Model tracing uses a model of students' problem solving to interpret their actions. It tries to diagnose the student's intentions by finding a path of cognitive actions that match the observed behavior of the student. Given such a match, the tutoring system is able to provide real-time instruction individualized to where that student is in the problem. The second process, knowledge tracing, attempts to infer a student's level of mastery of targeted skills and selects new problems and instruction suited to that student's knowledge state. While the principle of individualizing instruction to a particular student holds great promise, the practice is limited by the ability to diagnose what the student is thinking. The only information available to a typical tutoring system comes from the actions that students take in the computer interface. Inferring cognitive state based on such potentially impoverished data can be difficult and brain imaging data might provide a useful augmentation. Recent research has reported a variety of successes in using brain imaging to identify what a person is thinking about (e.g., 7, 10,11,15) and identifying when mental states happen (e.g., 1, 12, 13).

While the methods described here could extend to knowledge tracing, this article will focus on model tracing where the goal is to identify the student's current mental state. Two features of the intelligent tutoring situation shaped our approach to the problem:

1. Given that instruction must be made available in real time, inferences about mental state can only use data up to the current point in time. While inferences of

mental state may become clearer after observing subsequent student behavior, these later data are unavailable for real-time prediction.

2. Model tracing algorithms are parameterized with pilot data and then used to predict the mental state of students in learning situations. Therefore, we trained our algorithm on one set of data and tested it on a later set.

While many distinctions can be made about mental states during the tutor interactions, we focused on two basic distinctions as a first assessment of the feasibility of the approach. The first distinction involved identifying periods of time when students were engaged in mathematical problem solving and periods of time when they were not. The second, more refined, distinction involved identifying what problem they were solving when they were engaged and, further, where they were in the solution of that problem. While one might think only the latter goal would be of instructional interest, detecting when students are engaged or disengaged during algebraic problem solving is by no means unimportant. A number of immediate applications exist for accurate diagnosis of student engagement. For instance, there are often long periods when students do not perform any action with the computer. It would be useful to know whether the student was engaged in the mathematical problem solving during such periods or was off task. If the student was engaged in algebraic problem solving despite lack of explicit progress the tutor might volunteer help. On the other hand, if the student was not engaged, the tutoring system might nudge the student to go back on task.

The research reported here used an experimental tutoring system described in Anderson (2) and Brunstein et al. (5) that teaches a complete curriculum for solving linear equations based on the classic algebra text of Foerster (8). The tutoring system has a

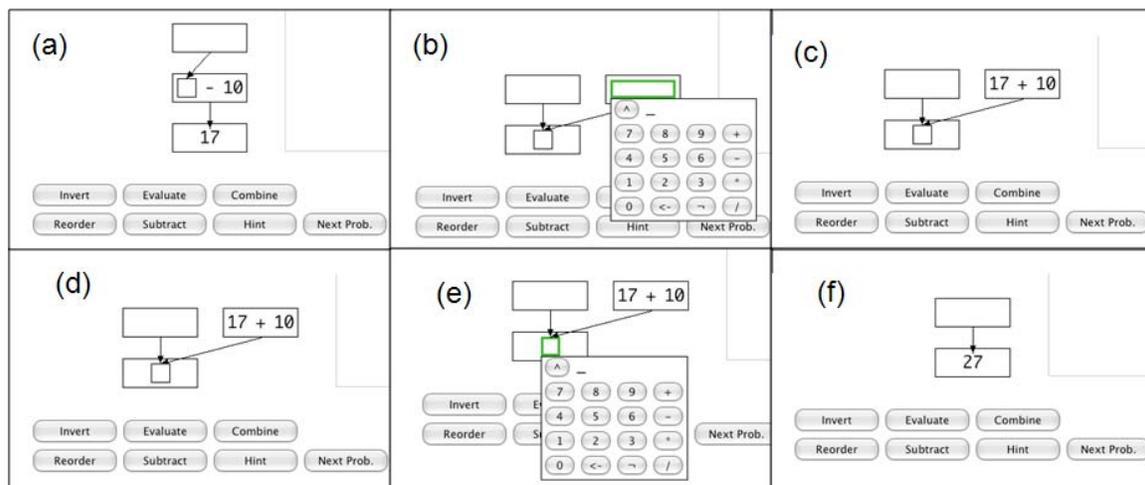


Figure 1 Interface for equation solving isomorph. (a) The student starts out in a state with a data-flow equivalent of the equation $x - 10 = 17$. The student uses the mouse to select this equation and chooses the operation “Invert” from the menu. (b) A keypad comes up into which the student enters the result $17+10$. (c) The transformation is complete. (d) The previous state (data-flow equivalent of $x = 17+10$) is repeated and the student selects $17+10$ and chooses the operation “Evaluate”. (e) A keypad comes up into which the student will type 27 . (f) The evaluation is complete.

minimalist design to facilitate experimental control and detailed data collection. It presents instruction, provides help when requested, and flags errors during problem solving. In addition to teaching linear equations to children, this system can be used to teach rules for transforming data-flow graphs that are isomorphic to linear equations. The data-flow system has been used to study learning with either children or adults and has the virtue of not interfering with instruction or knowledge of algebra. The experiment reported here uses this data-flow isomorph with an adult population. Figure 1 illustrates sequences of tutor interaction during a problem isomorphic to the simple linear equation $x - 10 = 17$. The interactions with the system are done with a mouse that selects parts of the problem to operate on, actions from a menu, and enters values from a displayed keypad.

2 Experiment

Twelve students went through a full curriculum based on the sections in the Foerster text for transforming and solving linear equations. The experiment spanned six days. On Day 0, students practiced evaluation and familiarized themselves with the interface. On Day 1, three critical sections were completed with functional magnetic resonance imaging (fMRI). On Days 2-4 more complex material was practiced outside of the fMRI scanner.

On Day 5 the three critical sections (with new problems) were repeated, again in the fMRI scanner. Each section on Days 1 and 5 involved 3 blocks during which they would solve 4 to 8 problems from the section. Some of the problems involved a single transformation-evaluation pair as in Figure 1 and others involved 2 pairs (problems studied on Days 2-4 could involve many more operations). Periods of enforced off-task time were created by inserting a 1-back task (17) after both transformation and evaluation steps. A total of 104 imaging blocks were collected on Day 1 and 106 were collected on Day 5 from the same 12 students. Average time for completion of a block was 207 2-second scans with a range from 110 to 349 scans. The duration was determined both by the number and difficulty of the problems in a block and by the students' speed.

Students solved 654 problems on Day 1 and 664 on Day 5. 76% of the problems on both days were solved with a perfect sequence of clicks. Most of the errors appeared to reflect interface slips and calculation errors rather than misconceptions. Each problem involved one or more of the following types of intervals:

1. Transformation (steps a-c in Figure 1): On Day 1 students averaged 8.2 scans with a standard deviation of 5.9 scans. On Day 5 the mean duration was 5.9 scans with a standard deviation of 4.1.
2. 1-back within a problem: This was controlled by the software and was always 6 scans.
3. Evaluation (steps d-f in Figure 1): Students took a mean of 4.9 scans on Day 1 with a standard deviation of 3.6; they took 3.8 scans on Day 5 with a standard deviation of 2.7.
4. Between Problem Transition: This involved 6 scans of 1-back, a variable interval determined by how long it took students to click a button saying they were done, and 2 scans of a fixation cross before the next problem. This averaged 9.1 scans with a standard deviation of 1.5 scans on both days.

In addition there were 2 scans of a fixation cross before the first problem in a block and a number of scans at the end which included a final 1-back but also a highly variable period of 6 to 62 scans before the scanner stopped. The mean of this end period was 11.0 scans and the standard deviation was 6.5 scans.

The student-controlled intervals 1 and 3 show a considerable range, varying from a minimum of 1 scan to a maximum of 54 scans. Anderson (2) and Anderson et al. (3) describe a cognitive model that explains much of this variance. For the current purpose of showing how to integrate a cognitive model and fMRI data, the complexity of that model would distract from the basic points. Therefore, we instead adapt a keystroke model (6) based on the fact that cognitive complexity is often correlated with complexity in terms of physical actions. Such models can miss variability that is due to more complex factors, but counting physical actions is often a good predictor.

We will use number of mouse clicks as our measure of complexity. As an example of the range in mouse clicks – it takes 15 clicks in the tutor interface to accomplish the following transformation¹:

$$\frac{1000 * X}{-10} \Rightarrow \frac{1000}{-10} * \frac{X}{-10}$$

but only 5 clicks to accomplish the evaluation:

$$X = 7 - 5 \Rightarrow X = 2$$

Transformation steps take longer than evaluation steps because they require more clicks (average 10.4 clicks versus 6.8). Figure 2 illustrates the systematic relationship that exists between mouse clicks required to accomplish an operation and the time that the operation took. The average scans per mouse click decreases from .77 scans on Day 1 to .57 on Day 5. On the other hand the average ratio shows little difference between transformations (.69 scans) and evaluations (.65 scans) and so Figure 2 is averaged over transformations and evaluations. As the figure illustrates, the number of scans for a given number of mouse clicks is approximately distributed as a log-normal distribution. Log-normal distributions estimated from Day 1 were part of the algorithm for identifying mental state. The only adjustment for Day 5 was to speed up the mean of the distribution by a constant 0.7 factor (based on Anderson (2), model in that volume figure 5.7) to reflect learning. Thus, the prediction for Day 5 is $.77 * .7 = .54$ scans per click.

2.1 Imaging Data

Anderson et al. (3) describe an effort to relate fMRI activity in predefined brain regions to a cognitive model for this task. However, as with the latency data, the approach here makes minimal theoretical assumptions. We defined 408 regions of interest (ROIs), each approximately a cube with sides of 1.3 cm that cover the entire brain. For each scan for each region, we calculated the percent change in the fMRI signal for that scan from a baseline defined as the average magnitude of all the preceding scans in that block. We used this signal to identify On periods when a student was engaged in problem solving (evaluation and transformation in Figure 1) versus Off periods when the student was engaged in n-back or other beginning and ending activities. A linear discriminant analysis

¹ For brevity we will give the standard algebraic equivalent of data-flow graphs.

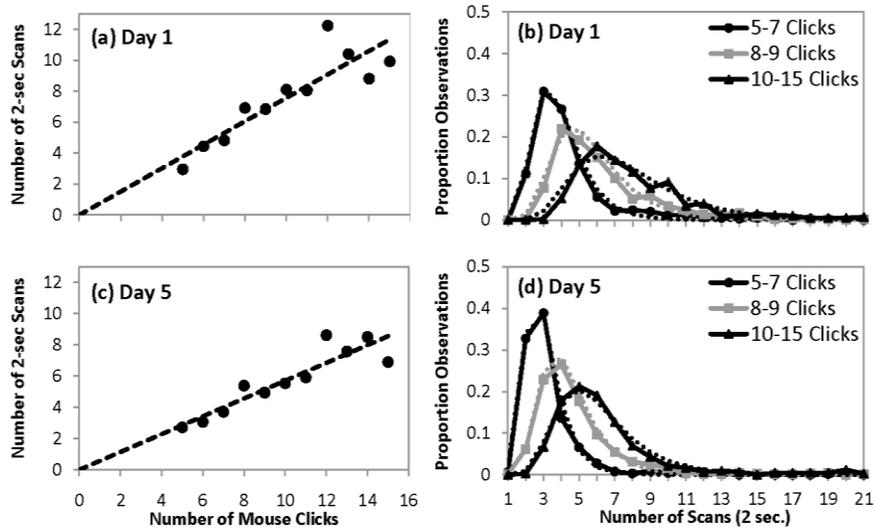


Figure 2. (a) and (c): The relationship between number of clicks and duration of problem solving in terms of number of 2-sec scans. (b) and (d): Distributions of number of scans for different numbers of clicks and log-normal distributions fitted to these.

was trained on the group data from Day 1 to classify the pattern of activity in the 408 regions as reflecting an On scan or an Off scan. Figure 3a shows how accuracy of classifying a target scan varied with the distance between the target scan and the scan whose activity was used to predict it. It plots a d-prime measure (9), which is calculated from the z-transforms of hit and false alarm rates. So, for instance, using the activity 2 scans after the target scan, 91% of the 7761 Day 5 On scans were correctly categorized and 16% of 11835 Off scans were false alarmed yielding a d-prime of 2.34. Figure 3 shows that best prediction is obtained using activity 2 scans or 4 seconds after the target scan. Such a lag is to be expected given the 4-5 second delay in the hemodynamic response. The d-prime measure never goes down to zero reflecting the residual statistical structure in the data.

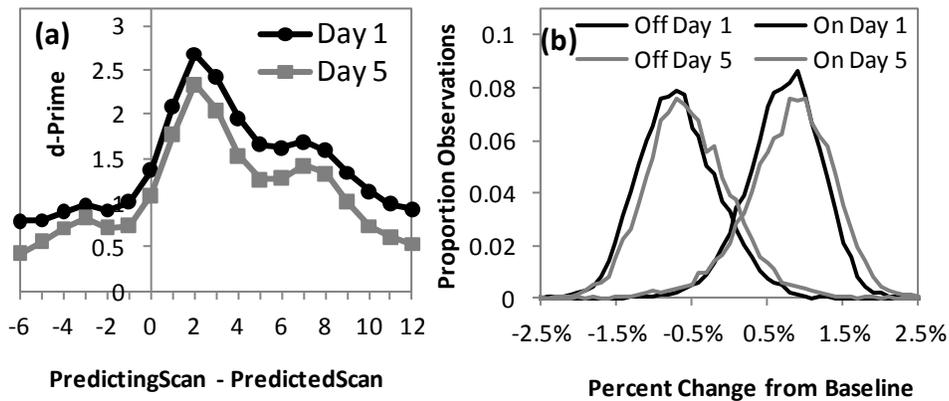


Figure 3. (a) Accuracy of classification as a function of the offset between the scan whose activity is being used and the scan whose state is being predicted. (b) Distribution of fMRI signal changes for Day 1 and Day 5 On and Off scans using an offset of 2. All 408 regions are used.

While we will report on the results using a lag of 0, the main application will use the optimal lag 2 results – meaning it was 4 seconds behind the student. Little loss occurs in d-prime going from training data to predicted data. The relatively large number of scans (21,826 on Day 1 and 19,596 on Day 5) avoids overfitting with even 408 regions. While our goal is to go from Day 1 to Day 5, the results are almost identical if we use Day 5 for training and Day 1 for testing. The weights estimated for the 408 regions can be normalized (to have a sum of squares of 1) and used to extract an aggregate signal from the brain. This is shown in Figure 3b for the On and Off scans on the two days.

2.2 Predicting Student State.

Predicting whether a student is engaged in problem solving is a long way from predicting what the student is actually thinking. As a first step to this we took up the challenge of determining which problem a student was working on in a block and where a student was in the problem. This amounts to predicting what equation the student is looking at. Figure 4.1 illustrates an example from a student working on a set of 5 equations. As the figure illustrates, each equation goes through 4 forms on the way to the solution: the first and third require transformation operations while the second and fourth require evaluation operations (see Figure 1). Adding in the 21 Off states between forms there are 41 states. Consider the task of predicting the student state on scan 200. Information available to the algorithm includes the 5 problems, the distributions of lengths for the various states, and that there are 41 states in all. The classifier additionally provides the probability that each of scans 1-200 came from an On state or an Off state. The algorithm must integrate this knowledge into a prediction about what state, from 1 to 41, the student is in at scan 200.

A key concept is an interpretation. An interpretation assigns the m scans to some sequence of the states $1, 2, \dots, r$ with the constraint that this is a monotonic non-decreasing sequence beginning with 1. For example, assigning 10 scans each to the states 1 to 20 would be one interpretation of the first 200 scans in Figure 4.1. Using the naïve Bayes rule, the probability of any such interpretation, I , can be calculated as the product of prior probability determined by the interval lengths and the conditional probabilities of the fMRI signals given the assignment of scans to On and Off states:

$$p(I | fMRI) \propto \left[S_r(a_r) \prod_{k=1}^{r-1} p_k(a_k) \right] * \left[\prod_{j=3}^{m+2} p(fMRI_j | I) \right]$$

The first term in the product is the prior probability and the product in the second term is the conditional probability. The terms $p_k(a_k)$ in the prior probability are the probabilities that the k th interval is of length k and $S_r(a_r)$ is the probability the r th interval surviving at least as long as a_r . These can be determined from Figure 2 for On intervals and from the experimental software for Off intervals. The second term contains $p(fMRI_j | I)$, which are the probabilities for the combined fMRI signal on scan $j+2$ given I 's assignment of scan j to an On or a Off state. The linear classifier determines these from normal distributions fitted to the curves in Figure 3b.

To calculate the probability that a student is in state r on any scan m one needs to sum the probabilities of all interpretations of length m that end in state r . This can be efficiently calculated by a variation of the forward algorithm associated with hidden Markov models² (HMMs, 19). The predicted state is the highest probability state. The most common HMM algorithm is the Viterbi algorithm, a dynamic programming algorithm that requires knowing the end of the event sequence to constrain interpretations of the events. The algorithm we use is an extension of the forward algorithm associated with HMMs and does not require knowledge of the end of the event sequence. As such it can be used in real time and is simpler. Figure 4.1 illustrates the performance of this algorithm on a block of problems solved by the first student. Figure 4.1a shows the 20 forms of the 5 equations. Starting in an Off state, going through 20 On states, and ending in an Off state, the student goes through 41 states. Figure 4.1b illustrates in maroon the scans on which the algorithm predicts that the student is engaged on a particular equation form. Predictions are incorrect on 19 of the 241 scans but never off by more than 1 state. In 18 of these cases it is one scan late in predicting the state change and in 1 case it is one scan too early.

Going beyond showing 1 student during 1 block, Figure 4.2 shows the average performance over the 104 blocks on Day 1 and the 106 blocks on Day 5.

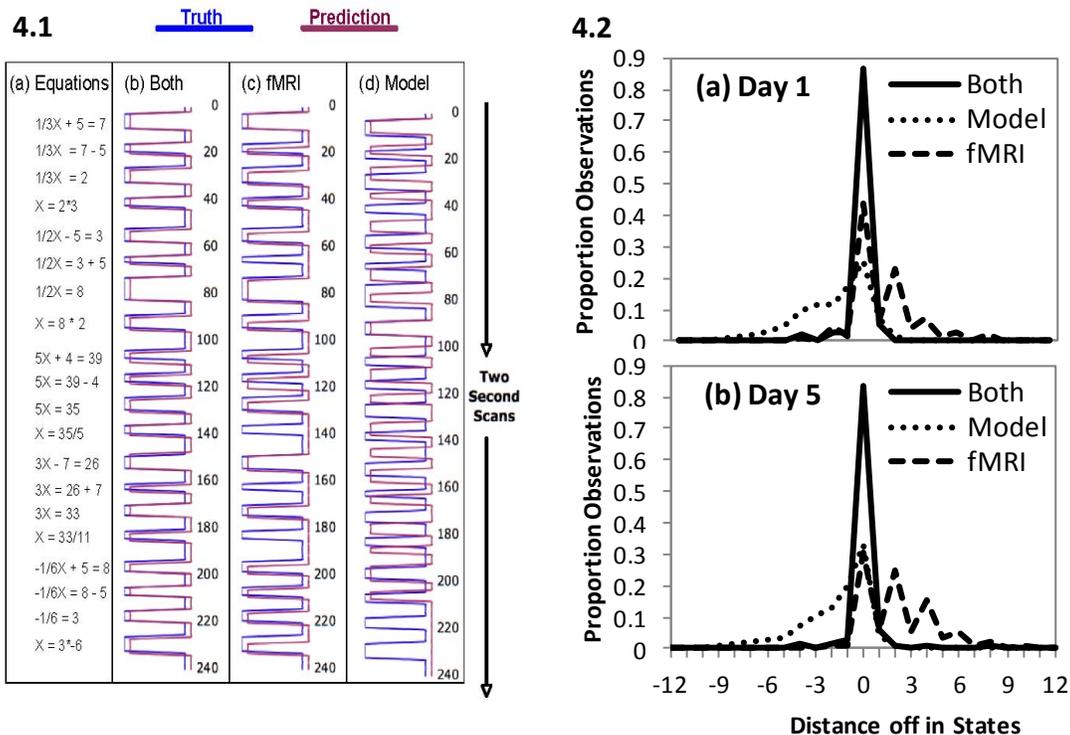


Figure 4. (4.1) An example of an experimental block and its interpretations. The sequence of equations is shown in column a. Columns b, c, and d compares attempts at predicting the states with both fMRI and model, just fMRI, or just model. On scans (when no equation is on the screen) are to the left and Off times (when an equation is on the screen) are to the right. (4.2) Performance, measured as the distance between the actual state and the predicted state, using both cognitive model and fMRI, just fMRI, or just a cognitive model on (a) Day and (b) Day 5.

² Since the states are not directly observable and their durations are variable our model is technically a hidden semi-Markov process (16).

Performance is measured in terms of the distance between the actual and predicted states in the linear sequence of states in a block. A difference of 0 indicates that the algorithm correctly predicted the state of the scan, negative values are predicting the state too early, and positive values are predicting the state too late. The performance of the algorithm is given in the curve labeled “Both”. On Day 1 it correctly identifies 86.6% of the 22138 scans and is within 1 state (usually meaning the same problem) on 94.4% of the scans. Since all parameters are estimated on Day 1, the performance on Day 5 represents true prediction: It correctly identifies 83.4% of the 19914 scans on Day 5 and is within 1 state on 92.5% of the scans. To provide some comparisons, Figure 4.2 shows how well the algorithm could do given only the simple behavioral model or only the fMRI signal.

The fMRI-only algorithm ignores the information relating mouse clicks to duration and sets the probability of all lengths of intervals to be equal. In this case, the algorithm tends to keep assigning scans to the current state until a signal comes in that is more probable from the other state. This algorithm gets 43.9% of the Day 1 scans and 30.6% of the Day 5 scans. It is within 1 scan on 51.8% of the Day 1 scans and 37.3% of the Day 5 scans. Figure 4.1c illustrates typical behavior -- it tends to miss pairs of states. This leads to the jagged functions in Figure 4.2 with rises for each even offset above 0.

The model-only algorithm ignored the fMRI data and set the probability of all signals in all states to be equal. Figure 4.1d illustrates typical behavior. It starts out relatively in sync but becomes more and more off and erratic over time. It is correct on 21.9% of the Day 1 scans and 50.4% of the Day 2 scans. It is within 1 scan on 32.9% of the Day 1 scans and 56.9% of the Day 5 scans.

The performances of the fMRI-only and model-only methods are quite dismal. Successful performance requires knowledge of the probabilities of both different interval lengths and different fMRI signals.

Conclusions

The current research attempted to hold true to two realities of tutor-based approaches to instruction. First, the model-tracing algorithm must be parameterized on the basis of pilot data and then be applied in a later situation. In the current work, the algorithm were parameterized with an early data set and tested on a later data set. Second, the model-tracing algorithm must provide actionable diagnosis in real time – it cannot wait until all the data are in before delivering its diagnosis. In our case, the algorithm provided diagnosis about the student’s mental state in almost real time with a 4 second lag. Knowledge tracing, which uses diagnosis of current student problem solving to choose later problems, does not have to act in real time and can wait until the end of the problem sequence to diagnose student states during the sequence. In this case one could also use the Viterbi algorithm for HMMs (19) that takes advantage of the knowledge of the end of the sequence to achieve higher accuracy. On this data set the Viterbi algorithm is able to achieve 94.1% accuracy on Day 1 and 88.5% accuracy on Day 2. Moreover, prediction accuracy using both information sources was substantially greater than using either data

source alone. A Bayesian analysis can explain the basis of the apparent superadditivity of prediction accuracy when using the combined information sources. The odds of a scan being On given the model and the fMRI signal can be expressed:

$$\text{Odds(On | Model \& Signal)} = \text{Odds(On | Model)} * \text{Likelihood-ratio(Signal| On \& Model)}$$

If

(a) **Likelihood-ratio(Signal| On \& Model) = Likelihood-ratio(Signal| On)** -- that is, the signal magnitude depends only on whether the state is On,

(b) **Odds(On) = 1** -- that is, that On scans and Off scans are equally frequent, which is approximately true, and therefore $\text{Likelihood-ratio(Signal| On)} = \text{Odds(On| Signal)}$,

then the equation above can be rewritten

$$\text{Odds(On | Model \& Signal)} = \text{Odds(On | Model)} * \text{Odds(On| Signal)},$$

or by inverting the odds

$$\text{Odds(Off | Model \& Signal)} = \text{Odds(Off| Model)} * \text{Odds(Off| Signal)}.$$

These two equations show there is a multiplicative relationship in the Odds(Correct Acceptance) and Odds(Correct Rejection). Increasing either the strength of the signal or the strength of the model multiplies the effectiveness of the other factor

This experiment has shown that it is possible to combine brain imaging data with a cognitive model to provide a fairly accurate diagnosis of where a student is in episodes that last as long as 10 minutes. Moreover, prediction accuracy using both information sources was substantially greater than using either source alone. The performance in Figure 4.2 is by no means the highest level of performance that could be achieved. Performance depends on how narrow the distributions of state durations are (Figures 2b and 2d) and the degree of separation between the signals from different states (Figure 3b). The model leading to the distributions of state durations was deliberately simple, being informed only by number of clicks and a general learning decrease of .7 from Day 1 to Day 5. More sophisticated student models like those in the cognitive tutors would allow us to track specific students and their difficulties leading to much tighter distributions of state durations. On the data side, improvement in brain imaging interpretation would lead to greater separation of signals. Finally, other data like eye movements could provide additional features for a multivariate pattern analysis.

References

- [1] Adelnour, F. & Huppert, T. (2009). Real-time Imaging of human brain function by near-infrared spectroscopy using an adaptive general linear model. *NeuroImage*, 46, 133-143.
- [2] Anderson, J. R. (2007). *How Can the Human Mind Occur in the Physical Universe?* New York: Oxford University Press.
- [3] Anderson, J. R., Betts, S. A., Ferris, J. L., & Fincham, J. M. (in press). Can Neural Imaging be Used to Investigate Learning in an Educational Task? In J. Staszewski (Ed.) *Expertise and Skill Acquisition: The Impact of William G. Chase*.

- [4] Anderson, J. R., Qin, Y., Sohn, M-H., Stenger, V. A. & Carter, C. S. (2003). An information-processing model of the BOLD response in symbol manipulation tasks. *Psychonomic Bulletin & Review*, 10, 241-261.
- [5] Brunstein, A., Betts, S., & Anderson, J. R. (2009). Practice enables successful learning under minimal guidance. *Journal of Educational Psychology*, 101, 790-802.
- [6] Card, S. K., Moran, T. P., & Newell, A. (1983). *The psychology of human-computer interaction*. Hillsdale, NJ: Erlbaum.
- [7] Davatzikos, C., Ruparel, Fan, Y., Shen, D. G., Acharyya, M., Loughead, J. W., Gur, W. R., & Langleben, D. D. (2005). Classifying spatial patterns of brain activity with machine learning methods: application to lie detection. *Neuroimage*, 28, 663–668.
- [8] Foerster, P. A. (1990). *Algebra I*, 2nd Edition. Menlo Park, CA: Addison-Wesley Publishing.
- [9] Green, D. M. & Swets, J. A. (1966). *Signal Detection Theory and Psychophysics*. New York, NY: Wiley.
- [10] Haxby, J.V., Gobbini, M. I., Furey, M. L., Ishai, A., Schouten, J. L., & Pietrini, P. (2001). Distributed and overlapping representations of faces and objects in ventral temporal cortex. *Science*, 293, 2425–2430.
- [11] Haynes, J. D. & Rees, G. (2005). Predicting the stream of consciousness from activity in human visual cortex. *Current Trends in Biology*, 15, 1301–1307.
- [12] Haynes, J.D., Sakai, K., Rees, G. Gilbert, S., Frith, C., & Passingham, R. E. (2007). Reading hidden intentions in the human brain. *Current Trends in Biology*, 17, 323–328.
- [13] Hutchinson, R., Mitchell, T.M., Niculescu, R.S., Keller, T. A., Rustandi, I. & Mitchell, T. M. (2009). Modeling fMRI data generated by overlapping cognitive processes with unknown onsets using Hidden Process Models. *NeuroImage*, 46, 87-104.
- [14] Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education*, 8, 30-43.
- [15] Mitchell, T. M., Shinkareva, S. V., Carlson, A., Chang, K-M., Malave, V. L., Mason, R. A., & Just, M. A. (2008). Predicting human brain activity associated with the meanings of nouns. *Science*, 320, 1191-1195.
- [16] Murphy, K. (2002). *Hidden semi-Markov models*. Technical Report, MIT AI Lab.
- [17] Owen, A., Laird, A., & Bullmore, E. (2005). N-back working memory paradigm: A meta-analysis of normative functional neuroimaging studies. *Human Brain Mapping*, 25, 46-59.
- [18] Qin, Y., Anderson, J. R., Silk, E., Stenger, V. A., & Carter, C. S. (2004). The change of the brain activation patterns along with the children's practice in algebra equation solving. *Proceedings of National Academy of Sciences*, 101, 5686-5691.
- [19] Rabiner, R. E. (1989). A tutorial on Hidden Markov Models and selected applications in speech recognition. *Proceedings of the IEEE*, 77 (2): 257-286
- [20] Ritter, S., Anderson, J. R., Koedinger, K. R., & Corbett, A. (2007). Cognitive Tutor: Applied research in mathematics education. *Psychonomic Bulletin & Review*, 14, 249-255.

Using multiple Dirichlet distributions to improve parameter plausibility

Yue Gong, Joseph E. Beck and Neil T. Heffernan

{ygong, josephbeck, nth}@wpi.edu

Computer Science Department, Worcester Polytechnic Institute

Abstract. Predictive accuracy and parameter plausibility are two major desired aspects for a student modeling approach. Knowledge tracing, the most commonly used approach, suffers from local maxima and multiple global maxima. Prior work has shown that using Dirichlet priors improves model parameter plausibility. However, the assumption that all knowledge components are from a single Dirichlet distribution is questionable. To address this problem, this paper presents an approach to integrate multiple distributions and Dirichlet priors. We show that modeling groups of students separately based on their distributional similarities produces model parameters that provide a more plausible picture of student knowledge, even though the proposed solution did not improve the model's predictive accuracy. We also show Dirichlet priors might be hurt by outliers and models with trimming work better.

1 Introduction

In educational research, one fundamental goal is assessing students and estimating constructs, such as their knowledge levels, behaviors, goals and mental states, etc. Since most of those attributes are difficult to directly measure, the technique of student modeling has been widely used for estimating latent characteristics. A common evaluation of student modeling focuses on how well the model fits the training data and how well the model can generalize to unseen test data. However, there has been increasing research focusing on utilizing the model parameters to answer scientific questions [e.g., 1]. Since we are interpreting the model's parameters, we need some means of validating the model's *parameters*, not just its *predictions*. We call this property parameter plausibility. In this paper, we extended our prior work [2], investigating new approaches for improving the student model in terms of predictive accuracy and parameter plausibility. First, we provide some background into our student modeling framework, knowledge tracing, and its problems. We also illustrate the weaker points in our prior work and present a method that overcomes that limitation.

1.1 Knowledge tracing model

Corbett and Anderson style knowledge tracing (KT) [3] has been successfully used in many tutoring systems to estimate a student's knowledge of a skill. It is based on a 2-state hidden Markov model where the student performance is observable, whereas his knowledge is latent. There are two parameters *slip* and *guess*, which mediate student knowledge and student performance. These two parameters are called the performance parameters in the model. The guess parameter represents the fact that the student may sometimes generate a correct response in spite of not knowing the correct skill. The slip parameter acknowledges that even students who understand a skill can make an occasional careless mistake.

In addition to the two performance parameters, there are two learning parameters. The first is *prior knowledge* (K_0), the likelihood the student knows the skill when he first uses the tutor. The second learning parameter is *learning*, the probability a student will acquire a skill as a result of an opportunity to practice it. Every skill to be tracked has these four parameters, *slip*, *guess*, K_0 , and *learning*, associated with it.

1.2 The problem and proposed solution

How to estimate the model parameters is an important issue. There are a variety of model fitting approaches. The Expectation Maximization (EM) algorithm is one of the most commonly used methods. It finds parameters that maximize the data likelihood (i.e. the probability of observing the student performance data). Compared to other model fitting approaches for KT, using EM to learn the parameters has been found to achieve the highest predictive accuracy [4]. However, it still suffers two major problems that are inherent in the KT model's search space: local maxima and multiple global maxima [2,5].

Local maxima are common in many error surfaces. The issue is that the algorithm has to start with some initial value of each parameter, and its final parameter estimates are sensitive to those initial values. The second difficulty, multiple global maxima, is known as identifiability and means that for the same model, given the same data, there are multiple (differing) sets of parameter values that fit the data equally well. Based on statistical methods, there is no way to differentiate which set of parameters is preferable to the others. Consequently, we have to be more careful to select the parameters' initial values when using EM to fit the model, as we want to neither be stuck with some local maxima, nor get unbelievable parameters which are meaningless for making scientific claims, even if those parameters make accurate predictions.

In order to solve the problems, in the previous work [2], we proposed that, rather than using a single fixed value to initialize the conditional probability table when training a knowledge tracing model, it is possible to use Dirichlet priors to start the algorithm. Briefly speaking, we assumed each parameter's values are drawn from Dirichlet distribution, which is specified by a pair of numbers (α , β). The two numbers specify not only the most likely value for a parameter, but also the confidence in the estimate. The Dirichlet priors, which usually represent the researchers' prior beliefs, provide a reasonable starting point and bias the model-fitting process, thus decreasing the probability of ending with an implausible value.

Modeling all skills using the same set of Dirichlet priors assumes that all knowledge components are drawn from a single Dirichlet distribution. That is to say, knowledge components are assumed to have distributional similarities with each other in terms of all four attributes, prior knowledge, guess, slip and learning. Therefore, Dirichlet priors provide bias to all skills towards the mean of the distribution, especially to those abnormal outlier skills. In general, outliers could arise due to lack of sufficient observations. Specifically, with sparse data, the model is trained with few constraints from the evidence; thus although it achieves the highest predictive accuracy it could get, still generates implausible parameter estimates. In this situation, we argue that it is

preferable to have parameters which are more similar to the other, better-estimated, skills. As shown in Figure 1, Skill A and Skill B are at the tail of the distribution. By using Dirichlets, those outliers are biased towards the mean of the distribution. The hypothesis is that it is probably good that they are moved towards the center.

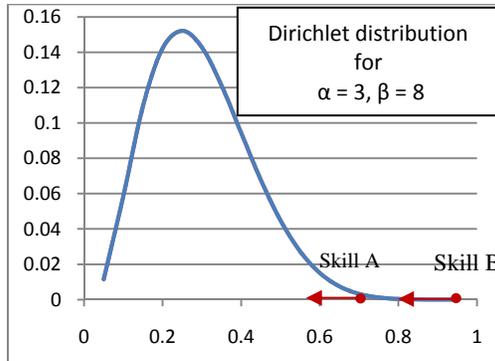


Figure 1 Dirichlet distribution with two outliers

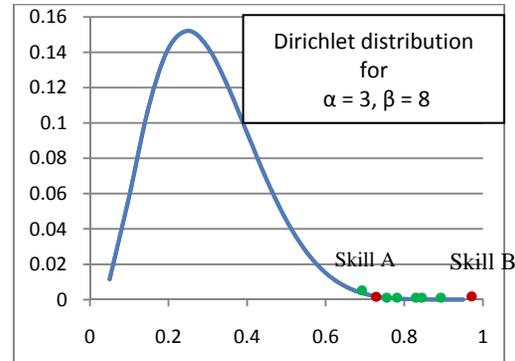


Figure 2 Dirichlet with more “outliers”

1.3 The problem with a single Dirichlet distribution

Dirichlets has been shown to work well on positively biasing outliers [2,5]. However, a key question was overlooked: are the outliers really outliers?

Since the assumption of using Dirichlets is that skills in the domain are from a single distribution, those skills which are located further away from the mean are considered outliers. However, is it really true that all skills are from the same distribution? As shown in Figure 2, which has the same distribution as the one in Figure 1, if there are additional skills, with similar parameter estimates to Skills A and B, perhaps they are not really outliers. A plausible hypothesis is that they are from a cluster of skills which behave differently, i.e. they were not drawn from the same Dirichlet distribution as the other skills. If so, then moving them towards the mean may be inappropriate as they are better modeled as a separate distribution.

2 Methodology

2.1 Clustering

We used clustering to discover which skills should be modeled separately with their own distribution. In the current context, a skill cluster is considered a region in the knowledge tracing parameter space where the skills share similar patterns with respect to the four knowledge tracing parameters. For example, possibly a group of skills might be described as “not previously known (low K_0), but easy to learn (high learning)”, or “hard to learn, but students have partial incoming knowledge”. The intuition is that the skills within a group are spatially located close to each other in the parameter space.

We used K-means clustering to identify the skill clusters. We did not use any self-adaptive clustering variants to automatically determine the number of clusters. The reason for this is that it is hard to evaluate the appropriate number of clusters, as our goal

is to find the clusters that will result in good predictive accuracy and parameter plausibility when modeled as Dirichlets. We had no *a priori* reason to believe that an automated clustering approach would optimize our metrics. Therefore, we used iteration until the number of clusters that works best on an unseen test set was found (i.e. we observed overfitting beginning to occur).

2.2 *Trimming the data to improve Dirichlet parameter estimation*

There are several approaches to setting Dirichlet prior values. One approach is using knowledge of the domain [e.g. 5]. If someone knows how quickly students tend to master a skill or the likelihood of knowing a skill, that knowledge can be used to set the priors. One problem with this approach is that it is not necessarily replicable, as different domains, subjects, and experts may give different answers. Therefore, following the methods in [2], we automatically derived the Dirichlet priors from the data.

It is important to note, however, that automatically calculated Dirichlets are susceptible to undue influence from outliers. Similar to calculating the arithmetic mean, outliers can distort the parameter estimates. In order to address this problem, we trimmed outliers from the data using two different approaches.

The first approach was value-oriented, in which we processed the four knowledge tracing parameters separately and trimmed out the largest and smallest 5% of the values. For example, the 0.001 learning rate of the Pythagorean Theorem skill was in the lowest 5% and thus removed, while the more believable 0.45 prior knowledge estimate was not. The second approach was skill-oriented, in which we calculated the relative distance between a skill's parameters and its cluster's centroid. Those skills furthest away from the centroid were considered outliers, 10% of which we trimmed from the data used to calculate the Dirichlet priors.

2.3 *Training with multiple Dirichlet distributions*

To compare the parameter plausibility and predictive accuracy of the fixed, single-Dirichlet prior and the multiple Dirichlet prior models, we trained a KT model on each of them using the following approach:

```

TrainWithMultiDirichlets (model, data)
1  [prior knowledge, guess, slip, learning] := EM (model, data, fixed prior());
2  for  $k := 1$  to  $n$ 
3    if ( $k \neq 1$ ) clusters[] := K-means ([prior knowledge, guess, slip, learning],  $k$ );
4    else cluster[1] := [prior knowledge, guess, slip, learning];
5    for  $i := 1$  to  $k$ 
6      for each dimension  $d$  from [prior knowledge, guess, slip, learning] $_{k,i}$ 
7        Dirichlet priors[] ( $\alpha, \beta$ ) := CalculateDirichlets( $d$ );
8    [prior knowledge, guess, slip, learning] $'_{k,i}$  := EM (model, data in cluster  $_{k,i}$ , Dirichlet priors[]);

```

We trained a knowledge tracing model for each skill using the same set of fixed priors for EM initialization. After finding each skill's parameter set (prior knowledge, guess, slip,

learning), we calculated the priors for a single Dirichlet distribution and reestimated the KT model. For multiple Dirichlet distributions, we classified the parameter sets into k clusters. For each cluster, we calculated its own Dirichlet priors, and then used those to initialize the EM algorithm and reestimated the models. We didn't specify an upper bound on the number of clusters (i.e. the value of n), as the number of clusters should depend on the improvement of predictive accuracy and parameter plausibility rather than the statistical properties of the clusters.

2.4 Data

For this study, we used data from ASSISTment, a web-based math tutoring system. The data are from 345 twelve- through fourteen- year old 8th grade students in urban school districts of the Northeast United States. They were from four classes, each of which only lasted one month. These data consisted of 92,180 log records of ASSISTment during Dec. 2008 to Apr. 2009. Performance records of each student were logged across time slices for 105 skills (e.g. area of polygons, Venn diagram, division, etc). We took 20% of the students as the unseen test subjects. Their performance records are our test data.

3 Results

We used BNT-SM [7] to apply the EM algorithm to estimate the KT model's parameters. Following the above procedure in Section 2.3, we trained several models to fit the test dataset. We compared the models focusing on the model's predictive accuracy and parameter plausibility.

3.1 Predictive Accuracy

We measured the models' predictive accuracy on the unseen test data set using two metrics: AUC (Area Under ROC Curve) and R^2 .

In Table 1, for the three models with Dirichlets, the Dirichlet priors are calculated based on the trimmed data (since that gave slightly better results). We see the AUC values don't show any difference among the first four models. The values remain unchanged even we considered the possibility that skills come from multiple distributions (shown in the third and fourth rows). R^2 also didn't show any meaningful differences.

Since Ritter et al. have found that predictive accuracy when using the cluster centers is not much worse than when each individual skill's parameters are used, we decided to see if that result will replicate on our data set. Therefore, we evaluated the models using the cluster centers to predict the test data. The results in the last two rows of Table 1 showed that AUC values are similar to their counterparts, but R^2 values are lower (0.053 vs. 0.071 and 0.056 vs. 0.072), suggesting that compared to the predictions done by the models with parameters estimated for each skill, using generic cluster information to fit the data achieves less accurate, but possibly still acceptable, predictions.

These results show that predictive accuracy is not improved by using Dirichlets even with multiple distributions. Related to the prior work [4] where we evaluated the predictive

accuracy of the knowledge tracing model using a variety of model fitting approaches, and also the Performance Factor Analysis model [8], it seems that improving the model’s predictive accuracy on the unseen students is a very difficult task.

Table 1. Comparison of AUC and R²

	AUC	R²
Fixed	0.66	0.072
Single Dirichlet	0.66	0.071
2 Dirichlet distributions	0.66	0.071
3 Dirichlet distributions	0.66	0.072
2 distributions (cluster center)	0.64	0.053
3 distributions (cluster center)	0.65	0.056

3.2 *Parameter plausibility*

In addition to using models for prediction, educational researchers also expect model parameters to be able to provide meaningful interpretations. Therefore, parameter plausibility is another important aspect for evaluating models. However, quantifying parameter plausibility or goodness is non-trivial due to the lack of gold standards. In our study, we used the two metrics we explored in [2].

For the first metric, we inspected the number of practice opportunities required to master each skill in the domain. We assume that skills in the curriculum are designed to neither be so easy to be mastered in three or fewer opportunities nor too hard as to take more than 50 opportunities. We define mastery as the same way as was done for the mastery learning criterion in the LISP tutor [9]: students have mastered a skill if their estimated knowledge is greater than 0.95. Based on students’ prior knowledge and learning parameters, we calculated the number of practice opportunities required until the predicted knowledge exceeds 0.95. Then, we compared the number of skills with unreliable values in both cases (fewer than 3 and more than 50).

As seen in Table 2, the results might not be consistent in the two conditions. Fixed priors results in more skills with too fast mastery rate, whereas the other three models produce 5-6 more skills mastered too quickly. It is worth pointing out that the skills found to be slowly mastered by the fixed model is a subset of those found by the other three models. Furthermore, the skills with low mastery rates found by the three Dirichlet models have high overlap.

One possibility is the skills really are learnt that slowly. For example, if the students lacked preparation, they are unlikely to learn just through an ITS. All of the skills that required more than 50 opportunities to master were from the same distribution in the 2-distribution model. That distribution with “unlearnable” skills has the parameter estimates of 0.5, 0.36, 0.22 and 0.08 for prior knowledge, guess, slip and learning, respectively. Compared to the learning rate of the other distribution, 0.36, the skills are captured as ones that students have difficulties to learn, thus the mastery rates are very slow. Interestingly, in the 3-distribution model, the “unlearnable” skills are from two

distributions. One has higher prior knowledge, 0.62, but lower learning rates, 0.07. The other has lower prior knowledge, 0.39, but normal learning rate, 0.11. We know that both cases could result in a slow mastery progress. Therefore, although the numbers seems to suggest those skills are poorly-estimated, if there really are skills students don't learn, the models are better at finding them due to clustering.

Table 2. Comparison of extreme number of practice until mastery

	# of skills with # of practices ≥ 50	# of skills with # of practices ≤ 3
Fixed	22	2
Single Dirichlet	28	0
2 Dirichlet distributions	27	0
3 Dirichlet distributions	27	0

We also tried to evaluate parameter values directly by calculating the correlation between a skill's estimated prior knowledge and the grade at which that skill was taught. We assumed that the earlier the students learned the skills, the higher their incoming knowledge would be. However, we found our data suffer a severe problem that most items require multiple skills to answer, especially skills learned in earlier grades. Consequently, it confounds the relationship between the estimated prior knowledge and the grade at which the corresponding skill was taught, thus this approach was not viable. Therefore, we still followed the technique in [2]: using external measurement to evaluate parameter plausibility. The students in our study had taken a 33-item algebra pre-test before using ASSISTment. Taking the pre-test as external measure of incoming knowledge, we calculated the correlation between the *students'* prior knowledge estimated by the models and their pretest scores. In order to acquire the student's K_0 parameter, we used KT to model the students instead of skills (see [2] for details).

Table 3 shows four interesting results. The first and the most important one is that more Dirichlet distributions generally result in higher plausibility (shown in the second row). The correlation values of 0.88 and above are significantly higher than the baseline value 0.83 from the fixed prior model with p-values < 0.05 . In the 7-distribution model, the value drops to 0.83. It suggests classifying students in a fine-grained level provides the models more confidence about the distributions where the data are from, thus taking the extra information specified by the Dirichlet priors, the models converge at more believable points. The second result is that we found the evidence of Dirichlet is hurt by outliers. As seen in the first column, the Dirichlet model produces lower correlation (0.80 vs. 0.83) compared to the fixed prior model. However, the Dirichlet model with trimming equals the fixed prior model, indicating the necessity of trimming for Dirichlets. However, the advantage from trimming decreases as the number of cluster increases, until eventually the untrimmed Dirichlet has better performance. Thus, the power from trimming is reduced as presumably the higher similarity of the students in a distribution reduced the problem of outliers.

The third result is the hypothesis that there is an interaction effect between using more distributions and using Dirichlets. To confirm that higher plausibility is not simply an

result of having additional distributions, we set each distribution’s mean values as the fixed priors to train the models (first row of Table 3). We see that fixed prior models performance is independent of the number of distributions (except for possible overfitting with 6 clusters). Thus, the improvement from multiple Dirichlet distributions is not an artifact of multiple distributions necessarily resulting in better performance. The fourth result is shown at the last row of Table 3 where we used cluster centers to represent the individual student’s prior knowledge. This approach achieves surprisingly high plausibility. With more distributions, it even outperforms the fixed prior models in spite of requiring less computation.

Table 3 Comparison of correlation between prior knowledge and pretest, by number of clusters

	1 cluster	2 clusters	3 clusters	4 clusters	5 clusters	6 clusters
Fixed	0.83	0.83	0.83	0.83	0.83	0.80
Dirichlet	0.80	0.83	0.85	0.86	0.88	0.91
Dirichlet (trimmed)	0.83	0.85	0.85	0.87	0.89	0.85
Cluster center	N/A	0.77	0.81	0.84	0.87	0.84

4 Contributions

This paper presents a new approach for strengthening the fundamental assumption of the usage of Dirichlet priors in order to improve the knowledge tracing model’s predictive accuracy and parameter plausibility. Although Dirichlets are a solution to the problem of parameter plausibility, the assumption that all skills are from a single distribution is troubling. Rather than modeling skills as a single homogenous group, we acknowledge that similar skills should be modeled similarly. We used clustering techniques to identify groups of similar skills, and then modeled those groups with their own, independent Dirichlet priors.

In spite of using multiple Dirichlet distributions, we failed to find any improvement in predictive accuracy, which is consistent with the results in our previous work of investigating a single Dirichlet distribution. However, we confirmed that using distribution centers to fit the data isn’t much worse than using the skill’s individual parameter estimates [6].

For parameter plausibility of modeling skills, it appears using Dirichlets does not produce a more believable mastery rate, even when using multiple distributions. It is worth pointing out that if there really are skills that students don’t learn, the Dirichlet approach is better at finding them. We also showed that using multiple Dirichlet distributions to model students results in high plausibility of the students’ knowledge parameters. With multiple Dirichlet distributions (6 clusters), the correlation between the model’s parameter estimates and the external standards reaches 0.9. We also showed that using the cluster centers, rather than individual student estimates, generates plausible results too, but with less computational work.

We found that Dirichlets are likely to be hurt by outliers, both with respect to predictive accuracy and parameter plausibility. For predictive accuracy, the models with trimming

perform comparable or even better than not using trimming. For the student knowledge parameter plausibility, trimming resulted in stronger results, except when six clusters were used. To understand this reversal requires additional experimentation.

Finally, our intuition that modeling a distribution as a single Dirichlet could be hurt since the “outliers” are the skills which are drawn from a different distribution has been partially supported by the results.

5 Future work and Conclusions

There are several unsolved problems related to this work. First of all, predictive accuracy is strongly desired in most student modeling applications. We have tried various approaches to improve accuracy in the knowledge tracing framework. However, we have found that there are no quick wins [2, 4, 5, 8]. We think perhaps only relying on the KT model with the basic structure might not be sophisticated enough to account for the substantial variability in student problem-solving efforts. One line of research is to consider integrating other useful information with KT, as it makes sense to be aware of other variables that might affect student performance such as question difficulty and student engagement. By accounting for other sources of variance, it enables us to better estimate the student’s knowledge and (hopefully) consequently have a higher predictive accuracy and estimate more plausible parameters.

Second, considering the existence of multiple distributions seems reasonable and using multiple Dirichlet distributions is found to be beneficial in improving parameter plausibility. Dirichlet priors work fine in parameter plausibility on the student models, but don’t have apparent benefit for skill models. It is an important task to understand how to overcome this issue, or even determine if it is a problem at all. At present, we lack the strong domain-driven parameter plausibility metric that was used in the initial work with Dirichlets for reading [5]. Determining better metrics for the domain of mathematics, or even better domain-independent metrics is a high priority. Human-generated Dirichlets might be a solution, as the single attempt [5] did result in more plausible parameters. Again, if we had more powerful parameter evaluation metrics we could better determine whether using human knowledge is a promising direction. It is interesting to see the outcomes from using other techniques to identify the distributions, such as latent Dirichlet allocation (LDA [10]), which is a generative model that allows sets of observations to be explained by unobserved groups. In this context, skills can be considered from several unobserved groups and each of them can be represented by a Dirichlet distribution. Thus, LDA is a promising technique rather than using clustering.

There is a limitation in this work. We took the benefit of looking at the test dataset for determining the number of clusters where the models result in the best performance. However, a better way that would be conducted in the future work is to use a tuning dataset besides the training and test datasets. This approach would enable us to tweak our models based on the models’ performances on the tuning data, and then validate our models on the test data.

This paper has explored the idea of integrating multiple Dirichlet distributions with the knowledge tracing model. In terms of predictive accuracy, we failed to find any improvement contributed by the proposed technique. This work provides some additional support that using the using cluster centers is a reasonable approach. We found that, with multiple Dirichlet distributions, student knowledge parameters achieved high plausibility, even when using cluster centers to represent student knowledge. We have also found Dirichlet priors could be hurt by outliers, and found that first trimming the data before Dirichlet parameter estimations usually gives better performance.

Acknowledgements

This research was made possible by the US Dept. of Education, Institute of Education Science, “Effective Mathematics Education Research” program grant #R305A070440, NSF CAREER award to Neil Heffernan, the Spencer Foundation, and a Weidenmeyer Fellowship from WPI.

References

- [1] Beck, J. E., Mostow, J. How who should practice: Using learning decomposition to evaluate the efficacy of different types of practice for different types of students. 9th International Conference on Intelligent Tutoring Systems, Montreal, 353-362.
- [2] Rai, D, Gong, Y, Beck, J. E. : Using Dirichlet priors to improve model parameter plausibility. Proceedings of the 2nd International Conference on Educational Data Mining, Cordoba, Spain, pp141-148.
- [3] Corbett, A. and J. Anderson, Knowledge tracing: Modeling the acquisition of procedural knowledge. User modeling and user-adapted interaction, 1995. 4: p. 253-278.
- [4] Gong, Y, Beck, J. E., Heffernan, N. T. : Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting. The 10th International Conference on Intelligent Tutoring Systems, Pittsburgh, 2010
- [5] Beck, J. E., Chang, K.-m.: Identifiability: A Fundamental Problem of Student Modeling. Proceedings of the 11th International Conference on User Modeling, Greece
- [6] Ritter, S., Harris, T. K., Nixon, T., Dickison, D., Murray, R. C., Towle, B.: Reducing the Knowledge Tracing Space. Proceedings of the 2nd International Conference on Educational Data Mining, 2009, Cordoba, Spain.
- [7] Kai-min Chang, Joseph Beck, Jack Mostow and Albert Corbett : A Bayes Net Toolkit for Student Modeling in Intelligent Tutoring Systems : Intelligent Tutoring Systems: Intelligent Tutoring Systems Volume 4053/2006
- [8] Pavlik, P. I., Cen, H., Koedinger, K.: Performance Factors Analysis - A New Alternative to Knowledge. Proceedings the 14th International Conference on Artificial Intelligence in Education, Brighton, UK, pp. 531-538
- [9] Corbett, A.T. Cognitive computer tutors: Solving the two-sigma problem. International Conference on User Modeling. 2001. p. 137-147.
- [10] Blei, D. M., Ng, A. Y., Jordan, M. I.. Latent Dirichlet allocation. Journal of Machine Learning Research. Vol. 3. Pp. 993-1022. 2003.

Examining Learner Control in a Structured Inquiry Cycle Using Process Mining

Larry Howard¹, Julie Johnson¹, and Carin Neitzel²

{larry.howard, julie.l.johnson, carin.neitzel}@vanderbilt.edu

¹Institute for Software Integrated Systems, Vanderbilt University

²Peabody College of Education and Human Development, Vanderbilt University

Abstract. High potential variation in prior knowledge, metacognitive skills, and motivation within learner populations can prompt design strategies that combine explicit structuring and scaffolding with increased learner control. We examine the use of such a strategy—a structured inquiry cycle—in a corpus of online modules (50) for adult informal learners using process mining. We apply process analysis techniques previously demonstrated by others to formative assessment data from the modules. We then use process modeling for mining module deliveries (N=5617) to investigate learner control within the inquiry cycle as a whole. Our experience suggests roles for these techniques beyond assessing conformity, both for design reflection and in preparation for deeper inquiry on self-regulation.

1 Introduction

Informal learning situations often exhibit high variability within the learner population, especially when learning experiences and environments offer broad availability, such as learning through the web. Varying school, work, and life experiences lead to differences in prior knowledge, learning skills, and learning styles among such learners. Disparate intrinsic and extrinsic motivations affect their engagement. Such variation presents interesting challenges for creating designs that are responsive to learners as individuals.

For over a decade, we have pursued design strategies that emphasize adaptive agency within computer-based environments for increasing responsiveness to individual learners. We have created and used a model-based design technology and adaptive learning platform [5] expressly for this purpose. But in designing a large corpus of online resources for adult informal learners (50 modules, each representing 4-8 contact hours) we chose to emphasize learner control as a primary strategy for individualization. Given research on learner control showing the positive *and* negative effects it can have on learning [8, 9, 14], we sought to strike a balance between the freedom of navigation afforded in cyberlearning environments and the need to scaffold learner experiences, particularly when prior domain knowledge or learning skills are weak.

This tension between self-direction and unambiguous instructional guidance is alleviated in environments where freedom of movement and explicit structuring can coexist within the same resource, such as in cyberlearning environments. To support learners who might otherwise make poor sequence or content choices within the environment, we present a well-formed structure—an inquiry cycle—in which to situate available learning activities.[11] Preserving navigational freedom, learners can follow a canonical path through the cycle and thus reduce cognitive load, or define their own unique pathway, exercising more control to create a more personalized learning experience. While the use

of such a “loose-tight” design strategy is informed by prior work, it presents many new issues warranting analysis and reflection.

Earlier analyses of observational data from modules designed in this way [7] suggested value in putting observations of learner behavior into a process context. Two process mining directions were pursued: a process discovery approach using hidden Markov models [8] and a process analysis approach that we address here. In [12], the authors described process analysis techniques and applied them to online assessment data. Inspired by their work, we began by replicating the application of these techniques to similar formative assessment data from our modules. We then enlarged the scope of this application to examine issues of learner control within the modules as a whole.

We begin the paper with a description of the learning cycle employed by the modules and its influences from prior work. In Section 3, we describe the formative assessment aspects of the modules and present a process analysis of associated data. Section 4 presents and discusses the use of process mining to investigate issues of learner control for the modules overall. We conclude by briefly reflecting on our experience and highlighting directions for future work.

2 Anchored Inquiry and STAR Legacy

The inquiry cycle used as an instructional design pattern for the modules is an instance of the Software Technology for Assessment and Reflection (STAR) Legacy Cycle [13] that was developed in the course of work by the Cognition and Technology Group at Vanderbilt (CTGV) on anchored instruction [1] and situated cognition. STAR Legacy organizes a student’s inquiry of a posed challenge around a set of activities. Among its central ideas is providing a structure that is both well-formed, including kinds of activities known to be beneficial to learner inquiry, and explicit, so that learners know where they are in the cycle, the intention of its activities, and therefore what it means to select and use them. STAR Legacy has been employed in many different educational settings, including a large corpus of classroom and blended instruction for undergraduate bioengineering education [2] and online continuing education for teachers [3].



Figure 1: STAR Legacy Inquiry Cycle

STAR Legacy arose from interest in “flexibly adaptive” instructional designs that are informed by research on effective learning experiences and easily tailored by educators to characteristics of particular learning situations. The original learning cycle consisted of six activities: The Challenge, Generate Ideas, Multiple Perspectives, Research and Revise, Test Your Mettle and Go Public.[13] In applying it to informal, asynchronous learning, the role of the original Multiple Perspectives activity was subsumed by Initial Thoughts, which guides the student’s initial exploration of the challenge in ways echoing

Multiple Perspectives. Also, the original Go Public activity, used to scaffold synthesis, was replaced by a Wrap Up activity, where learners reflect on their initial thoughts in the presence of expert views and are provided an opportunity to apply what they've learned to a related situation. These adaptations reflect serious compromises, yet they allowed us to introduce extensive learner control while preserving essential qualities of an explicit learning cycle and inquiry scaffolding afforded by its activities.

3 Formative Assessments

The *Assessment* activity in our adaptation of STAR Legacy provides a collection of questions that allow learners to confirm their understanding of learning materials presented in the cycle's *Resources* activity. Learners select questions from a menu organized around categories based on the module's terminal learning objectives. Each question's text is presented in the menu to facilitate selection. Questions can be used whenever, and as often as, learners choose.

The self-assessment questions support multiple attempts with feedback as shown in Figure 2. The feedback after an initial incorrect attempt (L1F) concerns clarifying the question by restatement. Following a second incorrect attempt, criticism of the learner's response is offered (L2F). This feedback takes the general form: "If X was true, as your answer indicates, then Y", where Y is some negative consequence. After the third and subsequent attempts, critiquing feedback is combined with a link back to related learning materials provided in the *Resources* activity (L3F).

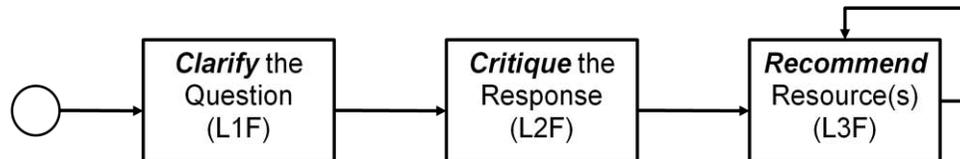


Figure 2: Feedback Progression in Module Self-Assessments

When a question is accessed, the feedback-giving process continues until either (1) the learner provides the correct response, at which time she is automatically returned to the question menu, or (2) the learner abandons the question, either by interface-supported navigation to another cycle activity or by selecting a resource link offered in the L3F. Successive accesses of a question restart the feedback process. On returning to the menu, an indication of success (or abandonment) for the most recent access is presented next to the question. The menu thus provides learners an "at a glance" view of their use of questions in the activity and the results.

3.1 Process Modeling

We began our process analysis of formative assessment data by constructing a Petri Net model of a question access, shown in Figure 3 below, which commences when a learner selects a question from the question menu. This model details the feedback process described above for consecutive attempts made during the access. Places in the Petri Net represent presentations of the question. Outbound transitions represent learner actions.

The CR and WR transitions represent the learner providing the correct or wrong response, respectively. WR transitions are labeled with the level of feedback incorporated into the subsequent question presentation. Two forms of question abandonment are modeled. The first (AR) occurs when the learner backtracks to a learning resource in the *Resources* activity immediately following question abandonment. The second (unlabelled) represents abandonment through other navigation supported by the interface without resource backtracking, such as continuing to a different question or transitioning to another activity.

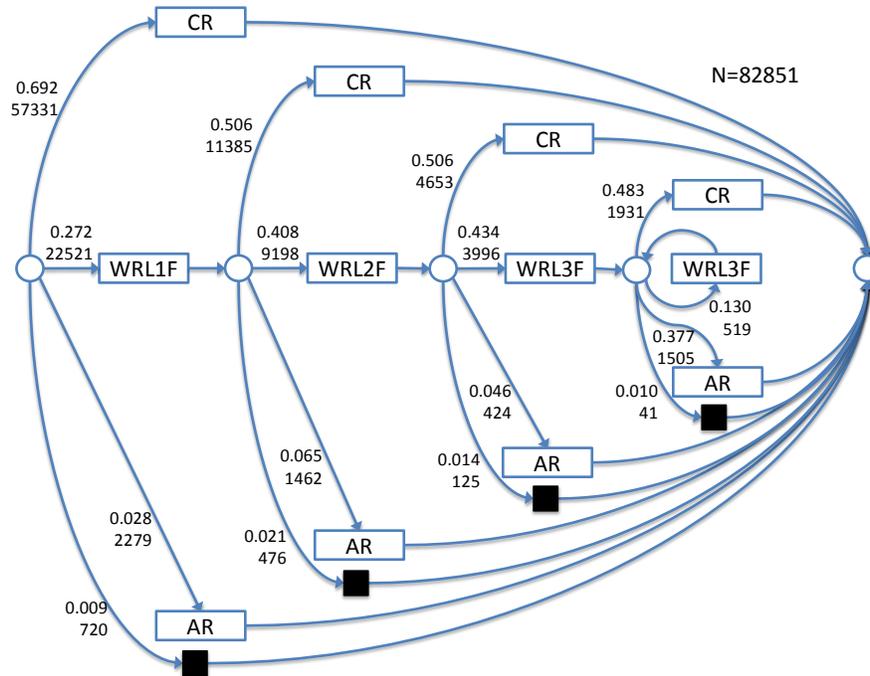


Figure 3: Petri Net Model of Accessing a Self-Assessment Question

The addition of the AR transitions represents modeling not solely for the purposes of fidelity, but also to address particular analysis interests. In this case, we wanted to examine prompted versus discretionary resource backtracking during a question access. The feedback provided on the third question attempt (L3F) and beyond includes remedial resource recommendations, so abandonment in this setting is prompted. In all earlier contexts, abandonment with resource backtracking is an unprompted discretionary move.

3.2 Discussion

The mining results, overlaid on the process model in Figure 3 above, detail 82,851 question accesses. 87% (72218) of these were first time accesses and the correct response was given 93% of the time over the sequence of attempts. Nearly 70% of the correct responses were given on the first attempt. The remaining question accesses were a combination of repetition following prior success (5%), suggesting review, and repetition following prior abandonment (8%).

One subject of our analysis was efficacy of the remediation scheme. The provided questions include a combination of multiple choice (MC) and fill-in-the-blank (FIB) types, skewed heavily towards the former for ease in constructing the critiquing (L2F) feedback. While the capability exists to provide response-specific FIB feedback, planned data mining and feedback preparation exceeded project constraints, so only non-specific feedback is given. Also different between the two question types is the falsification of alternatives that naturally occurs with MC questions over a series of attempts. We were interested in how these differences affected learner response to feedback.

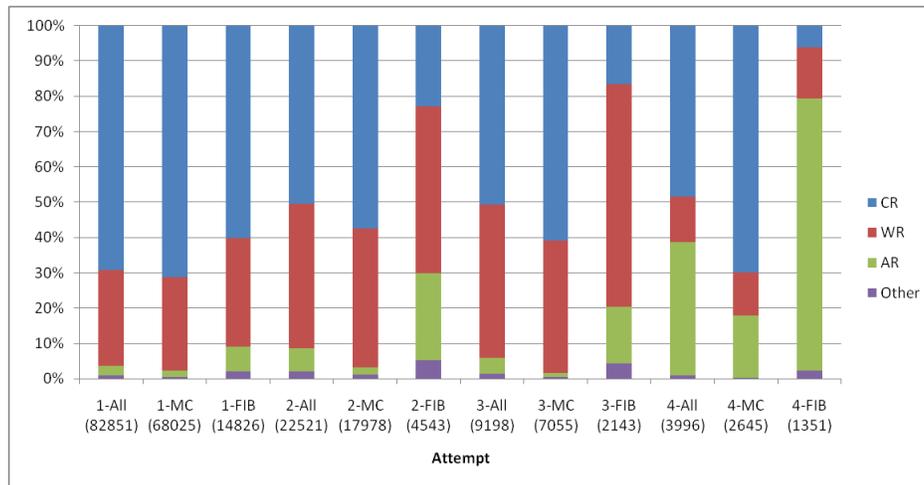


Figure 4: Question Accesses By Attempt Detailing Question Types

Figure 4 above provides a summary of results by attempt, both collectively and for each question type, which varied significantly. We will therefore present our discussion of the results by question type.

Overall, 95% of MC questions were correctly answered over the series of attempts. For 2nd MC attempts, with one choice falsified and the initial feedback, only 58% of learners provided the correct response, suggesting low efficacy for the question clarification feedback. With two choices falsified and specific criticism of the second response, correct responses on 3rd attempts improved marginally to 61%, a disappointing result for the much more labor-intensive response-specific feedback. On 4th attempts (where most questions provided just four choices) correct responses improved to just 70%. One possible explanation for results on later attempts (3+) is that a lack of penalty for incorrect responses, combined with benefits from specific formative feedback, lead some learners to explore, effectively using the questions as supplemental learning resources.

Only 70% of FIB questions were answered correctly over the sequence of attempts. Performance actually degraded, rather than improved, from 60% correct responses on the initial attempt to 23%, 17%, and 6% on subsequent attempts, respectively. Even the initial performance differed significantly from FIB questions on high-stakes assessments, where it was comparable to MC questions. On the formative assessments, the lack of penalty for attempts likely contributed to guessing or gaming to obtain more feedback, and non-specific feedback provided insufficient prompting.

Another analysis focus was to understand abandonment behavior. Figure 5 (at right) shows resource backtracking and other abandonment for each question type by attempt. Discretionary abandonment (attempts 1 thru 3) was clearly more pronounced for FIB questions, likely owing to weakness in feedback specificity and inability to proceed by falsifying choices, as with MC questions. Resource backtracking following abandonment predominated for both question types. While incidence of unprompted abandonment was low as an overall percentage, when viewed as a percentage of learners not making a correct response, between 1 and 2 in 10 made conscious decisions to remediate rather than (continue to) guess. For 4th attempts, abandonment without resource backtracking was practically non-existent, suggesting that the specific resource recommendations prompting abandonment were largely taken up by learners.

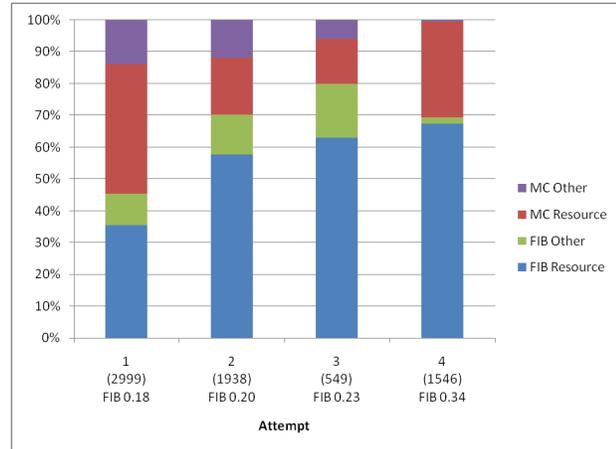


Figure 5: Question Abandonment By Attempt

An area for future work is to examine the effectiveness of self-remediation when backtracking to resources. An initial look showed that, on 88% of questions where learners backtracked to a resource following abandonment, a correct response was given on the 1st attempt when returning to the question. Comparative question evaluation, as in [10], might also help clarify performance differences between FIB and MC questions on self-assessments and on FIB questions between formative and summative assessments.

4 Examining Learner Control in STAR Legacy Modules

With this initial process mining, we turned our attention to issues of learner control within the modules as a whole. An initial area of interest concerned identifying the extent and nature of control that could be viewed as discretionary, as with unprompted question abandonment discussed earlier, versus ordinary forms of control, such as advancing to the next element in a sequence.

Discretionary navigation controls in the modules typically serve the dual purposes of affording action and informing status, as with the menu for self-assessment questions in the *Assessment* activity described earlier. The controls are not adaptive in the sense of controlling action through presentation, as typically found in adaptive hypermedia. A fixed navigation sidebar interface element continuously informs learners of where they are in the learning cycle and supports arbitrary transitions to any cycle activity at any time. Learners can decide for themselves when and how often to use the activities comprising the cycle. There is thus admitted a wide range of possible behavior. Such affordances for discretionary control are paired with traditional controls for advancing and backtracking, such that learners are not forced to choose direction, as a means of decreasing cognitive load [15].

4.1 Process Modeling

As before, we began by constructing a Petri Net model of navigation pathways within the inquiry cycle, shown in Figure 6 below. Places in the model represent the cycle activities and transitions are learner accesses of an activity. Transitions are labeled by the activity destination: “The Challenge” (C), “Initial Thoughts” (T), “Resources” (R), “Self-Assessment” (A) and “Wrap Up” (W). Unlabelled transitions represent leaving the module. Learners can freely move between a set of modules constituting a course, returning in-place to previously visited modules. Such “suspend-resume” decisions are not modeled here, so the exiting transitions represent final departures.

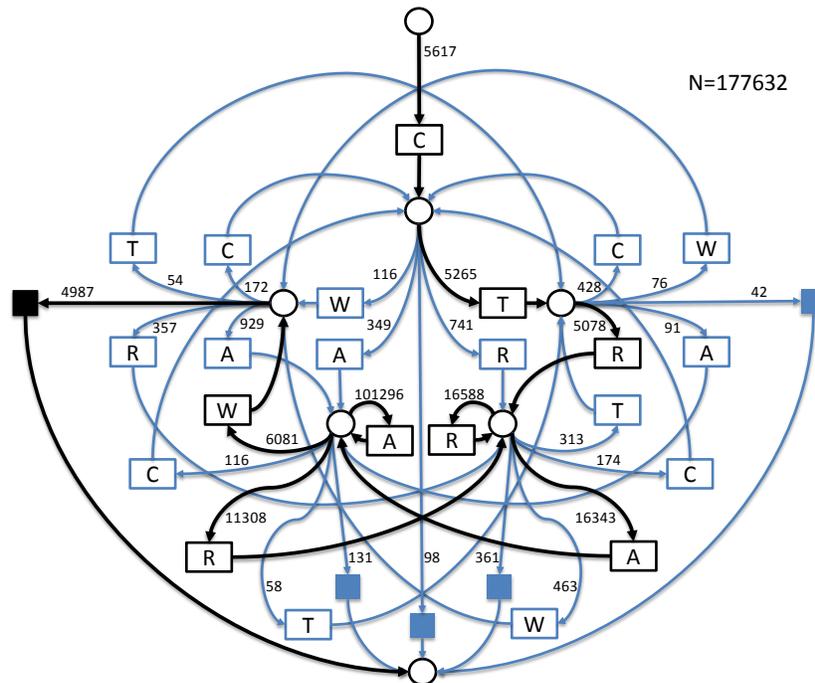


Figure 6: Petri Net Model of Learner Navigation in STAR Legacy

Annotating the transitions in this model with incidence data (177,632 observations from 5,617 individual module deliveries) showed that every possible navigation behavior was observed. The benefit of this exercise was that it yielded an “at a glance” view of learner control, allowing us to quickly compare relative transition incidences. Yet, to examine discretionary versus sequential control, we needed a more refined model. We began by marking the sequential pathway through STAR Legacy in Figure 6 (darkened edges.) From this we derived another process model, shown in Figure 7 below, to investigate deviation from sequential navigation. As before, places represent cycle activities. From each place is a transition (unlabelled) that represents deviation from the sequential path. For the *Resources* and *Assessment* activities, a cyclic transition was added (dashed edges) to differentiate just entering the activity’s menu from use of its contained elements (resources or questions) prior to departure. Another incidental transition was included to identify any occurrence of backtracking by the learner from *Assessment* to *Resources*.

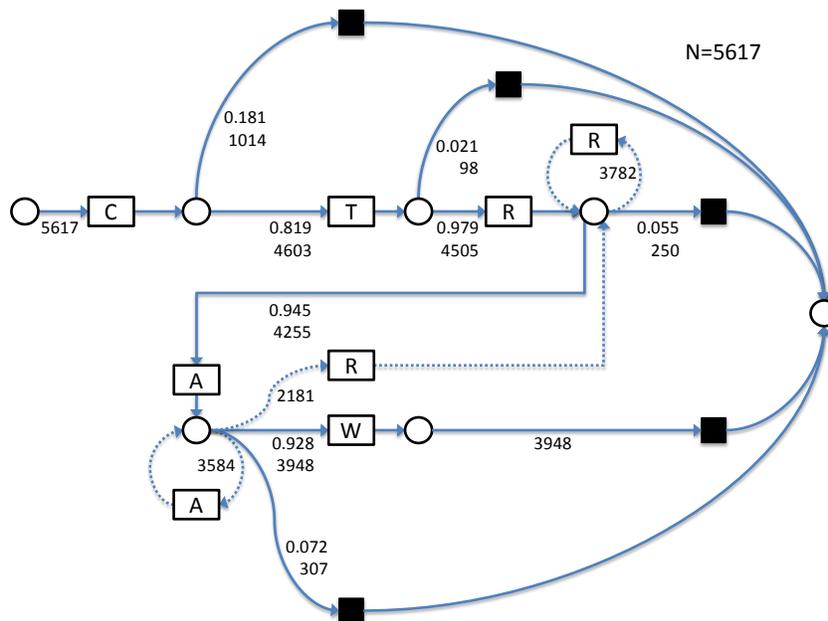


Figure 7: Petri Net Model of Linear Progression in STAR Legacy Cycle

4.2 Discussion

Results from data mining using the process model in Figure 7 are shown in annotations of the process model and summarized in Figure 8 below. The latter indicates cycle activities after which the learner deviated from the sequential process, both overall and for module progressions to examine changes over time. Overall, in 70% of the 5,617 modules examined learners stayed within the sequential pathway throughout the learning cycle. Only 51% of learners reaching the *Assessment* activity sequentially performed any resource backtracking. Curiously, 16% of learners reaching the *Resources* and *Assessment* activities linearly accessed no resource or question within the activity prior to transitioning or backtracking: a subject for future investigation.

The overall extent of sequential navigation agrees in some respects with earlier analyses performed using coded transition data, but process analysis clarified moments of deviation for further reflection. By-passing the *Thoughts* activity (that is, deviation immediately following the *Challenge*) was the most significant linear process deviation. This activity is intended to help learners consider what will be involved in addressing the challenge to highlight what they may already know and will need to learn in the course of their inquiry. In terms of self-regulation, it relates to the metacognitive task of strategic planning.[16] Typically an instructor-led discussion in classroom uses of STAR Legacy, with obvious social affordances, the online modules present open-ended questions that prompt learners to capture their initial thoughts. In feedback from course evaluations, some learners explicitly noted discomfort with this activity, regarding it as some form of evaluation prior to learning, even given guidance for the activity to the contrary.

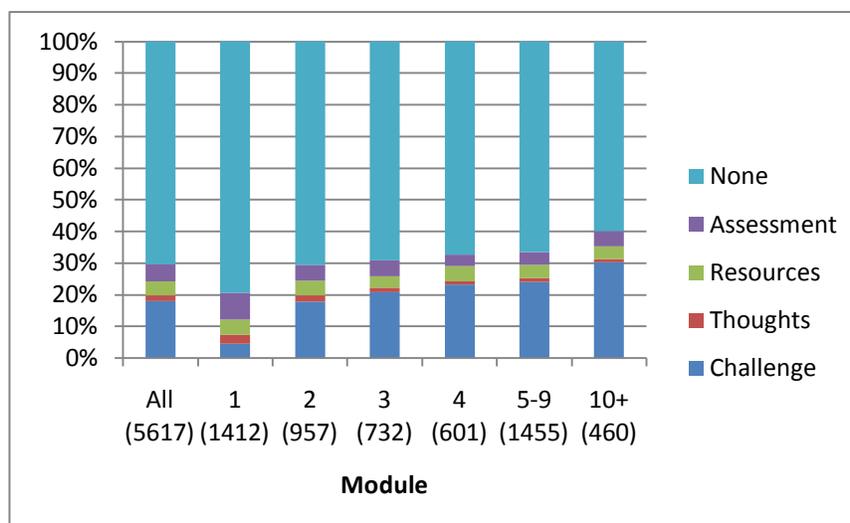


Figure 8: Changes in Deviation from Linear Pathway between Modules

One implication of greater learner control is that it allows avoidance of design elements intended to create “cognitive dissonance” [4]. As shown in Figure 8, over consecutive modules the avoidance of the *Thoughts* activity increases to the point where it is three quarters of the total deviation, with the most significant increase immediately following the first module. Examination of summative assessment performance showed weak correlation between participation in the *Thoughts* activity and increased learning outcomes. These preliminary findings indicate weakness in our design of the activity that warrants further investigation to inform potential redesign efforts.

5 Conclusions and Future Work

Analysis approaches to process mining use models as *a priori* representations. In the presented analyses of formative assessments and learner control within a structured inquiry cycle, we incorporated elements into process models to disambiguate forms of control or address other control-related interests. These examples demonstrate use of process analysis not just to address conformity, but for other inquiry that involves putting data into ordered contexts. Petri Net process models, being both formal and visual, aided collaborative planning for such analyses and in reviewing results, where unshared, informal understandings can contribute hindrances. As discussed in [12], technology to support mining directly from process models would have been beneficial, and we are tracking progress towards this goal in frameworks such as ProM [14].

Whether using discovery or analysis approaches, the potential to assign meaning to observed behavior using process mining is limited. In an investigation of self-regulated learning, we plan to use more traditional, expensive, and invasive methods for attributing learner behavior in following-up our preliminary analyses. Understanding gained with such methods will be used to enhance the value of passively collected instrumentation as a primary data source. We hope this approach will enable richer accounts of learner behavior in environments for informal learning that afford both structure and freedom in addressing broad and diverse populations.

Acknowledgements

This work was supported in part by Cooperative Agreement Number 2006-GT-T6-K009 administered by the U. S. Federal Emergency Management Agency in collaboration with the Univ. of Memphis Center for Information Assurance and Cobham Analytic Solutions.

References

- [1] Bransford, J.D., Sherwood, R.D., Hasselbring, T.S., Kinzer, C.K. & Williams, S.M. Anchored instruction: Why we need it and how technology can help. *Cognition, education, and multimedia: Exploring ideas in high technology*, p. 115-141, 1990.
- [2] Cordray, D., Harris, T. & Klein, S. Research synthesis of the effectiveness, replicability, and generality of the VaNTH challenge-based instructional modules in bioengineering. *Journal of Engineering Education* 98(4), p. 335 - 348, 2009.
- [3] Smith, D.D., Pion, G., Skow, K., Tyler, N.C., Yzquierdo, Z.W., Givner, C., & Brown, J. The IRIS Center for Faculty Enhancement: On-line course enhancement modules and materials for use in the preparation of education professionals. *New Horizons for Learning*, 11, p. 1-17, 2005.
- [4] Elliot, A.J. & Devine, P.G. On the motivational nature of cognitive dissonance: Dissonance as psychological discomfort. *Journal of Personality and Social Psychology* 67, p. 382-382, 1994.
- [5] Howard, L. Adaptive learning technologies for bioengineering education. *IEEE Engineering in Medicine and Biology Magazine* 22(4), p. 58-65, 2003.
- [6] Howard, L., Johnson, J. & Neitzel, C. Reflecting on online learning designs using observed behavior. Accepted to *The 15th Int'l Conference on Innovation and Technology in Computer Science Education*, 2010.
- [7] Joeng, H., Biswas, G., Johnson, J. & Howard, L. Analysis of productive learning behaviors in a structured inquiry cycle using hidden Markov models. Accepted to *The 3rd Int. Conf. on Educational Data Mining* (EDM 2010).
- [8] Karagiorgi, Y. & Symeou, L. Translating constructivism into instructional design: Potential and limitations. *Educational Technology & Society* 8(1), p. 17-27, 2005.
- [9] Katz, I. & Assor, A. When choice motivates and when it does not. *Educational Psychology Review* 19(4), 429-442, p. 2007.
- [10] Kuechler, W.L. & Simkin, M.G. Why is performance on multiple-choice tests and constructed-response tests not more closely related? Theory and empirical test. *Decision Sciences Journal of Innovative Education* 8(1), p. 55 - 73, 2010.
- [11] Merrill, M.D. First principles of instruction. *Educational technology research and development* 50(3), p. 43-59, 2002.
- [12] Pechenizkiy, M., Treka, N., Vasilyeva, E., van der Aalst, W. & De Bra, P. Process Mining Online Assessment Data. In: *The 2nd Int. Conf. on Educational Data Mining* (EDM 2009), p. 279-288, 2009.
- [13] Schwartz, D., Lin, X., Brophy, S.P. & Bransford, J.D. Toward the development of flexibly adaptive instructional designs. In: *C. Reigeluth, Ed. Instructional-design Theories and Models: A New Paradigm of Instructional Theory II*, p. 183 - 214, 1999.
- [14] van Dongen, B.F., Alves de Medeiros, A.K., Verbeek H.M.W., Weijters, A.J.M.M., van der Aalst, W.M.P. The ProM framework: A New Era in Process Mining Tool Support. In G. Ciardo and P. Darondeau, editors, *Application and Theory of Petri Nets 2005*, volume 3536 of *Lecture Notes in Computer Science*, pages 444-454. Springer-Verlag, Berlin, 2005.
- [15] van Merriënboer, J.J.G. & Sweller, J. Cognitive load theory and complex learning: Recent developments and future directions. *Educational Psychology Review* 17(2), p. 147-177, 2005.
- [16] Zimmerman, B.J. Becoming a self-regulated learner: An overview. *Theory into Practice* 41(2), p. 64 - 70, 2002.

Analysis of Productive Learning Behaviors in a Structured Inquiry Cycle Using Hidden Markov Models

Hogyeong Jeong, Gautam Biswas, Julie Johnson, and Larry Howard
{hogyeong.jeong, gautam.biswas, julie.l.johnson, larry.howard}@vanderbilt.edu
Institute for Software Integrated Systems, Vanderbilt University

Abstract. This paper demonstrates the generality of the hidden Markov model approach for exploratory sequence analysis by applying the methodology to study students' learning behaviors in a new domain, i.e., an asynchronous, online environment that promotes an explicit inquiry cycle while permitting a great deal of learner control. Our analysis demonstrates that the high-performing students have more linear learning behaviors, and that their behaviors remain consistent across different study modules. We also compare our approach to a process mining approach, and suggest how they may complement one another.

1 Introduction

One of the broader goals for Intelligent Learning Environments (ILEs) is to help students not only become proficient in particular topics, but also to become better prepared for future learning (PFL) [3]. Among others, two PFL considerations stand out: (1) students should learn the content with good understanding so that they may apply it in problem solving situations, and (2) students should develop good learning behaviors and strategies that they can apply in other learning situations, even those outside the computer learning environment. Computer tutoring systems typically impose learning or problem solving structure on the learner [1]. They are primarily designed to monitor students' performance on the task, and provide feedback to help students improve their learning performance. In contrast, students have choice in their learning activities in inquiry-based and exploratory learning environments. In such systems, it may be quite useful not only to monitor students' learning performance, but also track and interpret their activities, map them into good and bad learning behaviors, and then provide appropriate feedback to help the students become better learners. If done well, this would also help students become better prepared for future learning. In this paper, we employ data mining methods to analyze students' activity sequences and map them on to potential learning behaviors.

We have used hidden Markov models (HMMs) [6, 7, 10] as an exploratory tool to analyze traces of students' activities as they learn by teaching a computer agent called Betty. Exploratory data analysis makes few initial assumptions about the domain, but its results, even if they are preliminary, informs confirmatory data analysis. The system, Betty's Brain [2], is also designed to help students become aware of and learn self-regulated learning (SRL) skills through interactions with the teachable agent, Betty, and a mentor agent, Mr. Davis. The HMMs provide aggregated descriptions of the more prevalent patterns of students learning behaviors (both good and bad). A challenging task, after learning a HMM from the trace data, is the interpretation of its "hidden" states. In our work, we have interpreted the states in terms of the set of activities linked to the

states, and then characterized activities associated with states as learning behaviors. However, how best to assign meaning to the states in HMM analyses is application dependent.

In this paper, we demonstrate the generality of our HMM approach by applying it to analyze student learning behaviors in another domain, an asynchronous cyber learning environment in cyber terrorism. Unlike the middle school subjects in the Betty's Brain studies, the users of this system are adult professionals, who use this course as a degree requirement or for professional certification. A few use the system for self-study. The system had a wide variety of users (over 6,000) but we do not have access to the specifics of their background information. Therefore, we decided to focus our exploratory data analysis methods on examining the differences in behaviors between the high and the low performers on the system, where the categorization of high versus low was determined from the posttest scores. We believe that this analysis will provide insights on how learning behaviors relate to performance. As further discussion, we will compare our HMM analyses with the results of a model-driven analyses conducted by our colleagues at ISIS who developed this system [5]. In particular, we note how the process modeling techniques can complement our analysis by providing a finer-grained perspective on students' behaviors.

2 System Description

The asynchronous cyber learning environment on cyber-terrorism is derived from the STAR Legacy Cycle, a software shell designed to organize learning activities as an inquiry cycle [9]. A key feature of this learning model is that it makes the steps in the learning cycle explicit, making it easier for learners to understand the intent of the activities, and how they apply to learning. Some adaptations were made to the original STAR Legacy cycle to tailor it for asynchronous, online learning of a number of cyber-terrorism modules. The adaptation preserves the essential qualities of the explicit inquiry cycle: adaptive and progressive feedback, and scaffolds for planning, reflection, and synthesis. The *Overview* (O) phase is the portal that students use to enter and exit the cycle. The modified cycle is represented in five phases (see Fig. 1): (1) *Challenge* (C), where the students are presented with the problem description; (2) *Initial Thoughts* (T), where the students provide their initial thoughts on the problem; (3) *Resources* (R), where the students can learn about the problem; (4) *Self-Assessment* (A), where the students answer assessment questions to get formative feedback on their progress; and finally (5) *Wrap-up* (W), where the students can review their initial thoughts and conclude the module.

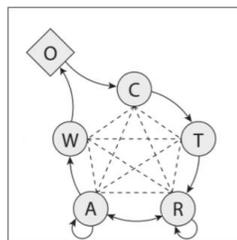


Figure 1. The Adapted STAR Legacy Cycle

While the canonical inquiry cycle supported by the STAR Legacy interface has students progressing in a linear fashion from *Challenge* to *Wrap-up*, the system is also designed to promote the idea of learner control [4, 11]. In other words, students are given complete freedom to navigate through the cycle to accomplish their learning tasks. We should note that it is precisely this freedom that allows for various behavior patterns to emerge as the students use the system; this makes the activity sequences collected from the system particularly amenable to our exploratory data analysis methods.

3 Experimental Study

The users of the system are primarily adults who take the course to meet certification requirements or for recommended workplace training. A few students took the course for other reasons, but this number was insignificant. In all, the preliminary dataset which we retrieved consists of 6,298 learners covering 50 learning modules in 10 independent-study courses.

Each module represents roughly 4 to 8 hours of self-study, and each independent-study course contains 3 to 7 different learning modules. The students could pass the pretest for a module and avoid the training. Otherwise they went through the training and took the summative assessment at the end (outside of the training system). Both the pretest and the summative assessment questions were randomly generated subset from a shared pool of questions. It is important to note that learners who successfully completed every section (pretest or summative assessment) were immediately awarded the course certification. In fact, around 18% of the students successfully prequalified during the pretest, and skipped the training modules. Hence, the students who used the system were necessarily those who did not have the requisite prior knowledge to pass the test the first time.

In our previous analyses, we created groups based on how the students improved from pre to posttest by using the system, and made distinction between a low (pretest performance) and low (posttest performance) group, low and high group, and high and high group [2]. However, in this study, as discussed above, all students who use the system universally belong in the “low prior knowledge” category. Therefore, our grouping uses only the posttest scores. For this study, we split the scores into three uniform groups (after determining that the data was unimodal) to define our low, middle and high performance groups. At the two ends of the middle group, there were a number of scores that were close to the low and high group scores respectively, so we ignore the middle group, and focus our analysis on the differences between the low and the high groups.

Using these performance groupings, we examine ways to characterize the behaviors of the different students, and determine, for example, whether the high-performing students have learning behaviors that are distinct from the low performers. In our previous domain, we have found that the higher-performing students exhibit patterns of behaviors that correspond to good learning strategies (e.g., explore one’s understanding of a topic by asking relevant follow-up queries and checking to see if the explanations for the answer are meaningful) [7]. For our current domain, cyber security, we direct our attention to studying how students transition between the different phases of the STAR

Legacy cycle. In particular, we are interested in determining whether the path students take when following the cycle is in any way related to the students' performance. For example, one may argue that the students who follow the cycle in a linear fashion (the implicit model of inquiry learning in STAR Legacy) would perform better in their learning tasks than students who make numerous jumps between the phases. An alternate hypothesis may state that students who make multiple forward and backward jumps in the cycle do so because they have formulated their own strategy for learning the content material, and hence would perform better.

Our behavior analysis will focus on the transitions students make as they go through the cycle and their context. Howard, et al. [9] have defined six different types of transitions: *linear* (L), *jumping* (J), *retrying* (R), *searching* (S), *transitioning* (T), and *backtracking* (B). *Linear* transitions comprise directed navigation steps in the cycle, and may be performed by clicking the "Next Up" navigation button appearing at the end of each activity or resource. *Jumping* transitions imply the use of Resources and Assessments in a non-linear way through the use of descriptive menus. *Retrying* transitions occur in the Assessment phase, and represent the activity of re-attempting a specific question. *Searching* transitions comprise of actions, where the students search the materials in either the Resources or the Assessments phase. *Transitioning* represents the students moving to and from the course module menu. Finally, *Backtracking* transitions indicate that the student has previously seen the destination activity, resource, or menu.

We hypothesize that students in the different performance groups will employ different learning behavior patterns (and likely, different strategies) when using the system. Further, students' behavior patterns will evolve as they study different modules. Therefore, hypothesis 1 states that there will be a marked difference between the two performance groups, especially in their use of linear transition behaviors as they learn using the STAR Legacy cycle. Hypothesis 2 states that the students' behaviors will evolve as they use the system, and that this can be seen when comparing early and late module behaviors. We will use two metrics to assess the difference between the models: (1) the definition and interpretation of the HMM states; and (2) the stationary probabilities, which represent the proportion of time spent in a state relative to the other states of the HMM. We will describe these metrics in more detail in the following section.

4 Methods

Before the students' trace data can be analyzed to generate HMMs, necessary preprocessing steps have to be executed on the raw activity sequences. As stated before, the focus of our analysis is to interpret students' behaviors as states and the transitions between states. Thus, the objective of the preprocessing is to extract and reduce the activity sequences to just the transition and the context associated with the transition. Because clear distinctions were made between the different types of transitions, all of the preprocessing can be done automatically. At the end of the preprocessing phase, we have activity sequences consisting of series of transition-context pairs (e.g. "AR-Linear; RR-Linear; RR-Retrying", where A represents the *Assessment* phase, and R represents the *Resources* phase).

4.1 *The HMM Procedure*

Deriving a HMM from students' activity sequences should result in a model that identifies frequently occurring sequence patterns, which are then interpreted as student behaviors. The HMMs are so named because the derived states are hidden, i.e., they cannot be directly observed. Instead, one provides meaning or interpretation to the states by studying the activities associated with the state. These activities imply behaviors, and the collection of states may imply one or more behavior patterns.

A first step in model generation is to initialize the parameters that define the states of the HMM and the possible state transitions. Starting from an initial model description, expectation-maximization techniques are applied iteratively until the model parameters converge [6, 10]. The estimation process is quite sensitive to the initial state description, and bad initializations may lead to generating suboptimal models. In past work [6, 8], we have used a conceptual clustering algorithm to generate good parameters for the initial state.

Another unknown in the model derivation process is the “best” number of states that define the HMM. The metric that we use to assess the model fit is the Bayesian Information Criterion (BIC) measure, which takes into account both the log likelihood of the model (i.e., how likely the model is given the data) and the number of states in the derived model (i.e., how complex the model is) to find the model that strikes the best balance between high likelihood and low complexity [8]. The full procedure for deriving HMMs from sequence data is outlined in our previous EDM paper [6].

The HMMs offer us a state-based aggregated interpretation of the students' activity sequences, and we can analyze them in various ways. A first step is to assign meanings or labels to the states of the model. For example, we may find that some types of activities tend to cohere in certain states, which would provide clues to determine what behaviors these states represent. Also, we can examine the stationary probabilities to get a sense of what the more relevant (i.e., more frequently occurring) states are. The stationary probability, as we define here, is the relative proportion of activities that belongs to a certain state (i.e., the stationary probability of a state A is the proportion of occurrences of State A among all states that occur in a sequence of length n iterations generated by the model; n is typically the average number of activities in the input sequences). For example, a state having a 20% stationary probability implies that 20% of students' activities during the session are related to the behavior(s) that state represents. We can also study the transitions between the states to see how students transition between different behaviors. In addition, the structures of the models themselves can be studied to see if certain patterns appear (e.g., cycles), and these patterns can then be interpreted in terms of students' learning behaviors.

4.2 *The Experimental Datasets*

The HMMs were run using two different datasets. For the first dataset, we investigated the first module that the students worked on. The second dataset was a module that the

students worked on toward the end of their study for that course. Table 1 lists the number of students in the early and late modules, and the average posttest scores for these modules.

Table 1. Modules Used

	Module Type	Numbers (low, high)	Average Scores (low, high)
First Dataset	Early	(189,83)	(51, 95)
Second Dataset	Late	(133,97)	(34, 93)

5 Results

Examples of the HMM structures generated are shown in Figs. 2 and 3. All four models (Low-Early; Late-Early; High-Early; and High-Late) had 5 or 6 states. The main difference among the models was more in the activities associated with each state, rather than the state transition behavior. We discuss one of the derived models in some detail to familiarize the reader with the HMM structure. Fig. 2 depicts the model for the high performance group in the early module. Table 2 complements the model by listing the major composition of activities for each state. The first two letters denote the phases involved in the transition (e.g. “AR” is the transition from *Assessments* to *Resources*), and the letter after the hyphen indicates the transition type (e.g. “L” is linear).

We see that the students begin from state 1, and moving from Overview to Thoughts is the only activity associated with this state. Fig. 1 confirms that this corresponds to the initial part of the STAR Legacy cycle, so we label this state as the *Start State*. From state 1, the students move directly to state 2, where they proceed to thoughts (82% of the time) or jump to the resources (14% of the time) (see Table 2). Since this corresponds to the start of the problem solving, we call this the *Initiation State*. Then, the students move to state 3, where the students perform multiple activities, but the two primary ones are proceeding linearly through the resources (56%) and retrying questions in the resources (13%). The dominant activities plus the others lead us to label this state as the *Assessment State*.

Other analyses have determined that students spend significant portion of the time working on assessments (can be thought of as self-assessments), so not surprisingly our model indicates a self-loop with high likelihood (90%) for this state. When students exit the *Assessment State*, the HMM indicates moves to state 4 (likelihood of 7%) or to state 5 (likelihood of 3%). State 4 is composed mainly of linear transitions involving the wrap-up phase (47%) and retrying assessments (35%). For this reason, we call this the *Wrap-up State*. States 5 and 6 are composed mainly of backtracking transitions. In particular, we note that state 5 contains a significant portion of backtracking through resources and assessments (44%). Hence, we call state 5 a *RA (Resources-Assessment) Backtracking State*. Meanwhile, state 6 is composed significantly of activities involving wrap-up, overview, or thoughts (48%). Hence, we call state 6 a *HL (High-Level) Backtracking State*.

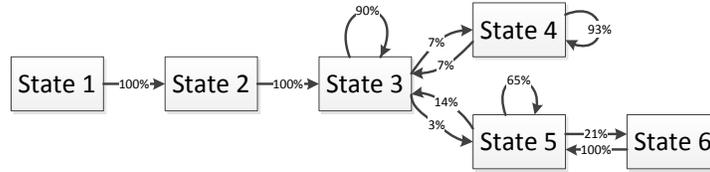


Figure 2. Early Module – High Performance Group

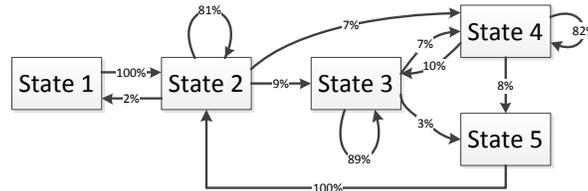


Figure 3. Early Module – Low Performance Group

Table 2. Distribution of Major Activities (those with probabilities greater than 5%)

	State 1	State 2	State 3	State 4	State 5	State 6
Early-High	OC-L (100%)	CT-L (82%)	AA-L (56%)	AA-R (35%)	RA-B (24%)	WO-B (26%)
		CR-J (14%)	AA-R (13%)	WO-L (25%)	AR-B (20%)	RA-B (13%)
			RA-J (8%)	AW-L (22%)	RR-B (8%)	CT-B (12%)
			TR-L (6%)	AR-J (6%)	WO-L (6%)	AA-B (11%)
			RR-L (6%)			WA-B (10%)
Early-Low	OC-L (100%)	RR-L (19%)	AA-L (45%)	AR-B (32%)	AW-L (51%)	
		CT-L (11%)	AA-R (35%)	RA-B (31%)	AW-B (19%)	
		WO-L (10%)		AA-B (14%)	AW-J (12%)	
		TR-L (10%)		AA-L (8%)		
		WO-B (8%)		RA-J (6%)		

Figure 3 shows the corresponding HMM for the low performing group working on the early module. This model has 5 states. Upon close observation, we see that the reduction in states is due to the *Wrap-up State* and the *HL Backtracking State* being merged into a single state (State 3). The other states that we have identified all appear in this model, albeit with a somewhat different compositions of activities, which we will be one of the key issues in our investigation. The identified states are summarized in Table 3.

Using these interpreted states, our first level of analysis would be to see if there are differences in the proportion of time that the students spend in each of the states. The computed stationary probabilities are summarized in Table 4. It is clear that the High-Early and the High-Late models are almost identical. Thus we may conclude that the high-performing students' learning behaviors remained the same as they studied the different modules. On the other hand, the low performers' behaviors changed significantly from the early to the late module, primarily in the amount of time they spent on wrap up, as well as the amount of backtracking between resources and assessments.

The mixed state that was observed early also went away in their learning activities in the late module. One may conclude that with time, the low performers started behaving more like the high performers, and one may conjecture that this was because the system was helping the low performers become better learners. This needs to be investigated further. In terms of hypothesis 2, Table 4 implies that the low-performer data supports hypothesis 2, but that the high-performer data does not. One may conclude that this is reasonable, because the high performers have figured out how to do well, and do not need to modify their behaviors.

Table 3. Descriptions of States

<i>Start State</i>	Students move from the <i>Overview</i> to the <i>Challenge</i> phase
<i>Initiation State</i>	Students move from the <i>Challenge</i> to mainly the <i>Initial Thoughts</i> phase or the <i>Resources</i> phase
<i>Assessment State</i>	Students stay in the <i>Assessments</i> phase, mainly moving to the next question, or retrying a question they get wrong
<i>Wrap-up State</i>	Students proceed to the <i>Wrap-up</i> phase and conclude the module
<i>RA (Resources-Assessment) Backtracking State</i>	Students revisit previous resources or assessment questions
<i>HL (High-Level) Backtracking State</i>	Students revisit <i>Initial Thoughts</i> or the <i>Challenge</i> phases

Table 4. Stationary Probabilities

	Start	Initiation	Assessment	Wrap-up	RA Backtrack	HL Backtrack	Merged Wrap-up and RA Backtrack
Low-Early	3	-	40	2	29	-	25
High-Early	2	2	50	33	10	2	-
Low-Late	2	2	37	12	46	-	-
High-Late	2	2	50	33	11	-	-

Another interesting difference revealed by the models is how the proportions of activities significantly differ among the different performance groups. In particular, there is much higher incidence of linear transitions in the *Initiation* State and the *Assessment* State among the higher performing groups. These results are summarized in Table 5.

6 Discussion and Comparisons

Using the results from the hidden Markov model analysis, we sought to investigate how the high-performing students transitioned through the different phases in the system in contrast with the low-performing students. In particular, we were interested in the students' relative tendencies to follow the explicit, canonical model presented by the system, given the great degree of freedom that they were also provided. Confirming our first hypothesis, we found that the high-performing students moved more linearly from

the *Challenge* to the *Initial Thoughts* phase, and also that they moved more linearly through the *Assessment* phase. We also found that the high-performing students spent markedly more time in the *Wrap-up* state, while the low-performing students engaged much more in backtracking through previous assessment questions and resource items. While these results are promising, we should note that the results likely stem from the high-performing students having a better understanding of the domain, and, therefore, showing better abilities to plan their learning and assessment tasks. On the other hand, the low performers seem to flounder more, which manifest as backtracking actions.

Table 5. Proportions of Linear Transitions

	Low-Early	Low-Late	High-Early	High-Late
Initiation State	-	75	82	82
Assessment State	45	52	68	68

Our results are complemented by our colleagues’ findings from using process mining techniques to examine similar problems [5]. This approach uses specifically structured models to investigate individual problems at a greater detail. For example, our colleagues provide a Petri Net Model for how the learners navigate through the STAR Legacy cycle. Such a model provides precise numbers of students who follow the sequential path and those who deviate from it at any point. A result of interest to us is the finding that most students follow the canonical model at first, but start to diverge as they use the system. Of particular concern is that most of the divergence stems from the students’ avoidance of the *Thoughts* phase in the later modules; the *Thoughts* phase figures prominently into the cycle as it is designed to create “cognitive dissonance” among the learners [4]. Complemented with our finding that higher-performing students are more likely to proceed linearly and not skip the *Thoughts* phase. It might be worth investigating why the students grow to avoid the *Thoughts* phase, and whether this avoidance actually leads to a drop in performance. Also, the direction of the causality should be explored: does proceeding linearly lead to higher performance, or does higher cognitive ability lead to proceeding linearly? Our hypothesis is that learners face increased cognitive load for the metacognitive tasks, such as strategic planning, and that following the structural scaffolds in the environment, i.e. progressing linearly, would help them manage the cognitive load, and hence perform better.

7 Conclusions and Future Work

In this paper, we have shown that the hidden Markov model approach is general, and can be applied to performing exploratory data analysis in new domains. The analysis implies that there are differences in transitional state behavior between the high and the low-performing students. Namely, the results indicate that the high-performing students move through the model more linearly and spend less time backtracking than the low-performing students. Our results, combined with our previous work, show the general nature of the hidden Markov model approach for exploratory sequence analysis. Like speech synthesis applications [8], HMMs may prove to be a valuable tool for analyzing learning behaviors in a variety of domains. We have also shown how exploratory data

analysis can be complemented by process mining analysis to help examine problems in greater detail. In particular, we imagine that interesting and inconclusive results from exploratory data analysis may be investigated using process mining to yield more definite findings. We believe that the combination of these approaches will lead to many promising avenues to extend our research in the future.

References

- [1] Biswas, G., Leelawong, K., Schwartz, D., & Vye, N. (2005). Learning by Teaching: A New Agent Paradigm for Educational Software. *Applied Artificial Intelligence*, 19, 363-392.
- [2] Biswas, G., Roscoe, R., Jeong, H., & Sulcer, B. (2009). Promoting Self-Regulated Learning Skills in Agent-based Learning Environments. Hong Kong: Proceedings of the 17th International Conference on Computers in Education.
- [3] Bransford, J., & Schwartz, D. L. (1999). Rethinking transfer: A simple proposal with multiple implications. *Review of Research in Education*, 24, 61-101.
- [4] Chen, M. (1995). A methodology for characterizing computer-based learning environments. *Instructional Science*, 183-220.
- [5] Howard, L., Johnson, J., & Neitzel, C. (2010). Examining learner control in a structured inquiry cycle using process mining. *3rd Int. Conf. on Educational Data Mining*. Pittsburgh.
- [6] Jeong, H., & Biswas, G. (2008). Mining Student Behavior Models in Learning-by-Teaching Environments. *Educational Data Mining*, (pp. 127-136). Montreal.
- [7] Jeong, H., Gupta, A., Roscoe, R., Wagster, J., Biswas, G., & Schwartz, D. (2008). Using Hidden Markov Models to Characterize Student Behavior Patterns in Computer-based Learning-by-Teaching Environments. *Intelligent Tutoring Systems*, (pp. 614-625). Montreal.
- [8] Li, C., & Biswas, G. (2002). A Bayesian Approach for Learning Hidden Markov Models from Data. *Special issue on Markov Chain and Hidden Markov Models, Scientific Programming*, 10, 201-219.
- [9] Neitzel, C., Johnson, J., & Howard, L. (2010). Behavioral data and indications of self-regulation: What learner choices reveal about their strategies. *Advances in Engineering Education*.
- [10] Rabiner, L. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proc. IEEE*, 77(2), 257-285.
- [11] Schwartz, D., Lin, X., Brophy, S., & Bransford, J. (1999). Toward the development of flexibly adaptive instructional designs. In C. Reigeluth, *Instructional-design Theories and models: A New Paradigm of Instructional Theory II* (pp. 183-214).

Data Mining for Individualised Hints in eLearning

Anna Katrina Dominguez, Kalina Yacef, James R. Curran

{adom0244, kalina, james}@it.usyd.edu.au

School of Information Technologies, University of Sydney, Australia.

Abstract. In this paper we present a tool where both past and current student data is used live to generate hints for students who are completing programming exercises during a national programming online tutorial and competition. These hints can be links to notes that are relevant to the problem detected and can include pre-emptive hints to prevent future mistakes. Data from the year 2008 was mined, using clustering, association rules and numerical analysis, to find common patterns affecting the learners' performance that we could use as a basis for providing hints to the 2009 students. During its live operation in 2009, student data was mined each week to update the system as it was being used. The benefits of the hinting system were evaluated through a large-scale experiment with participants of the 2009 NCSS Challenge. We found that users who were provided with hints achieved higher average marks than those who were not and stayed engaged for longer with the site.

1 Introduction

Educational Data Mining (EDM) has been applied to extract patterns from student data, but it is usually employed after the course has finished, for instance to analyse student behaviour. The idea of carrying out data mining as an integrated, functional part of the system has yet to be explored, and is outlined as future work by [1].

In this paper, we investigate whether data mining can be used to generate tailored feedback for users of an existing e-learning system without the overhead of building an ITS from scratch. We build on the NCSS Challenge¹, an annual web-based programming competition in which students are taught the basics of programming in Python. Each week they are given sets of notes and questions to answer by submitting source code. Challenge problems become very difficult in later weeks and in previous years, this had resulted in a high dropout of participants. We aim to address this issue by providing more support during the Challenge, in the form of hints and easier access to notes.

In our system, feedback is provided to users through the introduction of data mining *directly into the system loop* to provide hints. The hinting system helps students with topics that they are struggling with, by suggesting parts of the notes to reread and previous questions to revisit. The hints are generated automatically using patterns mined from both the previous year's data and the data generated by live users of the system in previous weeks. This is in contrast to the manual construction of solutions and feedback in previous systems. The hinting system is fully operational, and was used in the 2009 NCSS Challenge. We conducted an experiment to measure its effectiveness in providing

¹ <http://challenge.ncss.edu.au>

aid for students by presenting hints to some students and not others. We found that the system helped students considerably, resulting in a large and highly significant effect on the marks and retainment of those students who received automated hints.

2 Related work

The excellent review by Shute of formative feedback [2] provides guidelines on what to look for and what to avoid when designing feedback, and outlines that learning is best improved when the feedback is kept as simple and focused as possible. Hints and cues fulfil this objective. The Lisp Tutor [3], ELM-ART [4] and SQLT-Web [5] are all examples of successful programming tutors. Each is able to aid student problem solving by providing feedback in the form of very specific hints and error messages, even to the point of providing correct code snippets if the student is struggling. All three systems rely on example solutions and a finite set of rules that specify what code is valid for a given task, and any deviations that will result in error can be identified quickly. Encoding such sample solutions and rules requires the formulation of pre-determined feedback for students, which can be expensive in terms of time and cost, and may not cover all scenarios. These methods are fairly domain specific, and therefore cannot be easily generalised to other systems.

While the use of data mining to aid the diagnosis of students' behaviour and ability is common, relatively little work has been done in using data mining to support student problem solving. One system that aims to do this is the Logic-ITA [6], where the system takes into account past associations of student mistakes to provide on-the-fly, proactive feedback to the students. The identification of students who are in difficulty in a programming course has been the aim of some studies such as [7] and [8], but with the aim of predicting failure or dropping out of a course, not for providing hints.

3 The NCSS Challenge site

The NCSS Challenge is an annual online programming competition that has been running for Australian high school students and teachers since 2005. The Challenge runs for five weeks during which participants are provided with notes and questions on a weekly basis, and are given feedback and marks out of 10 based on their submissions to the questions. When a user submits an answer to a question, it is evaluated by an automatic marker. This marker generates basic feedback, giving high-level descriptions of tests that the submission passed or failed to the users. We have now developed an extension to the Challenge website that provides users with additional feedback when they fail a question, in the form of hints which we describe next.

3.1 Hints

We implemented two main types of hints: “pre-emptive” and “post-failure”. Pre-emptive hints are available to weak students before they submit any code for a question. Post-failure hints are provided after a student's submission failed. Both types of hints lead to a hints page, automatically generated and specific to each student and question. It suggests links for (1) topics to review, (2) similar questions to attempt and (3) easier questions to

attempt. A screenshot of a post-failure hint is shown in Figure 1. Pre-emptive ones look similar, except they do not contain any associated topics to review.

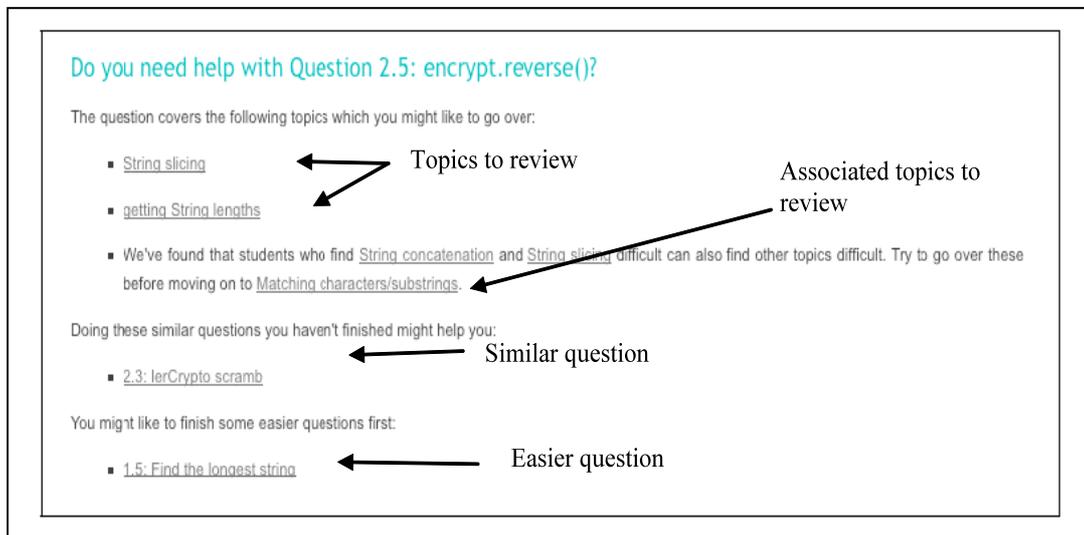


Figure 1. Screenshot of a post-failure hint page

The *topics to review* hints link to specific sections of the notes that address these topics (e.g. string slicing in Figure 1). Each section of the notes and each question were tagged with the relevant Python topics, as per a lightweight ontology that we built for that purpose, therefore allowing for the notes and questions to be related to each other. The *question* hints link to other Challenge questions selected by the data mining. The hints only suggest notes and questions that all students already have access to.

4 Data mining for the hinting system

As mentioned earlier, we used data mining as a basis for the generation and triggering of relevant hints for each student. The data came from the past 2008 Challenge and the live 2009 Challenge, including the questions, the corresponding topic tags from the ontology and the students' results and submissions. Data mining was carried out using clustering, association rule mining and simple numerical analysis, the results of which were used as various components in the final hinting system as indicated below.

Table 1. Generation methods for each part of the hint

Part of the hint	Method to generate it	Pre-emptive hints	Post-failure hints
Topics to review	Topic tags	√	√
Associated topics to review	Association rules on topics of failed submissions		√
Easier questions	Clustering questions (difficulty)	√	√
Similar questions	Clustering questions (similarity)	√	√

Table 2. Triggering mechanisms for hints

Hint	Triggered for	Method
Pre-emptive	Weak students	Clustering (students per ability)
Post-failure	All students after x failures	Simple numerical analysis

4.1 Mining the 2008 data

We mined the Challenge data from the previous year (2008) with the overall aim to (i) try to extract useful information that could be used to generate hints for our 2009 Challenge students and (ii) experiment with some techniques such as clustering to select and fine-tune our methods before using them live in 2009.

Throughout the five weeks of the 2008 Beginners Challenge, 16,814 submissions were gathered from 712 separate users. There were 25 questions in total (5 per week) that were available to the students, usually in increasing order of difficulty. Students could submit several times for the same question until successful. All these attempts were recorded, along with the mark eventually obtained by students in each question.

4.1.1 Clustering students

The aim was to group students based on their abilities. We used the K-Means clustering algorithm used in Tada-Ed [9]. For each student ID, we collated the following attributes for each of the 25 questions: whether the student attempted the question (nominal), whether the student eventually passed the question (nominal), and the marks gained for the question (numeric [0 or 5-10]). We also computed the average numbers of passed and failed questions, and the average number of submissions before the student passed a question. Clustering with these attributes produced three distinct groups, which we identified as being “strong”, “medium” and “weak” students. While this result was only relevant for the 2008 students, the effectiveness of clustering with these pre-processed attributes indicated that clustering was a viable technique for discriminating between students.

4.1.2 Clustering questions

We clustered the questions with two distinct aims: to find questions that were similar to each other and to group questions by difficulty. Our goal was to remind students of other related questions that may help them with the question they were considering at the time. We again used the K-means algorithm. The similarity-based clusters were extracted using the question metadata (topic tags). We found 5 clusters, as each of the 5 weeks of the Challenge introduced new topics. The difficulty-based clusters of questions were extracted based on the number of students who passed each question and the percentage of students who passed it that attempted it. Similarly to the clustering of students, we found three clusters, which we identified as “easy”, “medium” and “hard”. Table 3 shows the 2008 questions with performance statistics and their difficulty clusters. We found that the three groups were not grouped chronologically, but that several medium questions

appeared earlier than the last of the easy questions, and that the hard questions were interspersed through the medium ones. This meant that clustering to generate difficulty levels was worthwhile, as a simple chronological ordering would not have worked.

Table 3. The 2008 questions and their clustered difficulty rankings

Week	Question	Passed	%	Difficulty	Week	Question	Passed	%	Difficulty
1	hello-world	691	98	easy	4	aardvark-files	227	96	medium
1	hello-name	652	97	easy	4	word2sms	226	98	medium
1	string-repeater	589	96	easy	4	find-max	213	98	medium
1	fuel-consumption	461	94	easy	4	primes-contact	161	79	hard
1	eliza-simple	418	90	easy	4	animal-interactive	165	90	medium
2	aardvark	461	91	easy	5	translation-latin2english	139	92	medium
2	pig-latin-basic	434	97	easy	5	function-swedish	118	82	hard
2	simon-says	392	95	easy	5	rational-simple	101	95	medium
2	hailstone-numbers	353	96	easy	5	rpn-eval-simple	87	91	medium
2	animal-wordgame	315	93	medium	5	eliza-templates	38	64	hard
3	weird-units	418	98	easy					
3	leap-years	345	95	easy					
3	birthday-cards-for-loop	321	96	medium					
3	diamonds	278	96	medium					
3	eliza-interactive	226	90	medium					

4.1.3 Mining associations in topics

The aim was to find association rules that indicated which topics should be mastered before another question was attempted, so that the hints could suggest topics that students should review before moving onto a more complex one. We initially aimed to generate the association rules by assigning scores for each student on the 46 tags used in the 2008 Challenge. If a student passed a question, they were given one point for each tag in the question; if they did not pass a question a point was taken away for each relevant tag. Once all the scores were calculated, positive scores were labelled as “passed” and zero or negative scores were labelled as “failed”. This method, however, was too coarse-grained. If a student failed a question tagged with a large number of tags but only had problems with one or two topics, they would be penalised for all the topics. We revised this to a more fine-grained method, and mined sequences of tags that students failed on. We ordered the students’ results chronologically, and kept an ordered sequence of the tags for each question they made an incorrect submission to. We then used these sequences to generate association rules. Originally, we set the support and confidence to 70%. However, this excluded many advanced topics as they occurred less often in the students’ sequences. This was because they were introduced in later weeks, and as such, fewer questions were tagged with them and fewer students attempted those questions. We

therefore lowered the support and confidence to 20%, and used cosine, which has been shown to be a more appropriate evaluation metric for educational data [10]. We post-processed the rules generated by the aPriori algorithm [11] to discard rules with a cosine of less than 0.65 [10] and rules with topics out of the order in which they appeared in the notes. We only retained rules that had two topics in the antecedent and one in the consequent. We finally manually extracted the rules in which the three topics involved were related to one another to remove trivial rules. We ended up keeping 83 rules, two of which are shown in Table 4. The first rule means that students who struggle with basic arithmetic in Python and comparison operators also struggle with how to loop over a set of values, and the second one means that those who struggle with converting to integers and while loops also struggle with stopping after a number of iterations.

Table 4. Examples of association rules found

$\begin{aligned} &\text{mth:arth, mth:bool:comp} \rightarrow \text{ctrl:loop:term:vals} \\ &\text{typ:int:cast, ctrl:loop:whl} \rightarrow \text{ctrl:loop:term:nums} \end{aligned}$
--

4.1.4 Numerical Analysis

Aside from the more complex data mining algorithms, we subjected the 2008 data to some simple numerical analysis to find frequencies and averages for certain aspects of the data. An important measure was to have an idea of the “give-up point”, that is, the number of wrong submissions a student made before he or she stopped attempting it. To do this we found, for each question, the total number of submissions made by students who never passed the question. We then averaged this over the number of students who had attempted but not passed that question. We then computed the mean of the averages already found, which was found to be 3.7. This was used in the final system as the point at which students were presented with post-failure hints; a student would only receive such hints after making their fourth incorrect submission to a question.

5 Experiment and results with 2009 Challenge

We tested our hinting system on participants of the 2009 NCSS Challenge. We ran the experiment using the live data being generated by the Challenge participants. We evaluated through a controlled experiment whether the hinting system had a positive effect on student performance, based on their marks and the ability clusters they were grouped into.

5.1 Experiment design

The Challenge ran for five weeks from mid August to late September 2009. There was an overnight period between one week’s questions being closed to submissions and the next week’s questions being released, which allowed us to carry out the data mining using the live system and current participant data and upload the new clusters for the next week. This also involved finalising the tags for the questions, and clustering both the questions and the students. All data mining was carried out as described in the previous section.

1303 participants registered for the 2009 NCSS Beginners Challenge. We took the first 1000 participants to enrol to be involved in our experiment, and provided them all with hints for the first week of the Challenge. At the end of the first week, we found any participants who had not yet made any submissions, and excluded them from our experiment due to inactivity, ending with a population of 584 students. At the end of week 1 where everyone received hints, we split the students into 2 equal-sized groups: a test group, which received hints, and a control group, which did not from week 2 to 5. All 584 students were clustered based on their week 1 results. We then split each cluster in half for our hinted and control groups, based on the schools the students were registered with so students from the same school were in the same group.

At the end of each week, students were clustered according to abilities (as in 4.1.1), using the cumulative student data. Question clusters were also updated weekly. We discovered each cluster could be mapped to a specific topic when we clustered the 2008 questions. Since new topics were introduced each week, we increased the number of similarity clusters over the weeks. We created two clusters in week 1 and week 2, and then increased this weekly until there were five clusters in week 5.

Unlike the 2008 questions, in which all the data on student performance per question was known and available for clustering, the data for the 2009 Challenge was being generated during the experiment. As such, we were required to estimate the difficulties for each of the new week's questions instead of deriving them solely from the participant results. At the end of each week, we analysed the results data from that week and clustered the questions to assign their difficulty levels, then estimated and assigned a difficulty to the new questions for the next week. At the end of the next week we readjusted the difficulty level based on the results generated by the participants for those questions. While the difficulty levels sometimes needed readjustment, we were generally able to estimate the question difficulties accurately at the start of the week and could predict the difficulties that were generated at the end of the week by the clustering.

5.2 Results of 2009 clustering

The techniques and attributes of the clustering of questions and students was the same as in 2008. The final set of question clusters by similarity is presented in Table 5. These were the clusters as used for Week 5 of the 2009 Challenge.

Table 5. The 2009 questions clustered by similarity

Cluster ID	Question	Cluster ID	Question
0	file-merge snakes-n-ladders	3	anagram-super biscuit-value black-ballon-calc cellular-automata debug-longest-string debug-odd-one-out download-time hello-name hello-world pirate-count-lines
1	bfs-sudoku function-list-calculator function-pirate-translate		
2	banner-printer numbers-to-words pirate-decode-repeats shopping-list-file sort-uniq substitution-cipher sudoku-checker		
		4	crypto-scrambler-simple crypto-substitution-undo hollow-square pirate-yarrrr

The data for the difficulty clustering of the questions is shown in Table 6, which also includes the clusters questions were assigned to at the start of week 5. The week 5 difficulty clusters (in italics) were estimated.

Table 6. The 2009 questions and their clustered difficulty rankings

Week	Question	Week 5 Difficulty	Passed	%	Week	Question	Week 5 Difficulty	Passed	%
1	hello-world	easy	579	99	4	numbers-to-words	medium	272	96
1	hello-name	easy	563	99	4	file-merge	medium	186	94
1	pirate-yarrrr	easy	485	95	4	substitution-cipher	medium	171	93
1	download-time	easy	404	89	4	banner-printer	hard	123	95
1	debug-longest-string	easy	383	91	4	sudoku-checker	hard	118	87
2	biscuit-value	easy	442	97	5	shopping-list-file	<i>medium</i>	97	92
2	pirate-count-lines	easy	364	97	5	function-pirate-translate	<i>medium</i>	83	95
2	crypto-scrambler-simple	medium	282	93	5	function-list-calculator	<i>medium</i>	75	97
2	debug-odd-one-out	easy	364	97	5	cellular-automata	<i>hard</i>	57	95
2	crypto-substitution-undo	medium	286	88	5	bfs-sudoku	<i>hard</i>	29	74
3	black-ballon-calc	easy	343	97	5	snakes-n-ladders	<i>hard</i>	17	61
3	hollow-square	medium	305	97					
3	pirate-decode-repeats	medium	212	95					
3	anagram-super	medium	232	97					
3	sort-uniq	medium	223	98					

5.3 Evaluation

Firstly, we measured student performance based on their average overall marks. For each of the 584 students in the experiment, we calculated their average score out of 10 for the questions they made at least one submission to. We then calculated the mean of these scores over the hinted students, and the mean of the scores for the control group. The hinted group's mean score was 4.02 (sd = 2.78), while the control group had a mean score of 3.18 (sd = 2.71). This was a difference of 0.84, i.e. an increase of 26.4%, with a significance of $p < 0.0006$ using an Approximate Randomisation test [12]. We used this because the students' marks were not normally distributed, making a t-test inappropriate. This indicates that the hints substantially helped students when solving problems and lead to a significantly higher level of correct submissions being submitted.

Table 7. The number of students who submitted at least one question per week

Week	1	2	3	4	5
Hinted	292	244	203	172	75
Control	292	220	163	120	38

We also investigated whether users who received hints were active on the site for longer than those in the control group. Table 7 shows the number of hinted and control group users who submitted an answer to at least one question per week. There were consistently more users in the hinted group who made submissions, meaning the hinted group of users had an overall higher level of participation over the five weeks of the Challenge. The hints therefore had a distinctly positive effect on students' willingness to stay engaged with the course.

To get insights on students' satisfaction with the hinting system, we presented them with a survey at the end of the course. Students were asked to rate the relevance of the hints to the questions they were answering and the topics they had difficulty with by using a five-point Likert scale. 67% of students found the topics "relevant" or "somewhat relevant" and 90% of them found the questions "relevant" or "somewhat relevant". Therefore, it is clear that as far as the users were concerned, the methods for choosing topics to present were effective. In addition, 71% of students stated they would like more hints.

Overall, the survey responses showed that the students found the hints helpful. When asked to provide comments, many of the students emphatically stated that the hints had helped them with problem solving, with students giving extremely positive comments and requests for the hints to continue in future years of the Challenge. Furthermore, one student found that the hints helped her access the notes much more effectively, which was our overall aim for the system: *"I found the tips more helpful, because when we are using the notes to solve the problem we really don't know where to go and what to do or which formula to use. But after using the hint formula we know where to go and what to use for solving the problem. So I reckon that the hint boxes were a very smart way to access the notes that can help us to solve the problems."*

6 Future work and conclusion

Our project aimed to integrate data mining into an e-learning system to generate dynamically tailored hints for users. These hints give users immediate help by directing them to parts of the notes and questions that are relevant to questions that they find difficult. We built this hinting system for the NCSS Challenge website, using association rule mining and clustering on the data produced by live users, to update the system as it was being used. We evaluated the hinting system through a large-scale experiment conducted with participants of the 2009 NCSS Challenge. In the future, we would like to compare the effectiveness of our dynamic hints with statically generated hints.

We found that users who were provided with hints achieved a 26% higher average mark than those who were not provided with hints, with statistical significance of $p < 0.0006$. Furthermore, we found qualitative evidence through positive student feedback that the hinting system had greatly helped users. These results show that the use of data mining to provide hints as part of the system loop is extremely effective, and can be used to build intelligent systems with much less of the time and cost expenses associated with traditional ITSSs.

References

- 1 . Romero, C. and S. Ventura, Educational Data Mining: A Survey from 1995 to 2005. *Expert Systems with Applications*, 2007. **33**(1): p. 125-146.
- 2 . Shute, V.J., Focus on formative feedback. *Review of Educational Research*, 2008. **78**(1): p. 153-189.
- 3 . Reiser, B.J. and J.R. Anderson, The LISP tutor: it approaches the effectiveness of a human tutor. *Byte*, 1985. **10**(4): p. 159-175.
- 4 . Weber, G. and P. Brusilovsky, ELM-ART: An adaptive versatile system for Web-based instruction. *International Journal of Artificial Intelligence in Education*, 2001. **12**(4): p. 351-384.
- 5 . Mitrovic, A., An intelligent SQL tutor on the web. *International Journal of Artificial Intelligence in Education (IJAIED)*, 2003. **13**(2): p. 173-197.
- 6 . Merceron, A. and K. Yacef, Educational Data Mining: a Case Study, in proceedings of *Artificial Intelligence in Education (AIED2005)*, C.-K. Looi, G. McCalla, B. Bredeweg, and J. Breuker, Editors. 2005, IOS Press: Amsterdam, The Netherlands. p. 467-474.
- 7 . Ma, Y., B. Liu, C.K. Wong, P.S. Yu, and S.M. Lee, Targeting the right students using data mining, in proceedings of *ACM SIGKDD international conference on Knowledge discovery and data mining*. 2000: New York, NY, USA. p. 457-464.
- 8 . Hamalainen, W., T.H. Laine, and E. Sutinen, Data mining in personalizing distance education courses, in proceedings of *World conference on open learning and distance education*. 2004: Hong Kong. p. 1-11.
- 9 . Merceron, A. and K. Yacef, TADA-Ed for Educational Data Mining. *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning*, 2005. **Volume 7, Number 1**: p. <http://imej.wfu.edu/articles/2005/1/03/index.asp>.
- 10 . Merceron, A. and K. Yacef, Interestingness Measures for Association Rules in Educational Data, in proceedings of *International Conference on Educational Data Mining*. 2008: Montreal, Canada.
- 11 . Agrawal, R. and R. Srikant, Fast Algorithms for Mining Association Rules, in proceedings of *VLDB*. 1994: Santiago, Chile.
- 12 . Chinchor, N., Statistical significance of MUC-6 results, in proceedings of *Fourth Message Understanding Conference (MUC-4)*. 1992. p. 390-395.

Off Topic Conversation in Expert Tutoring: Waste of Time or Learning Opportunity?

Blair Lehman, Whitney Cade, & Andrew Olney

{balehman|wlcade|aolney}@memphis.edu

Institute for Intelligent Systems, University of Memphis, Memphis, TN 38152

Abstract. While many aspects of tutoring have been identified and studied, off topic conversation has been largely ignored. In this paper, off topic conversation during 50 hours of one-to-one expert tutoring sessions was analyzed. Two distinct methodologies (Dialogue Move occurrence and LIWC analysis) were used to determine the anatomy of off topic conversation. Both analyses revealed that the expected social talk occurred, but pedagogically-relevant talk emerged as well. These occurrences may reflect the discussion of more global pedagogical strategies. These findings suggest that off topic conversation may serve a useful purpose in tutoring and that further investigation is warranted.

1 Introduction

When it comes to tutoring, expert human tutors are widely acknowledged as the gold standard for producing learning gains. According to [1], accomplished tutors produce an effect size of 2.0 sigma (approximately 2 letter grades) when compared to learning gains found in a classroom scenario. Accomplished tutors also outperform tutors with domain knowledge but no pedagogical training [2] and Intelligent Tutoring Systems [ITSs; 3]. One assumes that expert tutors excel for a number of reasons. Expert tutors use pedagogical techniques that depart from those seen in previous work on novice tutors [4, 5]. It has also been hypothesized that expert tutors provide emotional support for their students [6] while increasing their feelings of competence [7]. The real question, unaddressed by current research, is this: how do expert tutors build the necessary rapport with students to produce high learning gains and boosted confidence?

Recent research proposes that this rapport building takes place during the non-pedagogical parts of a tutoring session [8]. While previous work on off-task behavior has shown a negative relationship with learning [9, 10], it is important to distinguish off-task behavior from off topic conversation. Off-task behavior involves such actions as gaming the system [9]. During off topic conversation, on the other hand, the student is still invested in the tutoring system and the divergence is still under the close supervision of the tutor. An investigation into a corpus of tutoring sessions collected by [8] suggests that rapport building occurs in these moments of "social talk." They contend that not only is social talk a considerable portion of their corpus (20%), but that it is repeatedly used for greetings, rapport building, social coordination, and general learning strategies. It is during social talk that the tutor will give "pep talks," potentially to increase both student motivation and self-confidence. They casually observed that those tutors who were the least effective in imparting knowledge also came up short in building positive rapport. Observations such as these deserve more systematic, follow-up attention. If social talk/off topic conversation is the source of rapport building, then it is certainly deserving of study. Rapport building helps to create good communication [11] and is viewed as very

important by students [12]. However, these observations were made in regards to a fairly contrived corpus; the tutors were graduate students, and the students were recruited as part of an experiment (though they were in a class on the domain topics). Additionally, these sessions occurred over the computer through an interface with video and chat capabilities. This raises an issue of ecological validity: do these observations generalize to natural tutoring sessions, and are they true of expert tutors as well?

To answer this question, an investigation into the practices of expert tutors in a natural tutoring setting is needed. The present study used a previously collected corpus of 50 expert tutoring sessions with ten expert tutors to investigate off topic conversation and rapport building. Though these sessions have been coded and examined in the past, off topic conversation in expert tutoring has been given little more than a cursory glance. However, we suspect that the interaction between student and tutor in these “off topic conversations” forms the basis of the student-tutor rapport, which will later allow the tutor to alternately encourage and critique the student without seeming disingenuous or harsh. We know that expert tutors give negative feedback where novice tutors never would [13, 14], but we also know that expert tutors help the student build confidence in themselves [7]. Perhaps the key to this paradoxical situation lies within the unstudied off topic conversation.

We approach this investigation using two distinct methods of exploring off topic conversation. The first of these methods seeks to establish a general layout of what is occurring during non-pedagogical modes through a manual coding scheme. This analysis will provide a broad picture of what is occurring during these discussions and reveal any potential pedagogical purpose to off topic conversation. We will also use the Linguistic Inquiry and Word Count tool (LIWC) [15] to get at the meaningful differences between Off Topic and Scaffolding. Both of these phases of tutoring are highly interactive, yet they greatly differ in purpose. While the purpose of Off Topic is assumed to be irrelevant conversations, Scaffolding has been shown to be the epitome of problem solving during these tutoring sessions [4]. By comparing these two different modes along linguistic feature dimensions, we can better establish the instructional and rapport-building components of Off Topic.

2 Expert Tutoring Corpus

In this study, we examined one-to-one, human-to-human expert tutoring sessions. Ten expert math and science tutors participated in 50 sessions in the current study. The tutors had to meet four criteria to be considered an expert: licensed to teach at the secondary level, five or more years of tutoring experience, employed by a professional tutoring agency, and highly recommended by school personnel. While one student was receiving tutoring in order to obtain a GED, the other student participants were in grades 7 to 12. All of the students were in academic trouble and actively sought out tutoring. Each student participated in a maximum of two tutorial sessions, while each tutor participated in between two and eight tutoring sessions.

Fifty, hour-long tutoring sessions were recorded with thirty-nine of the tutors’ students. The tutoring locations were chosen by the tutor and the student. The experimenters were

merely observers of a natural tutoring session that would have taken place regardless of this study. Prior to any tutoring session, the tutors signed an informed consent and gave informed consent forms to the parents of prospective participant-students.

Each transcript has been coded using two coding schemes: the dialogue move and the dialogue mode coding scheme [4, 16, 17]. While a move is a smaller unit of meaningful speech (such as a word or short phrase), modes are longer, sustained, pedagogically-distinct phases of a tutoring session. Dialogue moves have been categorized into three pedagogically-relevant groups: Tutor Motivational Dialogue Moves, Tutor Pedagogical Dialogue Moves, and Student Dialogue Moves. There are eight dialogue modes in the coding scheme, but only four are of interest in this study: Scaffolding, Off Topic, Introduction, and Conclusion; [4]). These modes accounted for 48.58%, 4.35%, 2.60%, and 1.85% of tutoring dialogue, respectively. For some of the analyses Introduction, Off Topic, and Conclusion are collapsed into one larger category of non-pedagogical modes (8.80% of tutoring sessions).

In preparation for the analysis of dialogue move distributions, non-pedagogical modes were isolated from the larger corpus. Proportional occurrence of student dialogue moves in non-pedagogical modes was then determined for each session. Proportions were based on the total number of dialogue moves in non-pedagogical modes in each session. Thus all instances of Introduction, Off Topic, and Conclusion in each session were combined for this analysis. Finally, the base rate proportional occurrence of each dialogue move within each session was determined to serve as a comparison.

Our next method of investigating off topic conversation involved the use of the Linguistic Inquiry and Word Count tool (LIWC), developed by [15]. In this study, we used LIWClite 7, which calculates the percentage of each document's words that fall into specific, predefined categories. Though this version of LIWC offers over 70 linguistic categories, only 16 were of interest in determining the nature of off topic conversation (see Table 3). These were selected because of their relevance to pedagogical and social dimensions of language. Only sessions including at least one instance of Scaffolding and Off Topic were included, leaving 30 sessions for this analysis. To prepare the transcripts for LIWC analysis, instances of Off Topic and Scaffolding were separated into two documents for each session. If a transcript had multiple types of each mode (e.g., three Scaffolding modes), the dialogue from those modes was compiled into one document. Those documents were then subdivided by speaker, making a maximum of four possible documents per session (Tutor-Scaffolding, Tutor-Off Topic, Student-Scaffolding, Student-Off Topic). Each document was capped at 1,000 words to control for verbosity (as Scaffolding is a frequently occurring, lengthy mode [4]). The resulting documents were then submitted to LIWClite 7 in order to find the means of every word category per document.

3 Results & Discussion

3.1 Comparison of dialogue move occurrence in off topic modes and base rate

For the dialogue move analyses, the non-pedagogical modes will simply be referred to as Off Topic. These proportions were then compared to the base rate occurrence for each dialogue move in each tutoring session. Base rate represents the average occurrence of each dialogue move throughout the entire tutoring session. Paired samples *t*-tests were conducted and significant differences were found. The means, standard deviations, *t*-values and effect sizes (Cohen's *d*) of each comparison can be found in Table 1 (tutor dialogue moves) and Table 2 (student dialogue moves).

Table 1. Comparison of tutor dialogue moves within off topic modes and base rate

	Off Topic		Base Rate		<i>t</i> -value	Cohen's <i>d</i>
<i>Motivational</i>	<i>M</i>	<i>SD</i>	<i>M</i>	<i>SD</i>	<i>df</i> = 49	
Positive Feedback	0.035	0.037	0.074	0.037	-6.86*	-1.05
Negative Feedback	0.003	0.006	0.009	0.006	-5.58*	-1.11
Humor	0.018	0.033	0.009	0.014	2.14*	0.348
Repetition	0.005	0.012	0.016	0.016	-4.67*	-0.814
Solidarity Statement	0.005	0.013	0.001	0.002	1.80 [†]	0.335
General Motivational	0.016	0.026	0.004	0.005	3.30*	0.613
Off Topic	0.268	0.109	0.068	0.039	14.46*	2.45
<i>Pedagogical</i>						
Forced Choice	0.000	0.000	0.002	0.003	-4.26*	-0.852
Hint	0.001	0.006	0.013	0.009	-8.20*	-1.65
Preview	0.007	0.019	0.003	0.004	1.79 [†]	0.341
Prompt	0.001	0.004	0.020	0.021	-6.33*	-1.28
Pump	0.000	0.002	0.020	0.005	-4.89*	-0.965
Paraphrase	0.000	0.002	0.003	0.003	-3.98*	-0.824
Simplified Problem	0.006	0.012	0.034	0.016	-10.25*	-1.95
Example	0.001	0.007	0.006	0.008	-3.63*	-0.586
Counter Example	0.000	0.000	0.000	0.001	-3.75*	-0.750
Provide Correct Answer	0.000	0.002	0.011	0.008	-9.16*	-1.81
Direct Instruction	0.069	0.069	0.182	0.045	-12.54*	-1.93
Comprehension Gauging Question	0.034	0.043	0.032	0.023	0.330	0.058

[†] = $p < .1$; * = $p < .05$. $d \approx .2, .5, .8$ indicate small, medium, and large effects, respectively [18]. Significantly larger mean values are italicized.

As was expected, student and tutor off topic dialogue moves along with other socially-focused dialogue moves occurred more frequently during the non-pedagogical modes [8]. Consistent with this expectation, problem solving and other pedagogically-focused dialogue moves occur outside of non-pedagogical modes. In particular, the absence of tutors asking questions, students answering questions, and tutors giving feedback demonstrates that non-pedagogical modes are truly a separate, distinct time of the tutoring session.

Table 2. Comparison of student dialogue moves within off topic modes and base rate

	Off Topic		Base Rate		<i>t</i> -value	Cohen's <i>d</i>
<i>Answer</i>	<i>M</i>	<i>SD</i>	<i>M</i>	<i>SD</i>	<i>df</i> = 49	
Correct	0.012	0.023	0.056	0.032	-8.77*	-1.583
Partial	0.004	0.012	0.024	0.026	-5.72*	-0.966
Vague	0.008	0.016	0.019	0.015	-4.10*	-0.706
Error-Ridden	0.001	0.004	0.011	0.009	-7.41*	-1.493
None	0.001	0.005	0.006	0.009	-4.76*	-0.666
<i>Question</i>						
Social Coordination	0.015	0.026	0.007	0.009	2.79*	0.425
Common Ground	0.005	0.013	0.024	0.019	-6.22*	-1.186
Knowledge Deficit	0.005	0.009	0.011	0.010	-3.85*	-0.665
<i>Metacognition</i>						
Misconception	0.000	0.002	0.004	0.004	-4.89*	-0.977
Metacomment	0.030	0.042	0.020	0.015	2.01*	0.321
<i>Action</i>						
Think Aloud	0.001	0.003	0.006	0.010	-3.26*	-0.643
Read Aloud	0.000	0.000	0.003	0.013	-1.93†	-0.385
Work Silently	0.001	0.005	0.015	0.019	-5.19*	-1.006
Off Topic	0.215	0.117	0.059	0.038	10.65*	1.791

† = $p < .1$; * = $p < .05$.

Those occurrences which break from these general patterns are of particular interest. The only pedagogical move to occur more frequently in non-pedagogical modes was preview. This could indicate that non-pedagogical modes are being used as a transition between new topics or problems. Preview could potentially be occurring within Introduction, which can be thought of as a preview to the entire tutoring session. For student dialogue moves, the higher occurrence of metacomment shows that non-pedagogical modes contain discussions of the student's knowledge. Many metacomments are delivered in response to tutor comprehension-gauging questions (i.e., "Do you understand?" "Okay?"). The fact that comprehension-gauging questions are not significantly occurring during non-pedagogical modes suggests that student knowledge and comprehension is being discussed in a different context.

However, the strongest occurrences by far were still tutor ($d = 2.45$) and student ($d = 1.79$) off topic dialogue moves. Given that these dialogue moves serve as a catchall for any topic outside of the tutoring topic (e.g., algebra), it is difficult to truly determine what occurs during non-pedagogical modes. While it was casually observed that these dialogue moves ranged from after school activities to study strategies, the exact proportion of each is currently unknown. There are two options for expanding our analysis of off topic dialogue moves. One is to create a new coding scheme that makes finer distinctions in off-topic. The other option, which we discuss next, is to use a text analysis tool to look for text-level features that might show what's going on inside off topic

3.2 LIWC analysis of off topic modes

The comparisons between the Off Topic and Scaffolding modes along LIWC dimensions were done using a series of paired *t*-tests. The means (values in % of words), standard deviations, *t*-values and effect sizes (Cohen's *d*) of each comparison can be found in Table 3. The analysis was conducted on both tutor (T) and student (S) contributions during the tutoring sessions. These same comparisons were also made using the Wilcoxon's signed-rank test, as a normal distribution of scores cannot be assumed. However, those results very closely mirror the results of the paired *t*-test, and so only the paired *t*-test comparisons are presented here. A Bonferroni correction was not used in this analysis; as this is exploratory research that will be used to orient future research, the authors felt that a conservative correction would result in a loss of critical, if minor, information. In sum, the results here seem to indicate that every significant category difference favored the Off Topic mode, with the exception of when students use ACHIEVEMENT and FUTURE words.

Table 3. Occurrence of LIWC category words

LIWC Category	T/S	Off Topic		Scaffolding		<i>t</i> -value <i>df</i> = 29	Cohen's <i>d</i> (<i>sig</i> only)
		<i>M</i>	<i>SD</i>	<i>M</i>	<i>SD</i>		
<i>Social Process</i>	T	11.15	3.09	7.75	1.66	6.272*	1.37
	S	8.25	4.73	4.87	2.43	4.148*	0.9
<i>Positive Emotion</i>	T	5.41	2.68	4.83	2.43	1.082	
	S	6.54	5.56	4.54	2.34	2.046*	0.47
<i>Insight</i>	T	3.00	1.40	2.06	0.67	3.480*	0.86
	S	2.75	1.75	1.62	0.68	3.029	
<i>Tentative</i>	T	3.10	1.70	1.91	0.64	3.874*	1.08
	S	2.68	2.19	1.60	0.80	2.562*	0.65
<i>Work</i>	T	2.90	2.81	1.10	1.00	3.289*	0.86
	S	2.70	2.70	2.09	3.19	0.801	
<i>Achieve</i>	T	1.02	1.14	0.95	0.55	0.313	
	S	0.52	0.72	1.89	1.98	3.682*	0.92
<i>Leisure</i>	T	0.78	2.74	0.23	0.20	1.115	
	S	0.50	0.95	0.15	0.32	1.859†	0.5
<i>Home</i>	T	0.30	0.68	0.04	0.08	2.013†	0.53
	S	0.24	0.89	0.01	0.03	1.412	
<i>Nonfluencies</i>	T	1.51	1.00	1.11	0.81	1.849†	0.44
	S	3.89	5.97	4.14	2.56	0.236	
<i>Future</i>	T	1.13	0.86	1.23	0.70	0.452	
	S	0.74	1.27	1.35	1.24	2.124*	0.49

† = $p < .1$; * = $p < .05$.

In general, it may be said that there is more to off topic conversation than simple socializing and “time wasting.” Instead, there is a complex dynamic within the Off Topic mode, where the tutor and student are achieving a balance of work-related discussion and subtle socializing. Though off topic conversation seems to be an unlikely place for WORK

words to arise, the Off Topic mode contains significantly more WORK words than does Scaffolding. At first, this seems counterintuitive; however, this difference may be due to the way in which work is talked about in these two different modes. In Scaffolding, work may not be discussed on a superficial level, as this is where work is actually performed. Off Topic may be a place to discuss work on a superficial level, without content. The authors of LIWC list examples of WORK words being things like “class” and “graduate,” so perhaps Off Topic conversation is a place where the student and tutor talk generally about schoolwork and homework. This could be supported by the significantly elevated use of HOME words in Off Topic as opposed to Scaffolding. Perhaps HOME is being discussed in the context of homework. Word categories that would hint at a more social use of HOME words, like FAMILY and FRIEND words, do not significantly differ from Off Topic to Scaffolding. In fact, those two categories make up less than 1% of the words in both modes. This suggests that HOME words are not being used in a social context, but rather, in work-related discussion. This is similar to the results of [8]; in that work, tutors engaged in “social talk” with their students, which involved discussing learning strategies. Our tutors may be doing something similar during their “social talk” (Off Topic). It may be the case that expert tutors use off topic conversation to discuss more general studying strategies.

This may also explain why TENTATIVE words and NONFLUENCIES occur significantly more in Off Topic than they do in Scaffolding on the part of the tutor. Rather than overtly stating problem solving and study suggestions, the tutor may use TENTATIVE words and NONFLUENCIES to lessen the face threatening nature of these suggestions. In one tutoring session, the tutor tells the student that, although the teacher assigned the odd problems for homework, she should work additional problems to get more practice. While this can be portrayed as a mere suggestion, it is alluding to the student’s deficient abilities and need for further practice. Suggesting additional work like this may induce the tutor to use more words like “maybe,” “perhaps,” and “umm”, given that barking orders is unlikely to lead to the completion of this additional work.

Although the LIWC results do suggest the presence of work-related discussion, there is an undeniable socio-emotional factor involved in off topic conversation. Generally, the Off Topic mode contains more POSITIVE EMOTION words than does Scaffolding; this aligns with work by [16], which found that the emotion “happiness” was much more likely to occur with tutor and student off topic conversation than other portions of the tutoring session. These off topic conversations, then, may be used as a sort of short “break” from the tutoring material that restores positive emotion and builds rapport between the tutor and student. This positive emotion and rapport building may act as a buffer against some of the direct, negative feedback that expert tutors give [17]. However, other affective LIWC categories like AFFECTIVE PROCESSES, NEGATIVE EMOTION words, and ANXIETY are not used in significantly different amounts between Off Topic and Scaffolding. This may be indicative of students’ greater comfort in discussing positive emotions, as their negative emotions are likely tied to past and current academic struggles and failures. However, it may instead reflect that the purpose of Off Topic is not to discuss the emotional state of the student during learning. Which may also mean that off topic conversation does not necessarily include “pep talks”, as [8] suggest. In addition, tutors do not use a larger amount of ACHIEVEMENT words Off Topic, suggesting

that they are not trying to overtly bolster students' feelings of confidence. Instead, rapport seems to be built in more subtle ways, such as by using more SOCIAL PROCESSES words like "we" and "us", and perhaps by using higher-level strategies of rapport building like humor and solidarity statements.

4 Conclusion

These two methodologies seem to converge and support our initial casual observations that off topic conversation is more than simply social talk or irrelevant ramblings. Off topic does not seem to be simply an "other" category. We feel that the evidence supports claims that off topic dialogue may serve motivational uses, to discuss more global pedagogy or study skills [8], build rapport [11], and in certain cases serve as a much needed mental break from tutoring.

Exploratory studies often are limited by the use of a single methodology. This study benefits from the use of two distinct approaches to investigating off topic conversation in tutoring. One approach utilized pre-existing coding schemes to determine whether the activities generally assigned to pedagogical conversation are occurring to any degree during off topic conversation. The second approach brings in a new and different analysis of the dialogue. This approach allows for a more objective look at the data, whereas results from the pre-existing coding scheme could be critiqued as simply an artifact of our coding methodology. Given that both approaches showed off topic conversation to be complex and multidimensional, this convergence gives support to further exploration. These results have given an improved depiction of what occurs during off topic conversation; however, we are still only able to speculate on its anatomy. Future research will reveal the true role and importance of off topic conversation in tutoring.

These findings have afforded a framework to begin future, more directed investigations. First, they have allowed us to determine whether further exploration is even worthwhile. These exploratory findings suggest that further investigation is, in fact, warranted. Second, the findings here can be used as the basis for future coding schemes. Whether a manual coding scheme or an automated methodology, such as probabilistic topic models [19], is employed, either can be used to determine the proportion of off topic conversation that is dedicated to global pedagogy, building rapport, social topics, and possibly even irrelevant ramblings. Through the accurate depiction of off topic conversation, its most advantageous features can be applied to the building of ITSs.

While implementing off topic conversation into ITSs under the current conception of discussing social topics seems peculiar, incorporating those pedagogical and rapport-building dimensions of off topic conversation would be a useful addition. So while an ITS may never form a deep, meaningful social bond with a student, it could help to increase learning in a broader scope than simply the present topic. ITSs such as MetaTutor already incorporate strategies similar to our proposed global pedagogy during learning [20-22]. By incorporating the global pedagogy of expert tutors as well as more local pedagogical strategies, ITSs can give greater aid to struggling students in over many disciplines. Further analysis will be needed to know the exact nature of off topic conversation and its potential usefulness in building ITS systems.

Acknowledgements. This research was supported by a grant awarded by the U. S. Office of Naval Research (N00014-05-1-0241) and the Institute of Education Sciences (R305A080594). 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 Office of Naval Research or the Institute of Education Sciences.

References

- [1] Bloom, B.S. The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational Researcher*, 1984, 13, p. 4-16.
- [2] Cohen, P.A., Kulik, J.A., & Kulik, C.C. Educational outcomes of tutoring: A meta-analysis of findings. *American Educational Research Journal*, 1982, 19, p. 237-248.
- [3] Corbett, A., Anderson, J., Graesser, A., Koedinger, K., & van Lehn, K. Third generation computer tutors: Learn from or ignore human tutors? *Proceedings of the 1999 Conference of Computer-Human Interaction*, 1999, p. 85-86.
- [4] Cade, W.L., Copeland, J.L., Person, N.K., and D'Mello, S.K. Dialogue modes in expert tutoring. *Proceedings of the 9th International Conference on Intelligent Tutoring Systems*, 2008, p. 470-479.
- [5] D'Mello, S.K., Person, N., & Lehman, B.A. Antecedent-consequent relationships and cyclical patterns between affective states and problem solving outcomes. *Proceedings of 14th International Conference on Artificial Intelligence In Education*, 2009, p. 57-64.
- [6] Lepper, M.R., Woolverton, M., Mumme, D.L., & Gurtner, J. Motivational techniques of expert human tutors: Lessons for the design of computer-based tutors. In Lajoie, S.P., & S.J. Derry, S.J. (Eds.), *Computers as cognitive tools*, 1993, p. 75-105. Hillsdale, NJ: Lawrence Erlbaum Associates.
- [7] Lepper, M.R., & Chabay, R.W. *Socializing the intelligent tutor: Bringing empathy to computer tutors*, 1988. New York, NY: Springer-Verlag.
- [8] Rosé, C.P., Kumar, R., Aleven, V., Robinson, A., & Wu, C. CycleTalk: Data driven design of support for simulation based learning. *International Journal of Artificial Intelligence in Education*, 2006, 16, 195-223
- [9] Baker, R.S., Corbett, A.T., Koedinger, K.R., & Wagner, A.Z. Off-task behavior in the cognitive tutor classroom: When students "game the system". In *Proceedings of ACM CHI 2004: Computer-Human Interaction*, 2004, p. 383-390.
- [10] Rowe, J., McQuiggan, S. Robison, J., & Lester, J. Off-task behavior in narrative-centered learning environments. In *Proceedings of the 14th International Conference on Artificial Intelligence and Education*, 2009, p. 99-106.
- [11] Catt, S., Miller, D. & Schallenkamp, K. You are the key: Communicate for learning effectiveness. *Education*, 2007, 127(3), 369-377.

- [12] Ramsden, P. Student learning and perceptions of the academic environment. *Higher Education*, 1979, 8, 411-428.
- [13] Cromley, J.G., & Azevedo, R. What do reading tutors do? A naturalistic study of more and less experienced tutors in reading. *Discourse Processes*, 2005, 40(2), 83-113.
- [14] Graesser, A.C., D'Mello, S.K., & Person, N.K. Meta-cognition, meta-communication, and meta-affect in tutoring. *Presented at the annual meeting of the American Educational Research Association*, 2009, San Diego, CA.
- [15] Pennebaker, J.W., Francis, M.E., & Booth, R.J. *Linguistic Inquiry and Word Count (LIWC): LIWC2001*, 2001. Mahwah, NJ: Lawrence Erlbaum Associates.
- [16] Lehman, B.A., Matthews, M., D'Mello, S.K., and Person, N. What are you feeling? Investigating student affective states during expert human tutoring sessions. *In Proceedings of the Ninth International Conference in Intelligent Tutoring Systems*, 2008, p. 50-59.
- [17] Person, N.K., Lehman, B.A., & Ozburn, R. Pedagogical and motivational dialogue moves used by expert tutors. *Presented at the 17th Annual Meeting of the Society for Text and Discourse*, 2007, Glasgow, Scotland.
- [18] Cohen, J. A power primer. *Psychological Bulletin*, 1992, 112(1), 155-159.
- [19] Steyvers, M., & Griffiths, T. Probabilistic topic models. In Landauer, T., McNamara, D., Dennis, S., & Kintsch, W. (Eds.), *Latent semantic analysis: A road to meaning*, 2007. Mahwah: Erlbaum.
- [20] Azevedo, R., Witherspoon, A.M., Graesser, A., McNamara, D., Rus, V., Cai, Z., et al. MetaTutor: An adaptive hypermedia system for training and fostering self-regulated learning about complex science topics. *Paper to be presented at a Symposium on ITSs with Agents at the Annual Meeting of the Society for Computers in Psychology*, 2008, Chicago, IL.
- [21] Azevedo, R., Witherspoon, A., Graesser, A., McNamara, D., Chauncey, A., Siler, E., et al. (2009). MetaTutor: Analyzing self-regulated learning in a tutoring system for biology. In Dimitrova, V., Mizoguchi, R., du Boulay, B., & Graesser, A. (Eds.), *Building learning systems that care: From knowledge representation to affective modeling*, 2009, p. 635-637. Amsterdam: IOS Press.
- [22] Azevedo, R., Witherspoon, A., Chauncey, A., Burkett, C., & Fike, A. MetaTutor: A MetaCognitive tool for enhancing self-regulated learning. *Paper presented at the Annual Meeting of the American Association for Artificial Intelligence, Symposium on Metacognitive and Cognitive Educational Systems*, 2009, Washington, DC.

Sentiment Analysis in Student Experiences of Learning

Sunghwan Mac Kim and Rafael A. Calvo

skim1871@uni.sydney.edu.au, rafa@ee.usyd.edu.au

School of Electrical and Information Engineering, University of Sydney

Abstract. In this paper we present an evaluation of new techniques for automatically detecting sentiment polarity (*Positive* or *Negative*) in the students responses to Unit of Study Evaluations (USE). The study compares categorical model and dimensional model making use of five emotion categories: *Anger*, *Fear*, *Joy*, *Sadness*, and *Surprise*. *Joy* and *Surprise* are taken as a *Positive* polarity, whereas *Anger*, *Fear* and *Sadness* belong to *Negative* polarity in the binary classes, respectively. We evaluate the performances of category-based and dimension-based emotion prediction models on the 2,940 textual responses. In the former model, WordNet-Affect is used as a linguistic lexical resource and two dimensionality reduction techniques are evaluated: Latent Semantic Analysis (LSA) and Non-negative Matrix Factorization (NMF). In the latter model, ANEW (Affective Norm for English Words), a normative database with affective terms, is employed. Despite using generic emotion categories and no syntactical analysis, NMF-based categorical model and dimensional model result in better performances above the baseline.

1 Introduction

Universities are increasingly interested in using quality measures that provide evidence that can be used for benchmarking and funding decisions. For this reason, questionnaires such as the Unit of Study Evaluation (USE) [1], or Students Evaluations of Teaching (SET) as they are called in the USA, have been developed as a means of collecting data from students on their experience of learning at the individual subject or unit of study level (these terms are interchangeable and used as synonymously in this study)

Reviews of the literature of student evaluations of teaching show the massive amount of evidence collected using these standard instruments [2, 3]. According to Marsh [2], SET are the most studied form of personnel evaluation. Most of the studies look into quantitative measures gathered in the questionnaires. This paper utilizes the textual open-ended responses that are also collected. Scholarly literature generally agrees on the validity, reliability, dimensionality and actual usefulness of this kind of data. Despite this standardized evaluations have been highly controversial for decades. Arguably because University faculty have normally no formal training in teaching, so those mechanisms that are used for assessing teaching effectiveness are threatening. Staffs are not generally aware of the above mentioned literature, or when they are, they are often skeptic about its meaningfulness.

Despite its criticisms USE and SET are increasingly used by academics, institutions and the students themselves. They reflect valuable aspects of the student experience that can complement other forms of feedback from students to academics and institutions. One of the obstacles is that reading and making sense of all the textual responses can be a daunting task. This paper aims at a combined analysis of the textual and quantitative

responses using novel data mining techniques in order to provide a more comprehensive understanding of the student experience.

Sentiment analysis [4] attempts to automatically identify and recognize opinions and emotions in text. One goal of sentiment classification is to determine whether a text is *objective* or *subjective*, or represents a *positive* or *negative* opinion, affect classification is to identify the expressions of emotion such as *happiness*, *sadness*, *anger*, etc. In the area of sentiment classification, most research has been built around corpora of users' reviews (e.g. movie reviews) that contain a rating system (e.g. number of stars) and a textual description, a data structure similar to the one in the USE used in this study. These reviews are also subjective and contain information about the user experience of the product.

This paper contributes a novel approach to study USE and potentially other descriptions of students' experience. It analyses a corpus of 909 student questionnaires containing 3,353 (raw) textual responses, and explores the feasibility of automatic approaches to making sense of this data. The combination of text and ratings invites the use of sentiment analysis techniques focusing on the valence of opinions (*positive* vs. *negative*), but a richer exploration of student experience would include the affective aspects of the experience beyond its valence. In fact, as it is shown in our study the rating and textual descriptions do not always coincide, possibly highlighting factors in the experience that only surface from the text. The paper explores techniques that could be used for this richer analysis.

The first goal of this paper is to evaluate the feasibility of using sentiment analysis to study the textual responses in USE, an aspect of the data normally sidelined by the ratings. The second goal is to evaluate the merits of two conceptualizations of emotions (*categorical model* and *dimensional model*) on this data.

The rest of the paper is organized as follows: Section 2 presents representative research of the two emotion models used to capture the sentiment of a text. We also describe the affect classifications utilizing the linguistic lexical resources. In Section 3 we will go over the sentiment dataset which comes from USEs. Section 4 provides comparison results from experiments, before coming to our discussion in Section 5.

2 Background

2.1 Emotion Models

There are largely two models for representing emotions: the *categorical model* and *dimensional model*. The categorical model assumes that there are distinct emotional categories such as Ekman's six basic emotions -*anger*, *disgust*, *fear*, *joy*, *sadness*, and *surprise*- [5]. These have been used in many studies despite not necessarily appearing in specific practical scenarios like in teaching. The advantage of such representation is that it represents human emotions intuitively with easy to understand emotion labels. ITS researchers have used other than Ekman's categories to group emotions that appear in

student-system dialogues. D’Mello [6] proposed five categories (boredom, confusion, delight, flow, and frustration) for describing the affect states in ITS interactions.

A second approach where core affects can be represented in a dimensional form [7] represents emotions in a 2 or 3 dimensional space. A valence dimension indicates *positive* and *negative* emotions on different ends of the scale. The arousal dimension differentiates *excited* vs. *calm* states. Sometimes a third, dominance dimension is used to differentiate if the subject feels in control of the situation or not.

The categorical model and the dimensional model have two different methods for estimating the actual emotional states of a person. In the former, a person is required to choose one emotion out of an emotion set that represents the best feeling. On the other hand, the latter exploits rating scales for each dimension like Self Assessment Manikin (SAM) [8], which consists of pictures of manikins, to estimate the degree of valence, arousal, and dominance.

2.2 Categorical Classification with WordNet-Affect

WordNet-Affect [9] is an affective lexical repository of words referring to emotional states. WordNet-Affect extends WordNet by assigning a variety of affect labels to a subset of synsets representing affective concepts in WordNet. In addition, WordNet-Affect has an additional hierarchy of affective domain labels. There are publicly available lists relevant to the six basic emotion categories extracted from WordNet-Affect and we used five lists of emotional words among them for our experiment.

In addition to WordNet-Affect, we exploited Vector Space Model (VSM) in which textual documents can be represented through term-by-document matrix. In general, both terms and documents are encoded as vectors in the reduced k -dimensional space. We take into consideration log-entropy with respect to a *tf-idf* weighting schema.

The vector-based representation enables all the contextual information such as words, sentences, and synsets to be represented in a unifying way with vectors. VSM provides a variety of similarity mechanisms between two vectors. In particular, we take advantage of cosine angle between an input vector (input sentence) and an emotional vector (emotional synsets) as similarity measures to identify which emotion the sentence connotes. This can be done in the reduced LSA or NMF representation. We also entail the predetermined threshold ($t = 0.65$) for the purpose of validating a strong emotional analogy between two vectors [10]. If the cosine similarity does not exceed the threshold, the input sentence is labeled as “*neutral*”, the absence of emotion. Otherwise, it is labeled with one emotion associated with the closest emotional vector having the highest similarity value. If we define the similarity between a given input text, I , and emotional class, E_j , as $\text{sim}(I, E_j)$, the categorical classification result is more formally represented as follows:

$$\text{CCR}(I) = \begin{cases} \arg \max_j (\text{sim}(I, E_j)) & \text{if } \text{sim}(I, E_j) \geq t \\ \text{"neutral"} & \text{if } \text{sim}(I, E_j) < t \end{cases}$$

One class with the maximum index is selected as the final emotion class.

Two statistical dimensionality reduction methods (LSA and NMF) are utilized in a category-based emotion model for the purpose of reducing dimensions in VSM. Graesser and colleagues [11] used Latent Semantic Analysis (LSA) for detecting utterance types and affect in students' dialogue within Autotutor.

Latent Semantic Analysis (LSA) [12] is the earliest model that has been successfully applied to various text manipulation areas. The main idea of LSA is to map terms or sentences into a vector space of reduced dimensionality that is the latent semantic space. The mapping of the given term/sentence vectors to this space is based on singular vector decomposition (SVD). It is known that SVD is a reliable tool available for matrix decomposition. It can decompose a matrix as the product of three matrices. The columns of one of three matrices represent the coordinates for documents in the latent space. Therefore, we make use of the columns in order to compute sentence similarities.

Non-negative Matrix Factorization (NMF) [13] has been successfully applied to semantic analysis. Given a non-negative matrix A , NMF finds non-negative factors W and H that are reduced-dimensional matrices. The product WH can be regarded as a compressed form of the data in A . This non-negative peculiarity is desirable for handling text data that always require non-negativity constraints. The classification of sentences is performed based on the columns of matrix H that represent the sentences.

2.3 Dimensional Estimation with ANEW

ANEW [14] is a set of normative emotional ratings for collections of words (1,035 words) in English, which means that it provides emotional dimensions. This collection provides the rated values for valence, arousal, and dominance for each word that are rated by means of the Self Assessment Manikin (SAM). For each word w , the normative database provides coordinates \bar{w} in an affective space as:

$$\bar{w} = (\textit{valence}, \textit{arousal}, \textit{dominance}) = \textit{ANEW}(w)$$

Therefore, it is possible to accomplish the mapping of contextual information into the 3-dimensional emotion space through ANEW dictionary. For example, words or sentences are scattered all over the emotional plane.

As a counterpart to the categorical classification above, this approach assumes that an input sentence pertains to an emotion based on the least distance between each other on the Valence-Arousal-Dominance (VAD) space. The input sentence consists of a number of words and the VAD value of this sentence is computed by averaging the VAD values of the words. A series of synonyms from WordNet-Affect are used in order to calculate the position of each emotion. These emotional synsets are converted to the 3-dimensional VAD space and averaged for the purpose of producing a single point for the target emotion.

$$\overline{sentence} = \frac{\sum_{i=1}^n \bar{w}}{n}, \overline{emotion} = \frac{\sum_{i=1}^k \bar{w}}{k}$$

where \bar{w} is the VAD value of a word, and n and k denote the total number of words in an input sentence and synonyms in an emotion, respectively. *Anger*, *fear*, *joy*, and *sadness* emotions are mapped on the VAD space. Let $Anger_c$, $Fear_c$, Joy_c , Sad_c , and $Surprise_c$ be the centroids of five emotions. Then the centroids, which are calculated by the above equation, are as follows: $Anger_c = (2.55, 6.60, 5.05)$, $Fear_c = (3.20, 5.92, 3.60)$, $Joy_c = (7.40, 5.73, 6.20)$, $Sad_c = (3.15, 4.56, 4.00)$, and $Surprise_c = (5.23, 5.33, 4.70)$. Apart from the five emotions, we manually define *neutral* to be (5, 5, 5). If the centroid of an input sentence ($\overline{sentence}$) is the most approximate to that of an emotion ($\overline{emotion}$), the sentence is tagged as the emotion. We define the distance threshold (empirically set to 4) to validate the appropriate proximity like categorical classification.

3 Unit of Study Evaluation (USE) Data

The Unit of Study Evaluation (USE) questionnaire has 12 questions, 8 of which are standardized University-wide and 4 that are selected by each Faculty. It is designed to provide information to those seeking a) to assess the learning effectiveness of a subject, for planning and implementing changes in the learning and teaching environments, and b) to assess the contributions of units or subjects to students' learning experience in their whole degree program, as monitored by the CEQ. The USE in our study contains 12 statements:

1. The learning outcomes and expected standards of this unit of study were clear to me.
2. The teaching in this unit of study helped me to learn effectively.
3. This unit of study helped me develop valuable graduate attributes.
4. The workload in this unit of study was too high.
5. The assessment in this unit of study allowed me to demonstrate what I had understood.
6. I can see the relevance of this unit of study to my degree.
7. It was clear to me that the staff in this unit of study were responsive to student feedback.
8. My prior learning adequately prepared me to do this unit of study.
9. The learning and teaching interaction helped me to learn in this unit of study.
10. My learning of this unit of study was supported by the faculty infrastructure.
11. I could understand the teaching staff clearly when they explained.
12. Overall I was satisfied with the quality of this unit of study.

Eleven items (I1-I11) focus on students' experience and one item (I12) on student satisfaction. Students indicate the extent of their agreement with each statement based on a 5 - point Likert scale: 1 - strongly disagree, 2 - disagree, 3 - neutral, 4 - agree and 5 - strongly agree. Below each statement there is a space requesting students to explain their response. Question 4 has a different sentiment structure therefore was removed in this study.

The USEs of subjects taught by two academics collected over a period of six years were used to create the dataset. After removing responses to question 4, the dataset contains a total of 909 questionnaires (each with 11 ratings), and out of the possible 9,999, students responded with 3,008 textual responses (each expected to be a description of a rating), a textual response rate of 30.1 %. Out of these we removed internal referencing (e.g. ‘see above’) and meaningless text (e.g. ‘?’).

The textual data has two characteristics that may significantly affect the classifiers. First the sentences are hand-written in an informal style, containing spelling errors, abbreviated non-dictionary words or hard to read text. The lack of proper grammar would make it extremely challenging to use part-of-speech (POS) tagging or other computational linguistic approaches. Examples include: “Computers in labs too slowk no lecture notes” (spelling mistakes and non-grammar), “tutes were overcrowded, stopping teacher / student interaction” (non-standard words). For these reasons, the techniques used in the experiment are based on the bag-of-words assumption (so word order is not used) and we do not use POS tagging that would require relatively correct grammar.

Table 1. Number of comments and sample comments for each sentiment

Rating	Number	Sentiment	Number	Comments tagged with each sentiment
Strongly Agree	381	Positive	1,455	lecturer and tutor was helpful and explained concepts well.
Agree	1,074			
Neutral	611	Neutral	611	It is a bit clear about staff response but need more examples in there answer.
Disagree	571	Negative	874	Not enough computers to accommodate all the students.
Strongly Disagree	303			

4 Experiments and Results

The following five different approaches are implemented in Matlab. One categorical model that has two variants, according to three corresponding methods of dimension reduction, one dimensional method, and two similarity comparison methods for each model are implemented. For evaluation purposes, we employ Majority Class Baseline (MCB) as our baseline and Keyword Spotting (KWS). We remove stop words and use stemming. Text to Matrix Generator (TMG), a Matlab toolkit [15], is used to generate term-by-sentence Matrix.

- Majority Class Baseline (MCB): classification that always predicts the majority class, which in this dataset is *Positive* across all sentiment classifications.
- Keyword Spotting (KWS): a naïve approach that counts the presence of obvious affect words like “frustrating” and “satisfaction”, which are extracted from WordNet-Affect for five emotion categories.
- CLSA: LSA-based categorical classification
- CNMF: NMF-based categorical classification
- DIM: Dimension-based estimation

Five emotion categories are utilized (*Anger, Fear, Joy, Sadness, and Surprise*) in which *Joy* and *Surprise* emotions are assigned to *positive* class while *Anger, Fear, and Sadness* are the members of *negative* class, respectively. Negative emotion, *disgust*, is removed because the emotion is similar to *anger* and leads to making sentiment classes biased. Likewise, *strongly agree* and *agree* belong to *positive*, and *strongly disagree* and *disagree* are referred to *negative*. The number of sentences for each rating and sentiment used in our experiment is shown in Table 1. In addition, sample comments of the annotated corpus appear in Table 1.

Table 2 shows the precision, recall, and F-measure values obtained by the five approaches for the automatic classification of three sentiments. The highest results are marked in bold for each individual class. We do not include accuracy values in our results due to the imbalanced categories (see Table 1). The accuracy metric does not provide adequate information, whereas precision, recall, and F-measure can effectively evaluate the classification performance with respect to imbalanced datasets [16].

Table 2. Sentiment identification results

Sentiment	Positive			Negative			Neutral		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
MCB	0.495	1.000	0.662	0.000	0.000	-	0.000	0.000	-
KWS	0.527	0.220	0.310	0.270	0.061	0.099	0.212	0.743	0.330
CLSA	0.575	0.362	0.445	0.388	0.203	0.266	0.218	0.560	0.314
CNMF	0.505	0.897	0.646	0.378	0.120	0.182	0.421	0.052	0.093
DIM	0.591	0.329	0.423	0.398	0.317	0.353	0.223	0.522	0.312

As can be seen from the table, the performances of each approach depend on each sentiment category. In case of the *positive* class, which has the largest number of sentences, MCB and CNMF get the best sentiment detection performance in terms of recall and F-measure. DIM achieves rather high precision score in comparison with all other classifications. We can see that DIM approach gives the best results for *negative* class. When it comes to *neutral*, KWS shows the best performance with respect to recall and F-measure. On the other hand, CNMF particularly outperforms the others for precision. Figure 1 indicates a result of the 3-dimensional and 2-dimensional attribute evaluation for USEs.

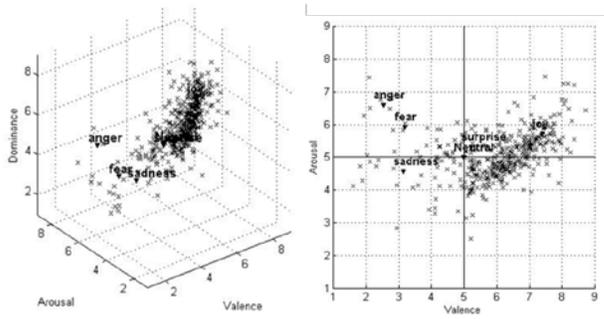


Figure 1. Distribution of the USEs dataset in the 3-dimensional (left) and 2-dimensional (right) sentiment space. The ‘x’ denotes the location of one comment corresponding to valence, arousal, and dominance.

A notable aspect observed in the USE data is that there are somewhat inconsistencies between students' ratings and written responses illustrated with examples in Table 3. For instance, the third row is unambiguously negative but the student graded this sentence as neutral. Therefore, all approaches have a weakness in recognizing sentiments due to the peculiarity of this data. Another factor, which makes the automatic classification difficult, is that all classifiers are not specific to education domains. For this reason, we speculate that this mediocre performance of the methods is owing to poor coverage of the features found in education domains.

Table 3. Sample feedbacks from misclassified results. (Positive values are those rates 4 as 5, neutral as 3 and negative 1 or 2)

Student's feedback	Student rating	System rating
It should be core to software gingerbeering	Positive (5)	Neutral (LSA)
The labs were not long enough with too few tutorials. 4 labs were too few. How about one for FETS/MOSFETS? Given the instruction was for AC/DC components (i.e. lower/uppercase) it was difficult to follow the hadn written notes on the overhead. Maybe print it all up?	Positive (4)	Negative (NMF)
We never got personal feedback.	Neutral (3)	Negative (DIM)
Hi my name is ABC, I like this LECTURER_NAME, I mean this course!!	Negative (2)	Positive (NMF)

Table 4 shows overall precision, recall, and F-measure comparison with respect to MCB, KWS, CLSA, CNMF, and DIM in two averaging perspectives (micro-averaging and macro-averaging). The notable difference between these to calculate is that micro-averaging gives equal weight to every sentence whereas macro-averaging weights equally all the categories. From this summarized table, we can see that MCB, KWS, and CLSA perform less effectively with a little low number of evaluation scores compared with CNMF and DIM. In case of macro-averaging, CNMF is superior to other classifications in precision, while DIM surpasses the others in recall and F-measure. On the other hand, DIM has the best precision and CNMF performs better for F-measure in micro averaging. Overall, CNMF and DIM vie with each other in precision, recall and F-measure and the best F-measure is obtained with the approach based on CNMF or DIM for each average. Our KWS conducted in all experiments is inferior to CNMF, DIM as well as CLSA. The result implies that keyword spotting techniques cannot handle the sentences which evoke strong emotions through underlying meaning rather than affect keywords. In addition, we can infer that the models (CNMF and DIM) with non-negative factors are appropriate for dealing with text collections. In summary, NMF-based categorical model and dimensional model shows the better sentiment recognition performance as a whole.

The most frequent words used by students to describe aspects of their experience, include terms such as *labs*, *lecturer*, *lectures*, *students*, *tutors*, *subject*, and *work*. When we remove these terms, the words most frequently used to describe positive experiences include: *good* ($n=263$), *helpful* and *helped* ($n=183$), *online* ($n=79$), *understand* ($n=49$). Those used to describe negative experiences include: *hard* ($n=72$), *understand* ($n=67$),

time ($n=47$). Neutral experiences contain a combination of both. These words lists are obtained from CNMF and DIM because two classifications have better overall performance as aforementioned. Stemming was not used for this analysis since in this particular corpus it might hide important differences as between ‘lecturer’ and ‘lecture’.

Table 4. Overall average results

Mean	Micro			Macro		
	Prec.	Rec.	F1	Prec.	Rec.	F1
MCB	0.245	0.495	0.328	0.165	0.333	0.221
KWS	0.385	0.281	0.325	0.337	0.341	0.247
CLSA	0.445	0.356	0.396	0.394	0.375	0.342
CNMF	0.450	0.490	0.469	0.434	0.356	0.307
DIM	0.457	0.366	0.406	0.404	0.389	0.363

5 Discussion

This paper described a dataset of ratings and textual responses of student evaluations of teaching. Sentiment analysis techniques for automatically rating textual responses as *positive*, *negative* or *neutral* using the students’ ratings were evaluated. In particular, the performance of categorical model and dimensional model were compared, each of which makes use of different linguistic resources.

This paper highlighted that NMF-based categorical and dimensional models have a better performance than the others. Moreover, despite not having an appropriate set of emotional categories to use, the efficacy of two emotion lexicons (WordNet-Affect and ANEW) promises to be useful in these sentiment classification tasks.

While two models and two lexicons are promising for identifying sentiments, there are still challenges to overcome. We believe that affective expressivity of text is on the basis of more complex linguistic features such as morphological features. Hence, we are going to delve into Natural Language Processing (NLP) to recognize fine-grained emotion in the future.

Future work will include extending the corpora with more student evaluations and this should provide more reliable results. The categorical model should be evaluated with a set of emotion categories better grounded in the educational research literature and we suspect that the literature on motivation would be particularly useful. With regards to the use of normative databases to study the dimensional model, we are aware that the terms in ANEW are not the best suited for the vocabulary that students use to describe their experiences, but we are not aware of other more appropriate databases.

Acknowledgement

This project was partially funded by a TIES grant from the University of Sydney.

References

- [1] ITL. *Purpose of the USE* 2008 [cited 2009 February 10]; Available from: <http://www.itl.usyd.edu.au/use/purpose.htm>.
- [2] Marsh, H.W., Students' evaluations of University teaching: Research findings, methodological issues, and directions for future research. *International Journal of Educational Research*, 1987. **11**(3): p. 253-388.
- [3] Richardson, J.T.E., *Instruments for obtaining student feedback: a review of the literature*. *Assessment & Evaluation in Higher Education*, 2005. **30**(4): p. 387-415.
- [4] Pang, B. and L. Lee, *Opinion mining and sentiment analysis*. *Foundations and Trends in Information Retrieval*, 2008. **2**(1-2): p. 1-135.
- [5] Ekman, P., *An Argument for Basic Emotions*. 1992. p. 200.
- [6] D'Mello, S., R. Picard, and A. Graesser, *Toward an Affect-Sensitive AutoTutor*. *IEEE Intelligent Systems*, 2007. **22**(4): p. 53-61.
- [7] Russell, J.A., *Core affect and the psychological construction of emotion*. *Psychological Review*, 2003. **110**(1): p. [Washington, etc.] American Psychological Association [etc.]--172.
- [8] Lang and P.J. (1980). "Behavioral treatment and bio-behavioral assessment: Computer applications." *Technology in mental health care delivery systems*: 119-137.
- [9] Strapparava, C. and R. Mihalcea. Learning to identify emotions in text. in *Proceedings of the 2008 ACM symposium on Applied computing*. 2008.
- [10] Penumatsa, P., M. Ventura, et al. (2006). The Right Threshold Value: What Is the Right Threshold of Cosine Measure When Using Latent Semantic Analysis for Evaluating Student Answers? *International Journal on Artificial Intelligence Tools*, WORLD SCIENTIFIC PUBLISHING.
- [11] D'Mello, S., et al. Automatic Detection of Learner's Affect from Conversational Cues. in *User Modeling and User-Adapted Interaction*. 2008.
- [12] Landauer, T.K., et al., eds. *Handbook of Latent Semantic Analysis*. 2007, Routledge.
- [13] Lee, D.D. and H.S. Seung, *Learning the parts of objects by non-negative matrix factorization*. *Nature*, 1999. **401**(6755): p. 788-791.
- [14] Bradley, M. M. and P. J. Lang (1999). "Affective norms for English words (ANEW): Instruction manual and affective ratings." University of Florida: The Center for Research in Psychophysiology.
- [15] Zeimpekis, D. and E. Gallopoulos, *TMG: A MATLAB toolbox for generating term-document matrices from text collections*. *Grouping multidimensional data: Recent advances in clustering*, 2005: p. 187-210.
- [16] He, H. and E.A. Garcia, *Learning from Imbalanced Data*. *IEEE Transactions on Knowledge and Data Engineering*, 2009. **21**(9): p. 1263.

Online Curriculum Planning Behavior of Teachers

Keith E. Maull¹, Manuel Gerardo Saldivar^{2,3} and Tamara Sumner^{1,3}
{keith.mauull, manuel.saldivar, tamara.sumner}@colorado.edu

¹Computer Science Department, University of Colorado, Boulder

²School of Education, University of Colorado, Boulder

³Institute of Cognitive Science, University of Colorado, Boulder

Abstract. Curriculum planning is perhaps one of the most important tasks teachers must perform before instruction. While this task is facilitated by a wealth of existing online tools and resources, teachers are increasingly overwhelmed with finding, adapting and aligning relevant resources that support them in their planning. Consequently, ripe research opportunities exist to study and understand online planning behavior in order to more generally characterize planning behavior. In this paper, we introduce a web-based curriculum planning tool and study its use by middle and high school Earth science teachers. We examine the web analytics component of the tool and apply clustering algorithms to model and discover patterns of the use within the system. Our initial results provide insights into the use of the tool over time and indicate teachers are engaging in behavior that show affinity for the use of interactive digital resources as well as social sharing behaviors. These results show tremendous promise in developing teacher-centric analysis techniques to improve planning technologies and techniques to study online curriculum planning patterns.

1 Introduction

A large body of research suggests that teachers are the single most important influence on students' academic achievement [1]. Thus, helping teachers do their jobs better should lead to improved student outcomes. One of the specific pedagogical techniques now being demanded of many K-12 teachers is *differentiated instruction*. Differentiated instruction involves the customization of curriculum and teaching practices to better foster student understanding of course material [17]. An example of differentiated instruction would be a teacher who uses animations and graphic images to impart a science concept to his students because he has discovered his largely immigrant population of students struggles with English reading comprehension. Given the increasing racial, ethnic, and linguistic diversity in American K-12 schools, differentiated instruction is becoming more important than ever precisely because a 'one size fits all' approach to teaching cannot reach all of today's diverse student body.

At the same time that differentiated instruction is being stressed in K-12, the Internet is changing the educational landscape to a degree not seen since the introduction of personal computers in the 1980s [11]. As Internet connectivity becomes available in more and more classrooms, cloud-based applications and databases have become increasingly important both for teachers' instructional practices and students' learning [3]. Our study lay at the intersection of these two important trends – the drive to differentiate instruction and the widening availability of the Internet in educational contexts.

One of the challenges of differentiating instruction is developing supplementary educational materials that target specific students' learning needs [17]. That is, given a common curriculum, *this* group of students may require additional visual aids to help them grasp a concept while *that* group of students might benefit most from an extra hands-on

activity that reinforces a lesson. The Internet provides a portal to a nearly infinite set of digital resources that could help teachers in their differentiation of instruction, but the unmanaged nature of the Internet places the burden of filtering and evaluating digital resources on teachers, adding to their already significant workload. If this filtering and evaluation process could be at least partially automated, teachers would be able to focus on teaching rather than on preparing to teach.

In this paper, we examine two research dimensions related to the process of differentiating instruction: (1) learning science for online curriculum planning and (2) educational informatics for application usage pattern detection and analysis. Since curriculum planning is a task which every teacher must perform before instruction, we begin by describing a Web-based application that is designed to help teachers review curricular objectives, locate relevant supplementary digital resources, and develop differentiated instructional plans that connect their curricular goals and digital materials with classroom activities and assessments.

The first research dimension is framed by a few key research questions : (1) What are the behavioral components of Web-based curricular planning? (2) How do these behaviors play out as patterns in online curricular planning? (3) What techniques are used to measure the effectiveness of curriculum planning behavior? (4) How can or do online tools and resources shape curriculum planning behavior? (5) How do online tools and resources impact curriculum planning outcomes? Not all of these questions will be covered within the scope of this study.

With an application context to study the learning science component of curriculum planning, our second research interest is focused on developing and applying tools and techniques for observing and classifying teachers' online behavior in educational applications. This research offers a unique view into the online usage patterns and behaviors of educators by examining the use of a curriculum planning application in the web mining and analytics context. Specifically, we observe and analyze the use of system features and functionality as well as commonly used resources available within the system to gather a more complete understanding of system use. Using web analytics and clustering algorithms, we develop the education informatics dimension of this research, framed by the following questions : (1) What computational tools can be used to discover and model the online behavior patterns of teachers engaged in curriculum planning? (2) What computational tools can be used to predict the online behavior of teachers once this behavior is modeled? (3) What techniques can be used to maximize teacher's use of online tools and resources?

2 Research Context

While much of the online learning science research to date tends to focus on students' behaviors vis-à-vis computer-assisted learning [12], little research has been done to understand teacher behavior in online curriculum planning tasks. The online behaviors of teachers performing planning tasks online may hold useful clues to the development of applications that not only improve student outcomes, but also teacher outcomes, particularly as they relate to improving teacher access to and use of digital materials within the

classroom instruction and learning context. This research presupposes that teachers' on-line activities are important in their own right and worthy of study.

Two broad contexts are applied to the research : the user context and application context. The focal point of the user context, described in more detail below, is middle and high school Earth science teachers. The corresponding application context is a web-based curriculum planning application called the Curriculum Customization Service (CCS) [15], designed to support those same middle and high school teachers in curriculum planning tasks. The application was designed to provide general support for accessing, searching, browsing, storing, sharing and reviewing curriculum goals, objectives, guidelines, materials and resources for the middle and high school Earth science curriculum.

2.1 User Context

The users of the CCS are Earth science teachers at the 6th and 9th grade levels within a medium sized urban school district. Nearly 120 sixth and ninth grade Earth science teachers within the school district were invited (but not required) to use the CCS program during the 2009-2010 school year. In July 2009, a four-hour face-to-face training session (and a Web-based teleconference for those who could not attend) was carried out to demonstrate the features and use of the CCS. User account access to the CCS was provided during the remainder of the summer so that teachers could further acquaint themselves with the tool and learn more about its features before the start of the semester. Users represent a wide cross section of experience, planning skill, technological ability and interests, with user teaching experience spanned from first year teacher to more than 30 years of experience. No information was collected before the study window about who would choose to participate or who would plan to fully use the tool during the semester, though a survey was given to teachers before the semester to probe their familiarity with technology within the classroom and initial perceptions of the CCS tool during the training sessions. The survey also provided further insights about the teachers' technology-related skills and perceptions, as well as demographic profile information.

2.2 Application Context

The CCS application is a Web-based application that provides access to the specific content of the 6th and 9th grade Earth science curriculum of the participating school district (see Figure 1). The application has several unique features that target teacher-centric behaviors around curriculum planning and customization, specifically planning, storing and sharing, and searching. First, the CCS encompasses the entirety of the district-wide Earth science curriculum – that is it contains all the curriculum units, goals and objectives for the district-wide 6th and 9th grade Earth science curriculum.

Traditional curriculum materials that were typically provided in paper form are instantly accessible digitally through the CCS. Within a single environment, teachers also can access critical planning artifacts : all of the Earth science materials for their lesson planning, including digital access to publisher materials (e.g. textbook content, assessments), curriculum learning goals and objectives, as well as relevant standards and concept hierarchies. Second, the CCS provides instant access to a number of digital resources that

are automatically coupled with relevant curriculum unit topics. Teachers accessing unit-level goals have access to pre-fetched digital resources that match the topics of the unit. This feature displays the most relevant and useful materials for quick access, circumventing the need for custom (and potentially time-consuming) searches to find the same materials. Third, a customizable space to save resources of interest, called “*My Stuff*,” is provided to encourage teachers to privately save what they may consider to be useful resources. This feature also has the ability to make available to other CCS users the resources a teacher may think his or her colleagues might also find useful. Therefore, within the same environment as planning tasks, teachers can engage fully in customization activities : finding and saving materials of value, sharing and contributing those materials to the participating teachers at large, rating, tagging and even uploading arbitrary materials they wish to store for later use within the planning environment or share with other users. Finally, since searching for contextualized materials is a difficult task within generic search engines (e.g. Google, Yahoo!, Bing), a customized search engine was provided within the application that returns concept-relevant digital resources obtained from search results against the Digital Library for Earth Systems Education (DLESE), a high quality and highly regarded Earth systems digital library [16].

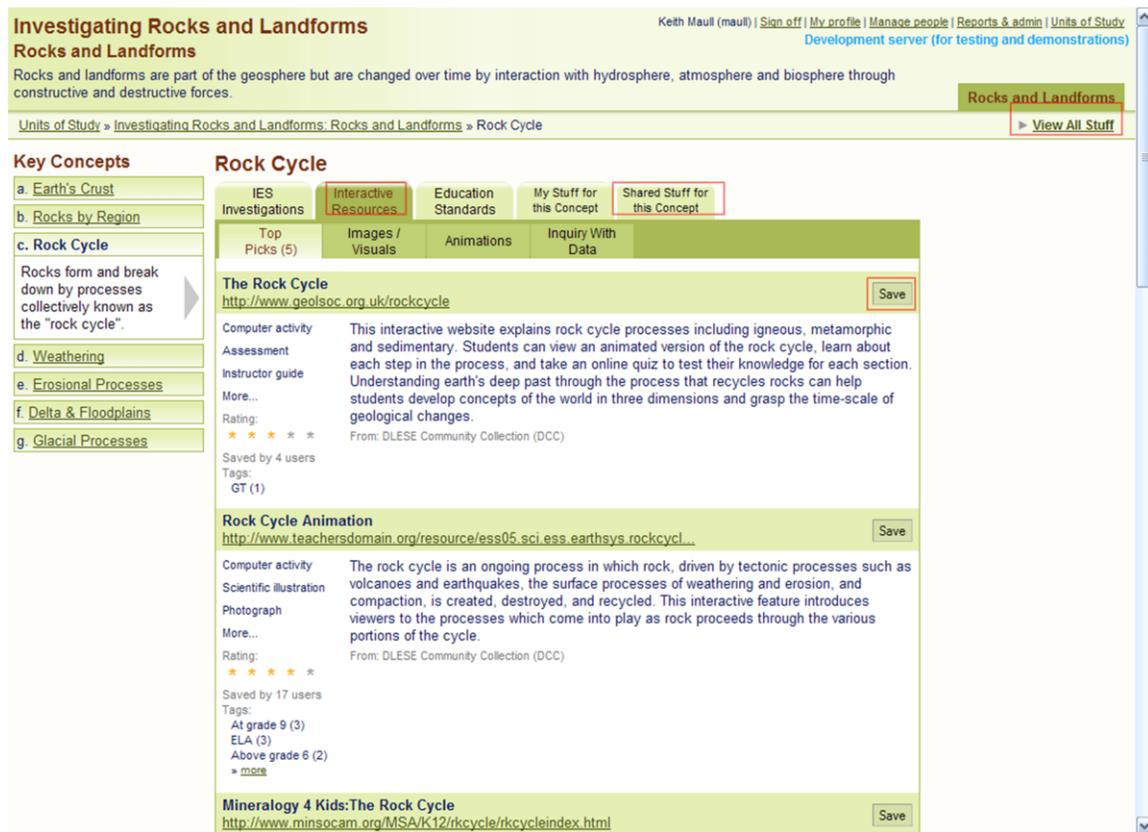


Figure 1: The Curriculum Customization Service User Interface
(A few sample features are in highlighted in red)

The CCS provides a single entry point for teachers to efficiently develop and execute their most important curriculum planning and customization tasks, and as such provides a powerful research platform for several key reasons : (1) it is bounded and constrained to

the most frequently used curriculum planning task space used by teachers, thus providing a single place to examine teachers' behavior around this task, (2) the user pool is constrained to a diverse but specific group of users, middle and high school Earth science teachers, so relationships between these users will be more amenable to analysis, (3) the subject area of analysis is narrowed to a middle and high school Earth science curriculum focus, thus amplifying inter-user and inter-group comparison and analysis, and (4) the nature of web-based application activity analysis is well understood and supported by techniques that are robust and effective.

3 Methodology

The goal of our methodology is to understand and prepare the required data to run experiments that allow us to detect patterns of use for further evaluation. The research methodology of this initial study has four components : (1) select and prepare an initial data set, (2) select an experimental feature set, (3) perform clustering experiments, (4) analyze experimental results. Our research draws on widely understood techniques for web server log analysis [11] of the CCS application. While it is not difficult to determine system use frequency, such as commonly visited pages or resources, frequent use alone provides a narrow view of actual use. System components are often used in concert with one another, thus providing different views of system use that may reveal unexpected or unusual relationships. We therefore turn to clustering algorithms to help build connections among the CCS system components that make up the experimental feature set.

3.1 Data Set and Data Filtering

Data for the initial set of experiments in this study was derived from the server logs of the core CCS system. The CCS system has a number of unique client-side scripting features that are not usually captured in server logs because there is typically no direct server interaction during such activity. This client-side activity was captured by instrumentation that stored such activity as if it were direct server interaction. All users of the system were given unique login IDs that were captured during session activity over a 16 week period from August 2009 to December 2009.

Allowing for the broad use of the CCS, session duration was defined to accommodate a complete working day of activity. Sessions longer than 10 hours or shorter than 30 seconds with fewer than 4 actions were eliminated from the data set. While the minimum session length may seem short, there were some sessions detected that lasted for abbreviated periods of time but nonetheless showed meaningful activity. Logging in to the system to access a singular, specific resource is one such example. Other filtering activities involved removing data with user IDs of non-teacher users not part of the research study.

3.2 Experimental Features

Feature selection for data mining is a difficult task and requires many considerations to be both valuable and effective [12]. For the initial research experiments, the user session was abstracted and examined for features to be used in the clustering experiments detailed in section 5. In all, 27 features were selected for our initial feature set.

Web sessions are usually seen as a temporal ordered stream of clicks (click stream). However, for our initial research these sessions were simplified into a larger grained *bag of clicks*. Instead of examining the transitions from one click to another, we assign and examine click types within the system for a given session. This *bag of clicks* model simplifies log processing and feature selection and borrows similar intuitions found in *bag of words* models in linguistic processing, where word frequency is used to analyze document content, sometimes arriving at similar conclusions as temporal, structural or semantic analysis with far less complexity.

Each client-side UI click has one of 5 associated *click types* which our initial experiments exploit as a feature. Furthermore, since click stream data is ultimately important in understanding session behavior, the *number of clicks* within a session was also selected as a feature. It would not be unreasonable to expect sessions with high click counts to represent sessions of interest. In addition to the number of clicks, *session duration* or the amount of time (in seconds) a user spends interacting with the system, was chosen as a feature.

Other data was also selected as part of the initial feature set selection process. Since the CCS is a highly visual and largely client-side interaction environment, commonly used visual controls of the environment were natural targets for the experimental feature set. This visual features set was narrowed to the *top 20 visual features* of the interface, which was computed by flattening the visual hierarchy of the interface controls and examining the global use frequencies of each of these visual components.

User activity can be measured in many ways and for the purpose of this study, we make note of several server log data. First, *total actions analyzed* provides an indication of how many discrete actions were logged from the server, *discrete sessions* details the total number of sessions in an interval (monthly, weekly, etc.), and *unique user accounts* shows the number of unique user accounts in a period. The total number of actions over the 16 week data collection period was 17,527 in 1,370 total sessions over 82 unique user accounts, detailed in Table 1.

Table 1 : Monthly data summary

<i>Month</i>	<i>Total Actions Analyzed</i>	<i>Discrete Sessions</i>	<i>Unique User Accounts</i>
September	7,371	526	82
October	3,626	331	65
November	4,698	337	60
December	1,832	176	50

Table 2 : Cluster ranks and relative size (%)

<i>Rank</i>	<i>K12</i>	<i>EM12</i>	<i>EM*</i>
1	58%	47%	37%
2	14%	17%	26%
3	8%	11%	11%
4	8%	5%	10%
5	5%	5%	–
6	4%	–	–

4 Algorithms and Feature Analysis

Clustering algorithms are commonly used to find patterns within large data sets [2, 6, 7] and two clustering algorithms were chosen to study the initial data set. First, the *K*-means clustering algorithm was used. *K*-means is an unsupervised learning, iterative descent al-

gorithm that partitions n data observations into K clusters. Each cluster is assigned a centroid and cluster membership is determined by minimizing the distance from each cluster member and the centroid. The second algorithm in the initial experiment was the Expectation-Maximization (EM) algorithm [4]. EM is a model-based iterative algorithm that examines data observations and represents each cluster as a probability distribution. Given n data observations, EM maximizes the likelihood of the observed distributions by estimating the means and standard deviations of each cluster.

Neither algorithm is without flaws and our experimental results in section 6 show this. K -means primary weakness is that the number of clusters must be determined *a priori*. This weakness, however, is inherent in many partitioning clustering algorithms and may require an experimentally selected n . Another weakness is that K -means is sensitive to outliers – data that are distant from the centroid may pull the centroid away from the real centroid in a given data set. Finally, it is difficult to understand which feature contributes more to cluster membership, since every feature is assumed to have the same weights. EM’s core weaknesses are the relative speed with which the clusters converge, and the possibility of convergence at the boundary of a cluster.

5 Experiments

Two experiments were performed on the initial feature set with a few variations for comparison. The first experiment was designed to run the K -means algorithm with $n = 12$ and the Euclidean distance function, referred to as K12. The n for this initial experiment was derived from the total number of sessions analyzed over the period (~1,400) divided by the number of users invited to participate (~120). This provides a baseline for comparison with the other algorithms. For comparison, the expectation maximization algorithm was chosen for the remainder of the experiments. EM was first chosen to automatically select n clusters using cross validation, referred to as EM*. This provided a baseline to compare the algorithm’s performance against the K -means algorithm (K12). The last experiment was run using EM again with a fixed cluster size of 12, referred to as EM12.

6 Evaluation and Results

Table 2 shows each of the algorithms and the sizes of the largest clusters they produced. EM* produced 10 clusters, the top 4 of which represent 84% of all the data. Similarly, for EM12, the top 5 largest clusters represent 84% of the data. Finally, K12 shows a similar trend, with the top 5 clusters representing 87% of the data. For the purposes of evaluation we consider the top 4 clusters in EM*, top 5 in EM12, and the top 6 clusters in K12, since the K12 distribution of clusters was more sparse.

Table 3 shows the features with the greatest means of the top clusters for each algorithm. The cluster labels represent UI features for example, A1 represents clicks on the *Interactive Resources* tab, A2 the *Shared Stuff for This Concept* tab, A6 for the *Embedded Assessments* toggle element, A12 for the *Images/Visuals* tab, and so on. The top features of the largest cluster in EM* (A1, A2 and A4) correspond to CCS UI tab clicks on *Interactive Resources*, *Shared Stuff for This Concept* and *Shared Stuff for This Activity*. This top cluster suggests a pattern of activity that is focused on both CCS-suggested interactive

resources *and* shared resources that others have saved, which may indicate the importance of what *others* have saved as well the automatically generated interactive resource list.

The EM12 and K12 algorithms indicate very similar patterns. For example, EM12’s largest cluster shows the exact same pattern as EM*. Similarly, K12 shows A1, A3 and A4 as its top features. Examining other clusters show cluster 4 of K12 and cluster 11 of EM12 share similar patterns over features A3, A6, A8 and A11. This pattern corresponds to clicks on *Instructional Support Materials, Embedded Assessments, Answers* and *Teaching Tips* system areas. Similarly, cluster 2 of K12 and cluster 1 of EM12 share similar features along A3, A7 and A14, which correspond to the *Instructional Support Materials* and *Activities* tabs, suggesting time being spent on preparing or reviewing student activities and corresponding materials.

Table 3 : Top cluster features and their cluster membership

	Cluster #	A1	A2	A3	A4	A5	A6	A7	A8	A11	A12	A13	A14	A16
K12	1 (8%)	■									■			
	2 (5%)			■				■					■	
	3 (14%)	■	■				■							
	4 (4%)			■			■		■	■				
	8 (4%)	■												■
	9 (58%)	■		■	■		■							
EM*	0 (26%)	■	■			■						■		
	1 (11%)				■			■					■	
	6 (10%)	■				■		■					■	
	7 (37%)	■	■		■									
EM12	1 (5%)			■	■			■					■	
	3 (5%)	■	■			■								■
	6 (17%)	■				■		■					■	
	7 (47%)	■	■		■									
	11 (11%)			■			■		■	■				

Each cluster algorithm revealed data that was consistent with overall system use seen in the server logs, though the smaller cluster sizes show greater differentiation of features. That there was not complete agreement in cluster features or sizes, however, may indicate more experiments are required.

7 Related Work

Much of the work here has been influenced by the body of work in web use analytics, which break down into two categories : (1) content analytics and (2) usage analytics [12]. This work is focused on usage analytics. Broadly, use analytics aims at understanding the aggregate activity and use patterns of a website primarily using advanced server log analysis. Such analytics often aim at understanding aspects of the site that are popular, content that seems to be frequently accessed, times of frequent/infrequent use, etc. with the goal of developing a sense of where the site could be improved or enhanced for optimal

performance, increased advertisement penetration or site content enhancement through recommender techniques [13]. Such use analytics are invaluable for developing site content, but also useful in developing models of user behavior. Website session characteristics are commonly studied to determine how users are accessing the site and statistical techniques are used to determine tasks being performed within a website, revealing cluster usage patterns in the ways we have discussed here. Markov models have been used to derive the meaning of certain behaviors within a session by observing page transitions and their probabilities to develop behavioral models of use [5]. Work has also been done to connect page semantics to web usage, for example [9] use Probabilistic Latent Semantic Analysis to determine if the content and subsequent usage of a page implies an underlying task. Finally, user interface event mining [8] aims at developing techniques to exploit detailed user experience and interaction data.

8 Discussion

The initial experiments presented in this paper offer some insights into the planning behavior of teachers online. However, two areas of improvement can be immediately discussed : improved feature selection and expanded algorithm experiments and comparison. The initial experimental feature set provides interesting insights into the behavior of teachers for the visual components selected in this initial observation. However, the flattened visual hierarchy of the CCS interface only provides a convenient way to discretize each visual element of the system without advancing the notion of the *semantic* structure of this hierarchy. For example, while it is clear that the *Interactive Resources* tab of the interface was widely used, there are substructures under that tab which also contain widely used features. The current feature set is not capable of capturing this hierarchy or its implied semantic structure, though considering it might yield new insights into the semantics of the features commonly accessed by users. Further extensions to the feature set might also include adding link-to-link features, for example, exploring high frequency transitions might reveal unique relationships between UI features and functionality.

The EM and K -means algorithms are commonly used in data mining, and while some clusters in K12 and EM12 had similar characteristics, all of the top clusters were not similar enough to say both algorithms were converging on *exactly* the same feature sets. This may underscore the differences in each algorithm or in the way they each treat the features. It may also reinforce the effects parameter sensitivity (e.g. n clusters) and feature selections have on the results. The focus of the next round of experiments will be to experiment further with EM and K -means parameters, and also to expand algorithm coverage to hierarchical-based algorithms. Such experimentation may also fit well with the semantic features already suggested and allow a comparison of the hierarchies that are produced from a semantic-based structure with the clusters already observed.

As with all learning algorithms, it is challenging to determining if the experimental data would be predicted by and hold up to some gold standard or human expert evaluation. Determining if the discovered behaviors match the observed data in practice is difficult and further research is underway to study actual and reported system use through on-site observation and survey instruments, which should lead to a higher fidelity confirmation of the patterns discovered thus far.

References

- [1] Brophy, J. and Good, T.L., Teacher behavior and student achievement. In M.C. Wittrock (Ed.) *Handbook of research on teaching*. McMillan : New York, 1986.
- [2] Cios, K.J., Pedrycz, W., Swiniarski, R.W. and Kurgan, L.A. *Data mining: A knowledge discovery approach*. Springer Verlag, 2007.
- [3] Collins, A. and Halverson, R. *Rethinking education in the age of technology: The digital revolution and the schools*, 2009.
- [4] Dempster, A.P., Laird, N.M., Rubin, D.B., et al. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.
- [5] Deshpande, M. and Karypis, G. Selective Markov models for predicting web page accesses. *ACM Transactions on Internet Technology (TOIT)*, 4(2):163–184, 2004.
- [6] Frank, E. and Witten, I.H. *Data Mining: Practical machine learning tools and techniques with java implementations*. Morgan Kaufmann, 2005.
- [7] Han, J. and Kamber, M. *Data mining: Concepts and techniques*. Morgan Kaufmann, 2006.
- [8] Hilbert, D. M. and Redmiles, D. F. Extracting usability information from user interface events. *ACM Computing Survey*, 32(4):384–421, 2000.
- [9] Jin, X., Zhou, Y., and Mobasher, B. Web usage mining based on probabilistic latent semantic analysis. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 197–205. ACM New York, NY, USA, 2004.
- [10] Leu Jr, D.J., Leu, D.D., and Len, K.R. *Teaching with the internet: Lessons from the classroom*. Christopher-Gordon Publishers, Inc., 480 Washington St., Norwood, MA 02062, 1997.
- [11] Liu, B. *Web data mining: Exploring hyperlinks, contents, and usage data*. Springer, 2007.
- [12] Liu, H. and Motoda, H. *Feature selection for knowledge discovery and data mining*. Springer, 1998.
- [13] Mobasher, B. Data mining for web personalization. *Lecture Notes in Computer Science*, 4321:90, 2007.
- [14] Smith, R., Clark, T. and Blomeyer, R.L. A synthesis of new research on K-12 online learning. Naperville, IL: North Central Regional Educational Laboratory. 2005.
- [15] Sumner, T., Devaul, H., Davis, L. and Weatherley, J. A curriculum customization service. In *Proceedings of the 2009 Joint International Conference on Digital libraries*, pages 479–480. ACM New York, NY, USA, 2009.
- [16] Sumner, T., Khoo, M., Recker, M. and Marlino, M. Understanding educator perceptions of "quality" in digital libraries. In *Proceedings of the 3rd ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 269–279. IEEE Computer Society Washington, DC, USA, 2003.
- [17] Tomlinson, C.A. and McTighe, J. Integrating differentiated instruction and understanding by design: Connecting content and kids. Alexandria, VA: Association for Supervision and Curriculum Development, 2006.

A Data Model to Ease Analysis and Mining of Educational Data¹

André Krüger^{1,2}, Agathe Merceron² and Benjamin Wolf²
{akrueger, merceron, bwolf}@beuth-hochschule.de
¹Aroline AG, Berlin, Germany
²Beuth University of Applied Sciences, Berlin, Germany

Abstract. Learning software is not designed for data analysis and mining. Because usage data is not stored in a systematic way, its thorough analysis requires long and tedious preprocessing. In this contribution we first present a data model to structure data stored by Learning Management Systems (LMS). Then we give an overview of the system architecture that performs the structure/export functionality and of its implementation for the Moodle LMS. Finally, we show first results using this data model for analysing usage data.

1 Introduction

It is a well known fact that data understanding and preprocessing constitutes the main work of the data analysis and mining process. Learning software is not designed for data analysis and mining. Even if many learning software do store usage data, they are designed to support learning and teaching, not to analyse the data they store. However, the field of educational data mining is emerging precisely because valuable pedagogical information is gained from analysing and mining data stored by educational software [17, 1]. As a consequence, the process of educational data mining requires long and tedious preprocessing as mentioned in various works, see [2, 13, 20] for a few examples.

In this contribution we present a data model to structure and export usage data stored by learning software. Note that separating the data used for the business, in our case the data stored by the learning system, from the data used for analysis and mining is well in the line of the usual approach in the data mining field, see for example [4]. The aim of this data model is to automate, at least partly, the usual long and tedious preprocessing and to facilitate the data exploration step that should precede data mining. Our data model is primarily oriented towards the particular educational software called Learning Management Systems (LMS). We have designed a modular and extensible architecture to realise the structure/export function and we present an implementation for the particular LMS Moodle [14]. Our system handles data that have been rendered anonymous. We show first results using this data model for analysis of usage data.

1.1 Background and Related Works

In our own university and in most universities at least in Germany, LMS are used both in distance education and face to face teaching. They are becoming a must-have in distance education [18]. They are more and more used in face to face teaching as they make the administration of a course much easier for teachers, and provide a handy way to cater for special needs. Experienced teachers can handle better a diverse classroom as

¹ This work is partially supported by the European Social Fund for the Berlin state.

supplementary learning materials can easily be made available through them for example. LMS provide a virtual place where students can find learning materials, study guides as well as an overview of their results, and where they can communicate with teachers and with other students. The functionality of an LMS can be divided in three main parts: Management of learning resources, management of users, and communication between users [18]. However statistics and reports are usually basic.

To illustrate this last point, we list below a few questions that reporting facilities of most of the currently used LMS cannot handle:

- 1.How many students have never viewed Learning Resource A?
- 2.If students do well on activity B, do they also do well on activity C?
- 3.If students solve exercise D, do they also solve exercise E?
- 4.What is the average mark on quiz G got by students who have viewed resource F?
- 5.Which courses use a lot of Audio Learning Resources?

To manage properly learners, especially distance learners, it is quite important to gain a good overview of their learning behaviours. To manage properly degrees, it is important to harmonize the different modules offered in a degree. The aim of our work is to complement LMS in all aspects dealing with data analysis and mining.

Many works to analyse and mine data stored by LMS have been undertaken, see [17, 1] for some overview. To the best of our knowledge, all these works do some ad hoc preprocessing of the data. They do not aim at proposing some data model for analysis and mining that could be shared by all LMS independently of their internal structure. The work undertaken in [16] bears similarity with our work in the sense that it considers all data stored by an institution in higher education and unifies it into a model to explore behaviours of students. Another work that bears similarities to our work is PSLC Datashop [9], an open repository of educational data. The analysis tools offered in Datashop are more suited for educational software such as intelligent tutoring systems. Our work is not concerned with discovering or sharing learning objects as other works such as [19, 7] do. It is concerned with describing and structuring the interactions of users with learning objects as stored in LMS. Our vocabulary to describe these interactions is partly borrowed from the vocabulary adopted by the LMS Moodle [14], as Moodle is used by a large community worldwide. The part related to the interactions of users with quizzes contains elements that the IMS specification [8] also contains, though it is much simpler than [8].

This paper is organised as follows. The following section introduces our data model. The third section gives an overview of the export tool that exports the data stored by an LMS into this data model. Section 4 presents a case study in analysing data stored in a specific course using our tool. Last section concludes this paper.

2 The Data Model

The data model we present is quite close to a fact constellation schema [4]. It contains three kinds of tables: tables to describe objects found in LMS, these tables can be seen as dimension tables; tables to describe interactions with learning objects, these tables can be

seen as fact tables; and third, association tables to describe associations between objects. The choice of having a table per object comes from the observation that most LMS have a limited set of objects that teachers are used to handle. Adopting this same set should make it easier for teachers to analyse usage of resources by students.

2.1 The schema

Our data model makes several general assumptions that we expose now. First we assume that an LMS contains users and courses. Users can enroll or sign in courses and sign off courses. Users can have roles like “lecturer”, “administrator”, “tutor”, “student” and so on. A user may have different roles in different courses. For example a user can be a tutor in the course “Introduction to Programming” and a student in the course “Early American History”. An LMS may contain groups that are associated to courses. Students enroll in those groups. An LMS contains forums, wikis, resources and quizzes. We call a quiz any kind of assignment, exercise or test a lecturer may wish to give to students. Forums, wikis, resources and quizzes are associated to courses. Thus a resource for example, can be used in several courses. A quiz may contain one or more questions that are also contained in the LMS. Questions are associated to quizzes and a given question can be associated to several quizzes. These general assumptions cover the particular case of LMS where resources, forums, wikis or quizzes exist only inside a given course. In this particular case an association table contains only one tuple. We assume that an LMS logs or stores interactions of users. For any given interaction, the LMS stores the identification of the user, of the course, of the resource, forum, wiki, quiz, as well as the timestamp, the nature of the interaction (“view”, “modification”, “creation”, “attempt”, “submit” and so on), the marks and the contribution when relevant.

We present now in more details the tables that are the most useful for our case study, see Figure 1. For the current full set of tables, we refer to [11, 12]. Every table contains an element *id*, which is the key or identifier of the tuple.

The five tables below describe objects usually found in LMS.

Table user: This table describes users registered in the LMS. The elements *firstaccess* and *lastaccess* are the times and dates when a user first and last accessed any kind of learning object, such as a resource, a quiz etc. in the LMS. The elements *lastlogin* and *currentlogin* are the times and dates a user logged in the LMS for the last time, respectively currently. Note that a user can log in without accessing any learning object.

Table course: This table describes courses existing in the LMS. We assume that a course exists for a given period of time. The element *timecreated* is the date and time the course has been created, usually by the administrator. The element *startdate* is the time and date the course is supposed to start, this time is usually fixed by the lecturer in charge. The element *enrolstart* is the date users are allowed to enroll in this course, and the element *enrolend* is the date users can not enroll anymore in the course. The element *timemodified* is the time and date this course has been last modified. The elements *title* and *shortname* are self explanatory.

Table quiz: This table describes quizzes existing in the LMS. As already mentioned, we call a quiz any kind of assignment or test a lecturer gives to students. The element *qtype* is the type of the quiz. It can take values such as “assignment”, “SCORM” and so on,

according to the different kinds of quizzes an LMS makes available. The elements *qid* combined with *type* make up the identification of a quiz. A quiz may contain one or more questions, see table **question**. The element *title* is a title the lecturer in charge gives to this quiz. The elements *timeopen* and *timeclose* refer to the dates and times students are allowed to answer the quiz, while the element *timecreated* is the date and time the quiz has been created and the element *timemodified* is the date and time the quiz has been last modified.

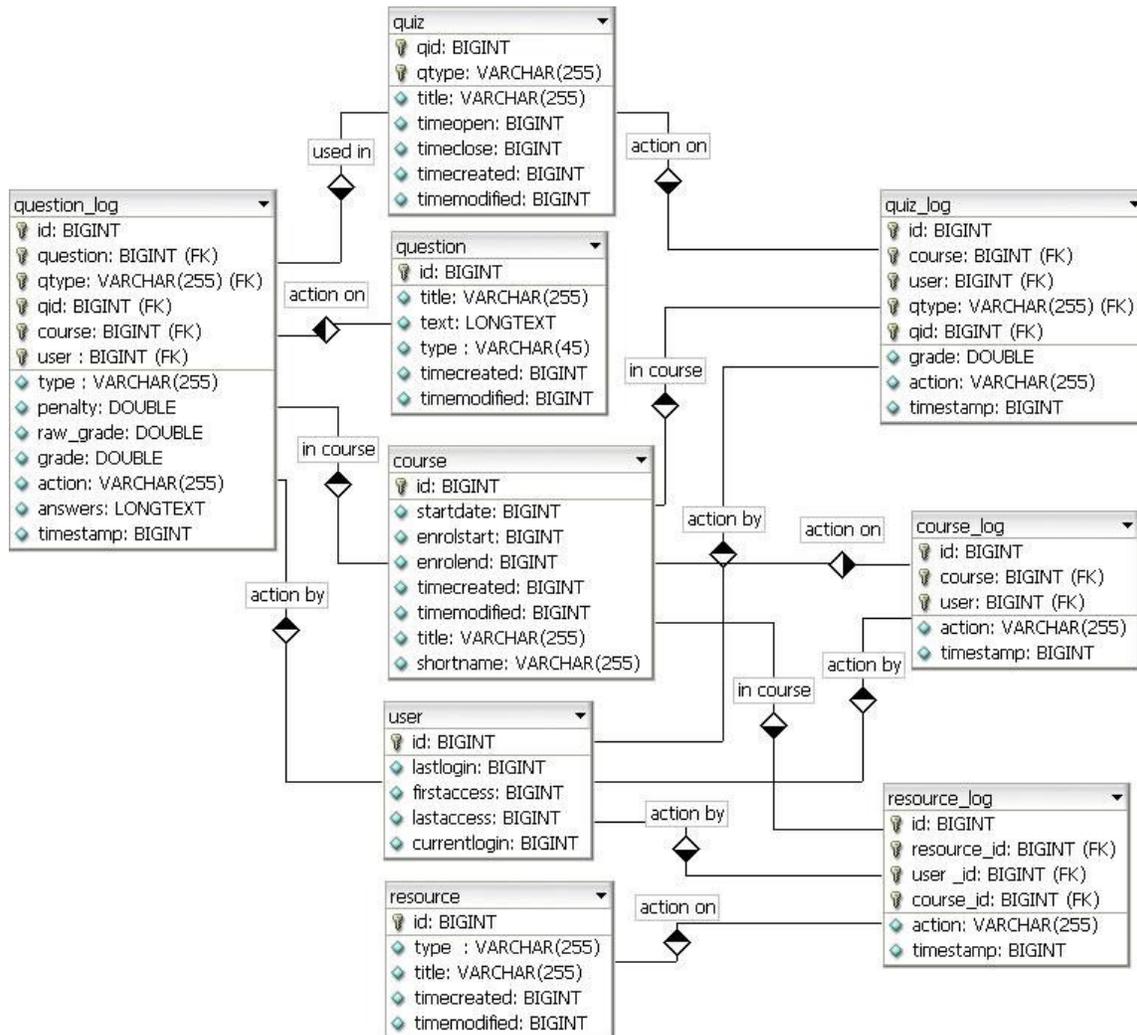


Figure 1. Snapshot of the relational schema

Table question: This table describes questions that make up quizzes. The element *title* is the title of the question, while the element *text* is the actual text of the problem to solve. The element *type* is a category like “multiple-choice”, “true-false” etc. The elements *timecreated* and *timemodified* are as described in the table quiz.

Table resource: This table describes resources available in the LMS that lecturers may use in courses. The element *type* describes the type of the resource like “file”, “uri”, “directory”, “audio”, “picture” and so on. The elements *timecreated* and *timemodified* are as for quiz. The element *title* is the title of this resource like “transparencies01”.

The three following tables describe interactions with learning objects. They are the facts that are stored while users use objects of an LMS.

Table quiz_log: This table describes the information that a LMS stores when users interact with quizzes. The element *user* is the *id* (the key) of the user who interacted. The element *course* is the *id* (the key) of the course in which the interaction took place. The elements *qid* and *qtype* refer to the quiz that was tackled. The element *grade* is the mark obtained in the quiz. The element *timestamp* gives the date and time of the interaction. The element *action* gives the kind of action that took place. An action can be “view”, in that case the user simply looked at the quiz, “attempt”, in that case the user attempted the quiz, “submit”, in that case the user attempted and finished the quiz, “modify” if the quiz has been modified etc. .

Table question_log: This table describes the information that a LMS stores when users interact with a question of a quiz, and contains all elements already included in the table **quiz_log**. The element *penalty* gives the penalty marks given in that interaction. If a quiz is run in adaptive mode then a student is allowed to try again the question after a wrong answer. In this case one may want to impose a penalty for each wrong answer to be subtracted from the final mark for the question. The amount of penalty is chosen individually for each question when setting up or editing the question. The element *raw_grade* gives the raw mark obtained in that interaction. The element *grade* gives the marks for that question in that interaction when *penalty* has been taken into account. It includes also an element *question*, the *id* of the question that was tackled, the elements *type*, which can take values like “multiple choice”, “true/false” and the element *answers*, the actual answer or answers, when several answers are allowed, given by the user in the interaction.

Table resource_log: This table describes the information that a LMS stores when users interact with resources. This table contains elements that are similar to the ones of table **quiz_log**.

Finally our data model contains a number of association tables to associate objects with each other, see [11,12].

3 System Architecture

Figure 2 presents an overview of the system architecture. The central part of the system is the abstract class “*ExtractAndMap*” that describes and partly implements functionalities concerning data extraction from an LMS and data generation for the data model. To create the present data model with an LMS, what is needed is to implement the abstract *extract* methods according to the features of the LMS. The concrete *save* function can be inherited as is. The system contains an implementation for Moodle. It is implemented in Java, uses the Database Mysql [15] and the persistence framework Hibernate [6].

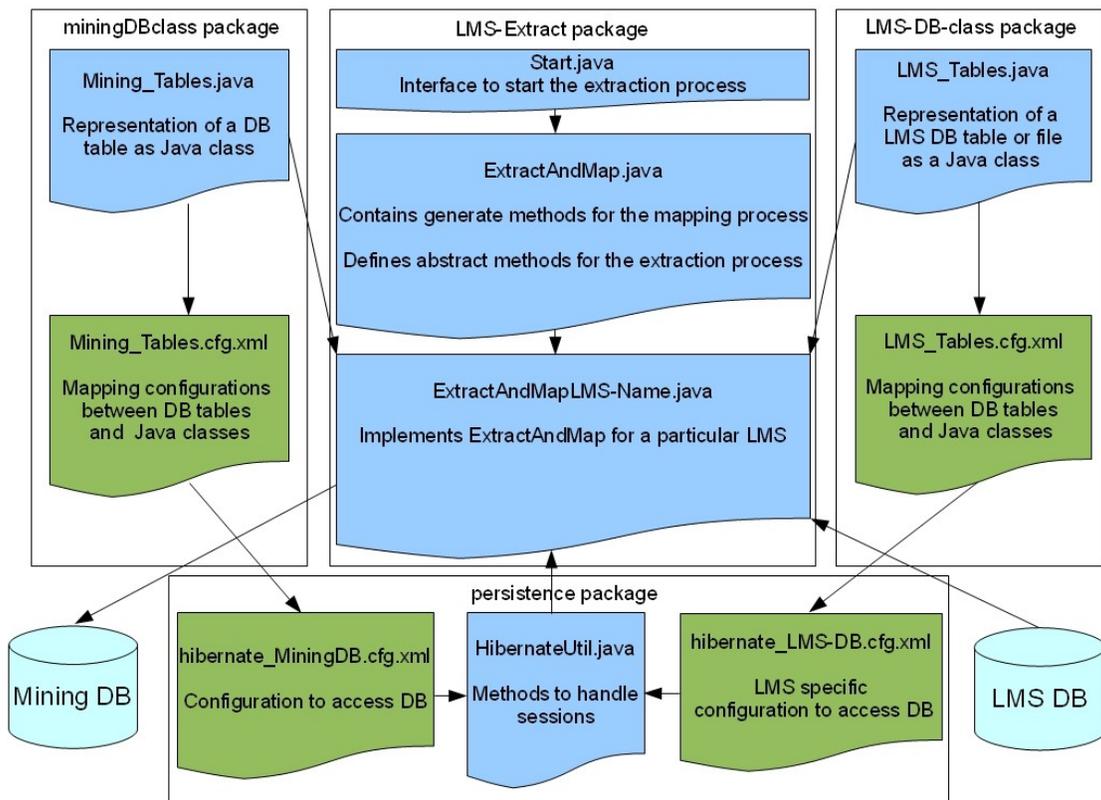


Figure 2. System architecture

4 First Results

We have used our system to analyze the course “Introductory Programming with Java” taught in face to face teaching to first semester students enrolled in the degree “Computer Science and Media” at the Beuth University of Applied Sciences, Berlin, in winter semester 2009/2010. In that semester 65 students were enrolled in this course.

The teaching of this course is supported by the use of the Learning Management System Moodle in which mandatory as well as additional resources are uploaded for students. A list of 8 exercises belongs to the mandatory resources. Students have to solve these 8 exercises to get a mark for the practical part of the course. The lecturer has additionally offered gradually in the semester 7 self-evaluation exercises on key concepts of Java programming like methods, arrays, statements etc.. These exercises are not compulsory. Solving them is left to the sole discretion of the students. The lecturer is interested to know whether students have used these self-evaluation exercises and, most importantly, whether solving or attempting them could have a positive impact on the marks obtained in the final exam. Figure 3, obtained by simple queries, gives an overview of how students have used these exercises. For each exercise the column on the left means *view*, the column in the middle means *attempt* and the column in the right means *close attempt*. Note that *view* means that students have clicked on the resource, *attempt* means that they have submitted a solution and *close attempt* means that they have finished the exercise.

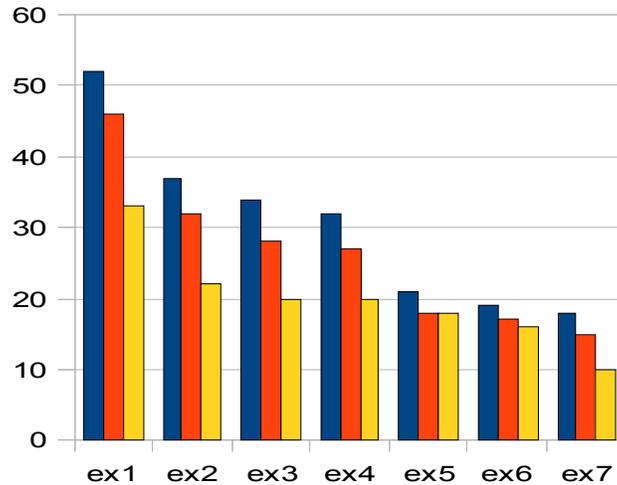


Figure 3. Access to self-evaluation exercises

One notices a pattern that we have already observed in other courses regarding optional self-evaluation exercises [13, 11]: As the semester progresses always less students make use of them. We are interested in investigating whether a dedicated group of students emerges that keep doing the exercises during the semester. Therefore we want to know whether associations such as “if students complete exercise 2, they complete exercise 1” or , “if students complete exercise 3, they complete exercise 2”, and so on, hold. For that we have used the method exposed in [10]. Indeed, the answer is positive as Table 1 shows. The association rule $2 \rightarrow 1$ means “if students complete exercise 2, they complete exercise 1”. All rules have a rather high confidence and are rated as interesting both by lift and cosine. We recall that confidence is a number between 0 and 1 (highest is 1), that lift rates a rule as interesting if its value is above 1, and that cosine rates a rule as interesting if its value is above 0.66. Support gives the proportion of the data involved in the rule.

Table 1. Association rules “if students attempt exercise x, they also attempt exercise x-1”

Associations	$2 \rightarrow 1$	$3 \rightarrow 2$	$4 \rightarrow 3$	$5 \rightarrow 4$	$6 \rightarrow 5$	$7 \rightarrow 6$
support	0.34	0.28	0.25	0.2	0.22	0.14
confidence	1	0.9	0.8	0.72	0.88	0.9
lift	1.97	2.66	2.6	2.35	3.16	3.66
cosine	0.82	0.86	0.8	0.69	0.82	0.71

To investigate whether solving these self-evaluation exercises has a positive impact on the marks in the final exam cannot be achieved by correlation or regression analysis as not so many students have solved them. That means for many students we would have missing data, since 45 students have written the final exam. We have simply queried our

data and looked at the average mark on each group of students. The result is given in Table 2.

Table 2. Completing self-evaluation exercises and marks in the exam.

Exercise	min.	max.	mean	s.deviation	meanNoEx
<i>General</i>	1	13	8.63	4.34	7
<i>Ex1</i>	1	13	9.48	4.06	7.33
<i>Ex2</i>	1	13	9.56	3.97	7.95
<i>Ex3</i>	1	13	9.2	4.4	8.26
<i>Ex4</i>	1	13	8.56	4.77	8.68
<i>Ex5</i>	1	13	9.21	4.54	8.29
<i>Ex6</i>	1	13	10.91	3.4	7.70
<i>Ex7</i>	9	13	11.67	1.41	7.69

The line *General* is the minimum, maximum, mean and standard deviation obtained taking all students who have taken part in the final exam. The last column gives the mean for students who have not solved any exercise. The line *Ex1* gives similar results restricting the population to students who have completed the first self-evaluation exercise. The last column gives the mean for students who have not solved the first self-evaluation exercise. And so on till *Ex7*. One notices that the highest average and smallest standard deviation in the final exam is obtained in the group of students who have completed exercise 7 (11.67 and 1.41 respectively), while smallest average is obtained in the group that has not solved any exercise (7). Given the results of the association rules, students who have completed exercise 7 have most probably completed all optional self-evaluation exercises. These results may speak for a positive impact on the final mark of the self-evaluation exercises. However our small population prevents of making any strong conclusion since statistical tests to check the significance of the difference in the average, like t-test, usually require a sample of size 30 or more. The line for exercise 4 looks different and requires more investigation.

5 Conclusion and Future Work

In this paper we have presented a data model to structure and export the data that most LMS usually store in scattered places into an homogeneous schema. The first aim of our data model is to automate and alleviate the preprocessing that is needed to explore, analyse and mine these data. We have designed an architecture of the tool that does the actual structure/export functionality, and implemented it for the LMS Moodle. Finally we have used our tool to analyse the data stored in the course “Programming 1” in our university. We have focused our analysis on the optional self-evaluation exercises. The analysis shows that, as the semester progresses, less students solve them. It shows also that a group emerges that keeps solving them and that reaches slightly better marks in the final exam.

Another aim of this data model is to couple loosely an LMS and the analysis of the data it stores. If the persistence functionality of an LMS is changed, only the structure/export tool needs to be changed, not the analysis tools linked to the data model. Note that the implementation of the module that performs the export functionality for a concrete LMS has to be programmed with particular consideration regarding performance as the data stored inside an institution can be huge. However this programming happens only once.

As noticed in [5] on-line learning is likely to grow, and so is the use of LMS. As pointed out in [1] the number of works tackling data stored by LMS is increasing. We hope that this work will help boost results and best practices in that area.

We have used this data model mainly to explore thoroughly how students use learning resources in a course. Results of such an exploration are enlightening for teachers and are necessary to conduct a better informed data mining afterwards. Will this model be robust enough to answer any pedagogical question using data mining techniques? It depends on who is asking. Our data model contains all interactions that users perform with any object of the LMS along with the timestamp. Therefore a whole range of pedagogical questions related to navigation, performance prediction, activity of students for instance should be treatable.

We begin to notice recurring questions that users of LMS are interested in. A future work is to continue enhancing and structuring those questions, so that we try out our data model further. We aim also at not restricting our data model to LMS, but consider other learning software as well. Our next step in that direction is to consider learning portals. We are also working on a graphical user interface for users to query and mine the data model in an intuitive way. This interface should work as an adaptive front-end for the user, the real analysis work will be done by connecting suitable queries and mining tool already available. Our work is open for the community and will be released soon on [12].

Acknowledgment We thank warmly our colleague Prof. Dr. Ripphausen-Lippa for her cooperation in analyzing the data of her course “Introductory Programming with Java”.

References

- [1] Baker, S.J.D.R, Yacef, Y. The State of Educational Data Mining in 2009: A Review and Future Visions. *JEDM - Journal of Educational Data Mining*, 2009, 1(1), p. 3-17.
- [2] Dekker, G., Pechenizkiy, M., Vleeshouwers, J. Predicting Students Drop Out: A Case Study. In [3]. p. 41-50.
- [3] Barnes, T., Desmarais, M., Romero, C. & Ventura, S. (Eds.), *Proceedings of the Second International Conference on Educational Data Mining*, 2009. Cordoba, Spain.
- [4] Han, J., Kamber, M. *Data mining: concepts and techniques*, 2006. Morgan Kaufman publishers.
- [5] Hislop G.J. The Inevitability of teaching Online. *Computer, IEEE Computer Society*, 2009, 42(12), p. 94-96.
- [6] Hibernate Relational persistence framework. //www.hibernate.org/, 06.05.2010.

- [7] IMS Global Learning Consortium Learning Object Discovery and Exchange Project Group. <http://www.imsglobal.org/lode.htm>, 06.05.2010.
- [8] IMS Question and Test Interoperability Results Reporting. http://www.imsglobal.org/question/qtiv2p1pd2/imsqti_resultv2p1pd2.html, 25.04.2010.
- [9] Koedinger, K.R., Cunningham, K. A. S., Leber, B.. An open repository and analysis tools for fine-grained, longitudinal learner data. *First International Conference on Educational Data Mining, 2008*. Montreal, Canada, p. 157-166.
- [10] Krueger A., Merceron, A., Wolf, B. When data exploration and data mining meet while analysing usage data of a course. *Third International Conference on Educational Data Mining*, 2010. Pittsburgh, USA.
- [11] Krueger A., Merceron, A., Wolf, B. Leichtere Datenanalyse zur Optimierung der Lehre am Beispiel Moodle. *Proceedings of the 8. e-Learning Fachtagung Informatik*, 2010. Duisburg, Germany: Lecture Notes on Informatics.
- [12] Krueger A., Merceron, A., Wolf, B.. User's Data and Profiles in LMS. <http://learn.beuth-hochschule.de/datamining>, 13.05.2010.
- [13] Merceron, A., Yacef, K. Interestingness Measures for Association Rules in Educational Data. *First International Conference on Educational Data Mining, 2008*. Montreal, Canada, p. 57-66.
- [14] Moodle. Learning Management System. [//moodle.org](http://moodle.org), 06.05.2010.
- [15] MySQL open source database. <http://www.mysql.com/>, 07.05.2010.
- [16] Pechenizkiy, M., Treka, N., Vasilyeva, E., van der Aalst, W., De Bra., P. Process Mining Online Assessment Data. In [3], p. 279-288.
- [17] Romero, C., Ventura, S. Educational Data Mining: A Survey from 1995 to 2005. *Expert Systems with Applications* 33, p. 125-146.
- [18] Schulmeister, R. (2005): *Lernplattformen für das virtuelle Lernen. Evaluation und Didaktik*. 2005. Oldenbourg.
- [19] Verbert K., Wiley, D., Duval, E. A Methodology and Framework for the Semi-automatic Assembly of Learning Objects. *Proceedings of the European Conference on Technology Enhanced Learning EC-TEL*, 2009. p. 757-762 .
- [20] Vialardi Sacin, C., Bravo Agapito, J., Shafti, L., Ortigosa, A. Recommendation in Higher Education Using Data Mining Techniques. In [3], p. 190-199.

Identifying Students' Inquiry Planning Using Machine Learning

Orlando Montalvo², Ryan S.J.d. Baker^{2,1}, Michael A. Sao Pedro¹,
Adam Nakama², and Janice D. Gobert^{2,1}

{amontalvo, rsbaker, mikesp, nakama, jgobert}@wpi.edu

¹Computer Science Department, Worcester Polytechnic Institute

²Social Science and Policy Studies Department, Worcester Polytechnic Institute

Abstract. This research investigates the detection of student meta-cognitive planning processes in real-time using log tracing techniques. We use fine and coarse-grained data distillation, in combination with coarse-grained text replay coding, in order to develop detectors for students' planning of experiments in Science Assistments, an assessment and tutoring system for scientific inquiry. The goal is to recognize student inquiry planning behavior in real-time as the student conducts inquiry in a micro-world; the eventual goal is to provide real-time scaffolding of scientific inquiry.

1 Introduction

Self-regulation is recognized as a highly important aspect of learning [3, 12, 25]. Self-regulation includes planning, meta-cognitive monitoring, reflection, and checking outcomes. While several studies on self-regulation within computer-based learning environments have been conducted [15, 18, 23, 27], there is no consensus about how to automatically measure self-regulation [3, 23]. Furthermore, very few studies have addressed planning within the context of scientific inquiry. Some research has shown that deliberate scaffolding of self-regulation leads to better learning in science [22], but it is difficult to figure out what to measure [15]. Our study seeks to demonstrate a method for detecting one aspect of self-regulation, students' planning in the context of scientific inquiry. Planning is one of the inquiry skills outlined by the National Science Education Standards [21]. Since inquiry problems require several meta-cognitive processes, one of which is planning [11], detecting students' inquiry strategies and skills, including planning, is a critical first step in order to provide students with support in the form of computer-based adaptive scaffolding during real time inquiry [13, 14]. This study brings together research on self-regulation and planning during scientific inquiry.

In this paper, we present a machine-learned model that detects student planning by tracing time spent looking at data tables and hypothesis lists within our inquiry-based learning environment, Science Assistments (http://users.wpi.edu/~sci_assistments; [13, 14]) and microworld for Phase Change. Planning is required especially when applying the control for variables strategy (CVS), a key cognitive strategy within the domain [10], but also in deciding what experiments are needed. We leverage from the success of [6], in using text replays [4] to provide training instances for machine-learned detectors of gaming the system within intelligent tutors. Specifically, by manually inspecting and coding a proportion of the student inquiry sequences using text replay tagging of log files, we extended this approach in order to develop detectors that determine whether a student is planning by viewing data from their previous trials and/or their hypotheses. Our text replay coding approach differs from previous text replays in two ways. First, whereas text replays allow for the classification of a replay clip as a single category (out

of a set of categories), text replay tagging allows multiple tags to be associated with one clip. For example, a clip may be coded as using the data table for planning, using the hypothesis table for planning, both, or neither. Second, the behaviors we are studying are temporally more coarse-grained than in [6], displaying the entire sequence of experimental trials for part of a hypothesis rather than specific trials. In addition to this coding, we summarized each clip by creating a feature set from the action data. In accordance with our coding, we consider problem-level features of the student data rather than step or transaction-level data, unlike in many prior EDM models of student behavior (e.g. [2, 8, 9, 24, 26]). Using the coding and the feature set, we created detectors for planning.

2 Learning Environment

Our phase change environment (Figure 1), hosted by the Science Assistments [13, 14], enabled students to engage in authentic inquiry using a microworld and inquiry support tools. Each problem in our learning environment required students to conduct experiments to determine if a particular independent variable, e.g., container size, affected various outcomes like the melting point or boiling point of a substance.

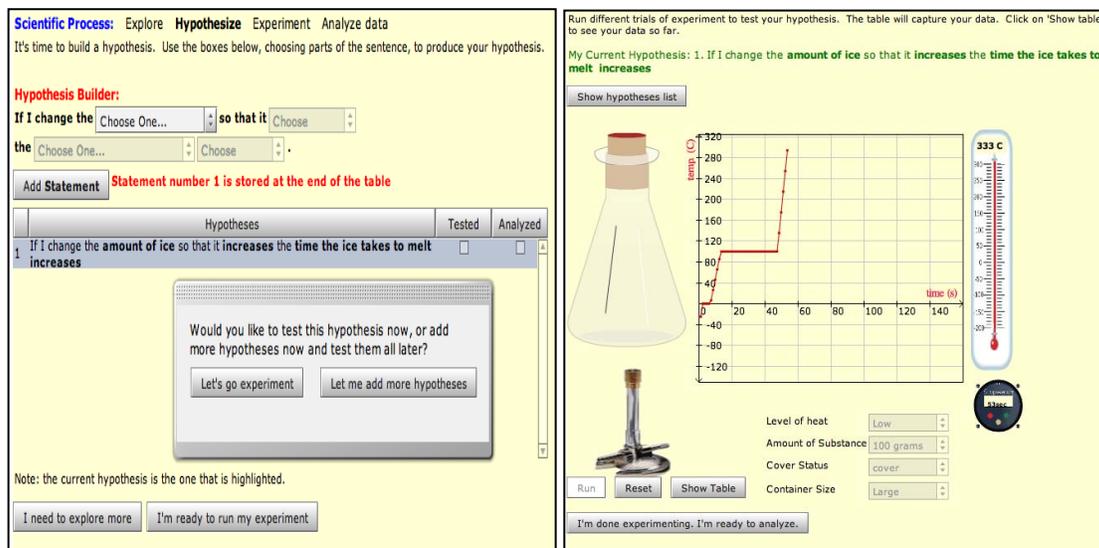


Figure 1. Hypothesizing widget (left) and data collection panel (right) for the phase change microworld.

We scaffold students’ inquiry processes by organizing these tasks into different inquiry stages, namely, “observe”, “hypothesize”, “experiment”, and “analyze data”. Students start in the hypothesizing stage and move between stages in a suggested order but can navigate back and forth between some of the inquiry phases. For example, from the “analysis” stage students can collect more data by returning to the “experiment” stage, they can create new hypotheses by returning to the “hypothesize” stage (starting a new inquiry loop), or can submit their final experimentation procedures and analyses and begin the next problem. While in the hypothesizing stage, they can either explore the microworld or begin collecting data in the experiment phase. Finally, within the experiment phase, students can only move to the analysis phase.

This learning environment has a moderate degree of learner control, less than in purely exploratory learning environments [2], but more than in model-tracing tutors [17] or constraint-based tutors [20]. Though our scaffolding restricts when students can switch inquiry phases, there is enough freedom such that students can approach these inquiry tasks in many ways, e.g., while experimenting, students could set up and run as many different experiments as they desired.

In the Hypothesis stage, the student is prompted to build a hypothesis using drop down boxes (Hypothesis Builder). The fields are: independent variable, change to the independent variable, dependent variable, and change to dependent variable. So, for example, the student can change the first [Choose One...] box to “amount of ice”, which enables the next box. Proceeding, the student can create the hypothesis “If I change the [amount of ice] so that it [increases], the [melting point][doesn’t change].

When students reach the experimentation stage, they can then change independent variables, such as Level of Heat, and see the results within the microworld by running a trial (by clicking on Run). They can also view representations of their full set of hypotheses (by clicking on Show hypotheses list) and they can view the trial run data (clicking on Show Table). Both the data table and the hypothesis list provide external memory aid, allowing the student use information about previous decisions to reflect and plan new experimental trials.

As students solve these inquiry problems, they could engage in a number of behavior patterns. Particular to collecting data, systematic [24] students collect data that test their hypotheses by designing and running controlled experiments. Additionally, such students may use the table tool and hypothesis list to reflect upon their results and plan for additional experiments they may need to run. Students who are unsystematic in their experimental design and collection of data may exhibit haphazard behaviors such as: constructing experiments that do not test their hypotheses, not collecting enough data to support their hypotheses, not using CVS, or running the same experimental setup multiple times [17].

3 Data Set

Participants were 148 eighth grade students, ranging in age from 12-14 years, from a public middle school in Central Massachusetts. These students used the phase change microworld. Students engaged in authentic inquiry problems using the phase change and density microworlds within the Science Assistments learning environment. As part of the phase change activities, students attempted to complete four tasks using our interactive tools.

Each of these students completed at least one data collection activity in the phase change environment (two other students did not use the microworld, and were excluded from analysis). As students solved these tasks, we recorded fine-grained actions within the inquiry support tools and microworlds. The set of actions logged included creating hypotheses, setting up experiments, showing or hiding support tools, running experiments, creating interpretations of data, and transitioning between inquiry activities

(i.e., moving from hypothesizing to data collection). Each action's type, current and previous values (where applicable – for instance, a variable's value), and timestamp were recorded. In all, 27,257 student actions for phase change were logged. These served as the basis for generating text replay clips consisting of contiguous sequences of actions specific to experimenting.

4 Method

The data used for this study was collected by Science Assistments, which logs every widget action performed by the student including button clicks, checkbox choices, etc. Each action has a time stamp, student/problem identifiers, and widget information, and is tagged as to its step (step tag) in the inquiry process. The step tags are a level above a simple action (this is captured by the widget information), representing a step within the inquiry process, across microworlds. This allows us to analyze similar actions across microworlds. These step tags are used for two purposes: as markers to create clips for text replay coding and to categorize data for fine-grain feature extraction.

4.1 Text Replay Coding

Text replay hand coding presented our team with two significant challenges: specific codes and grain size. In designing our text replays, it was necessary to use a coarser grain-size than in prior versions of this method [4]. In particular, it is necessary to show significant periods of experimentation in order to put usage of the table and hypothesis list into context, while limiting clip size to reduce memory load. We decided to use clips that include both the hypothesis and the experiment stages, which is long enough to see context, but short enough to tractably code. Another important issue in grain-size selection is that trial run data from one hypothesis test can be used in another to make inferences about the hypothesis at hand (for instance, by comparing a current trial to one conducted earlier). To compensate for this, we code using both the actions in testing the current hypothesis, and cumulative measures which include actions performed when testing previous hypotheses.

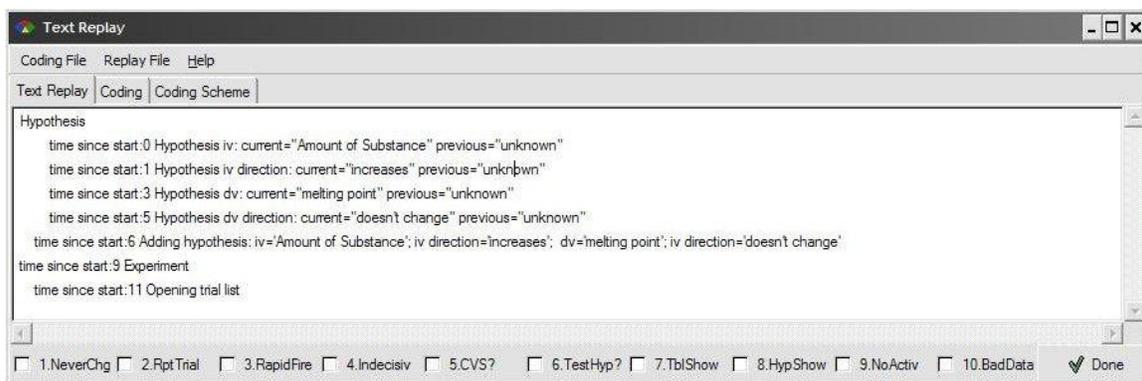


Figure 2 - Clip showing a single Hypothesis-Experiment run (clips may be significantly longer)

To support coding in this fashion, a new tool for text replay tagging was developed in Ruby, shown in Figure 2. The start of the clip is triggered by a hypothesis variable change after the beginning of a new problem. The tool displays all student actions

(hypothesis and experiment) until the student transitions to the analysis stage. Subsequent clips include previous clips and any single new cycle which includes the Hypothesis and Experiment stage. A clip could be tagged with one of 10 tags: “Never Change Variables”, “Repeat Trials”, “Non-Interpretable Action Sequence”, “Indecisiveness”, “Used CVS”, “Tested Hypothesis”, “Used Table to Plan”, “Used Hypothesis Viewer to Plan”, “No Activity”, and “Bad Data.” Specific to our study, we tagged a clip as “Used Table to Plan” (TablePlan) if the clip contained actions indicative that the student viewed the trial run data table in a way consistent with planning for subsequent trials. “Used Hypothesis Viewer to Plan” (HypPlan) was chosen if the clip had actions indicating that the student viewed the hypotheses list in a way consistent with planning for subsequent trials.

4.2 Coding Agreement

Two coders (the third and fourth authors) tagged the data collection clips using at least one of the ten tags. To ensure that a representative range of student clips were coded, we stratified our sample of the clips on condition, student, problem, and within-problem clip order (e.g. first clip, second clip, etc.). The corpus of hand-coded clips contained exactly one randomly selected clip from each problem each student encountered, resulting in 581 clips. Each coder tagged the first 50 clips; the remaining clips were split between the coders. Of the 50 clips tagged, 7 were discarded because of a problem with an early version of the text replay tool where the problem number of the code did not match the problem number of the microworld.

For the 43 clips tagged by each coder, there was high overall tagging agreement, average $\kappa = 0.86$. Of particular relevance to this study, there was strikingly high agreement on the TablePlan, $\kappa = 1$ and of HypPlan, also $\kappa = 1$. Kappa at this level suggests particularly good agreement between coders, which was achieved in part through extensive discussion and joint labeling prior to the inter-rater reliability session. In particular, the coders found these two categories easy to code, as students either tended to spend significant amounts of time reflecting on these tools, or viewed them extremely briefly (or not at all). These categories were also relatively rare, potentially increasing κ by chance; only 8% of clips involved TablePlan and only 4% of clips involved HypPlan.

4.3 Feature Distillation

Features extracted can be grouped into 10 categories: all actions, total trial runs, incomplete trial runs, complete trial runs, pauses, data table display, hypothesis list display, field changes in Hypothesis Builder, hypothesis made, and microworld variable changes. For each of these categories we traced the number of times the action and the time taken for each action. Two other categories were included indirectly related to actions: the number trials where only one independent variable was different between the two trials and the number of times a trial was repeated. These last two had no times associated with them.

The microworld activity was divided into tasks in which the focus was a specific independent variable. Since there were four independent variables, there were four tasks. Within a task, the student is allowed to make and test several hypotheses. For each of the

12 categories above, we extracted data for each hypothesis the student worked on (non-cumulative data), and across all hypotheses in the task (cumulative data). The reason for this is that within each task, the data table accumulates the trial run data across hypotheses. This allows the students to compare trial runs testing previous hypotheses with the runs made in the current hypothesis.

Lastly, the time data was distilled to obtain the following values: minimum, maximum, standard deviation, mean and mode. It is these values plus the count which was used in the machine learning model. This data was arranged in a comma-delimited flat file suitable for input into RapidMiner. The data was divided into files, one for each coded feature. The coded feature being the first item on the line, followed by the distilled features described above.

4.4 Machine Learning Algorithms

Machine-learned detectors of the two behavioral patterns were developed within RapidMiner 4.6 [19] using the default settings. Detectors were built using J48 decision trees, with automated pruning to control for over-fitting, the same technique used in [26] and [6]. Six-fold cross-validation was conducted at the student level (e.g. 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 groups of students. We assessed the classifiers using two metrics. First, we used A' [16]. A' is the probability that if the detector is comparing two clips, one involving the category of interest (TablePlan or HypPlan) and one not involving that category, it will correctly identify which clip is which. A' is equivalent to both the area under the ROC curve in signal detection theory, and to W , the Wilcoxon statistic [16]. A model with an A' of 0.5 performs at chance, and a model with an A' of 1.0 performs perfectly. In these analyses, A' was used at the level of clips, rather than students. Statistical tests for A' are not presented in this paper. The most appropriate statistical test for A' in data across students is to calculate A' and standard error for each student for each model, compare using Z tests, and then aggregate across students using Stouffer's method [5] – however, the standard error formula for A' [16] requires multiple examples from each category for each student, which is infeasible in the small samples obtained for each student in our text replay tagging. Another possible method, ignoring student-level differences to increase example counts, biases undesirably in favor of statistical significance.

Second, we used Kappa (κ), which assesses whether the detector identifies is better than chance at identifying the correct action sequences as involving the category of interest. A Kappa of 0 indicates that the detector performs at chance, and a Kappa of 1 indicates that the detector performs perfectly. As Kappa looks only at the final label, whereas A' looks at the classifier's degree of confidence, A' can be more sensitive to uncertainty in classification than Kappa.

5 Results

We constructed and tested detectors using our corpus of hand-coded clips. TablePlan and HypPlan detectors were constructed from a combination of the subset of the first 43 clips

that the two coders agreed on, the remaining clips, tagged separately by the two coders. In total, 570 tagged clips were used for each detector. Of these clips, 47 out of 570 were tagged with TablePlan (8%) and 20 out of 570 (4%) were tagged with HypPlan.

Table 1. Best results for detectors of each coding category

Category	A'	κ	Attribute type	% students in data
HypPlan	.93	.14	Non-cumulative	8%
TablePlan	.94	.46	Cumulative	4%

Detectors were generated for each behavior using J48 decision trees and two sets of attributes, cumulative and non-cumulative attributes. Thus, four different detectors were constructed two for TablePlan and two for HypPlan. The TablePlan detector using cumulative attributes ($A' = .94$, $\kappa = .46$) performed slightly better than the detector built with non-cumulative attributes ($A' = .96$, $\kappa = .36$). Both versions of the detector achieved excellent performance, comparable to detectors of gaming the system refined over several years (e.g., Baker & de Carvalho, 2008), and are very likely to be appropriate for use in interventions. The HypPlan detectors did not perform as well, achieving $A' = 0.93$, $\kappa = 0.14$ for the non-cumulative attributes and $A' = .97$, $\kappa = 0.02$ for the cumulative attributes. The substantial difference between A' and κ is unusual. It appears that what happened in this case is that the model, on cross-validation, classified many clips incorrectly with low confidence; in other words, A' by considering pair-wise comparisons catches the overall rank-ordered correctness of the detector across confidence values even though many clips were mis-categorized at the specific threshold chosen by the algorithm. One possibility is that the low number of HypPlan labels in the data set made the detectors more prone to over-fitting. This result suggests that the HypPlan detector is probably acceptable for fail-soft interventions, where students assessed with low confidence (in either direction) can receive interventions that are not costly if mis-applied.

6 Discussion and Conclusions

In this paper, we have presented models for detecting planning within science inquiry learning. Our efforts to detect planning from data table usage have met with greater initial success than our attempts to detect planning within the hypothesis list, although both detectors are, we feel, good enough to use for some forms of instructional intervention. The detector for showing data table use (TablePlan) in planning can detect a student using the data table effectively from one not using the data table effectively for planning 94% of the time. The $\kappa = .46$ is respectable, so this detector can be used robustly to scaffold table use for planning during inquiry. If we detect that a student is not using the table effectively, we can suggest that the user look at the table and provide hints on how to compare one table row with another, and how to use this to plan the next trial.

On the other hand, the detector for using the hypotheses table for planning (HypPlan) did not perform as well. Although it had a very good A' (.93 and .97), the κ was low, meaning that if we used this detector for scaffolding, we will need to do it in a fail-soft manner. There is reason to believe this approach may be successful. For example, an early detector of gaming the system [7] with a similar κ and lower A' was found to be effective for improving gaming students' learning when used in a fail-soft manner. In addition, combining the HypPlan detector with another (for example, one that detects control for variables strategy or CVS) may compensate for its low κ . So for example, if a detector indicated that CVS was not being used, this detector also can be used to decide if scaffolding should include a hint regarding how the student should use the hypothesis table in order to reflect on their work. In this fashion, interventions based on this detector will only be given when there is additional reason to believe that intervention is needed.

Future work will include improving our κ for HypPlan and finding other meta-cognitive tasks that can be detected effectively. This would require an expansion of the tags we used and perhaps a way to track student progress from one problem to another, since lesson-wide attributes may be useful for measuring students' progress.

By detecting planning in real time, rich adaptive scaffolding becomes feasible [13, 14]. In addition, with helping students learn both content and inquiry skills, scaffolding for planning can help them become better learners, possibly by influencing their meta-cognitive skill development [1, 23]. This study makes an important contribution towards linking these two areas of research, namely, meta-cognitive skills and planning during scientific inquiry.

Acknowledgements

This research is funded by the National Science Foundation (NSF-DRL#0733286), Janice Gobert, Principal Investigator, Neil Heffernan, Ryan Baker, and Carolina Ruiz, Co-Principal Investigators, and the U.S. Department of Education (R305A090170), Janice Gobert, Principal Investigator, Neil Heffernan, Ken Koedinger, and Joe Beck, Co-Principal Investigators. Any opinions expressed are those of the authors and do not necessarily reflect those of the funding agencies.

References

- [1] Alevan, V., McLaren, B., Roll, I., Koedinger, K.: Toward Meta-cognitive Tutoring: A Model of Help Seeking with a Cognitive Tutor. *International Journal of Artificial Intelligence in Education* 16(2), p. 101-128, 2006
- [2] Amershi, S., Conati, C. Combining Unsupervised and Supervised Machine Learning to Build User Models for Exploratory Learning Environments. *Journal of Educational Data Mining*, 2009, 1(1), p. 71-81.
- [3] Azevedo, R.: Theoretical, conceptual, methodological, and instructional issues in research on metacognition and self-regulated learning: A discussion. *Metacognition Learning* 4(1), p. 87-95, 2009

- [4] Baker, R. S. J. d., Corbett, A. T., Wagner, A. Z. Human Classification of Low-Fidelity Replays of Student Actions. *Proceedings of the Educational Data Mining Workshop at the 8th International Conference on Intelligent Tutoring Systems*, 2006. p. 29-36.
- [5] Baker, R. S. J. d., Corbett, A. T., Aleven, V. Improving Contextual Models of Guessing and Slipping with a Truncated Training Set. *Proceedings of the 1st International Conference on Educational Data Mining*, 2008. p. 67-76.
- [6] Baker, R. S. J. d., de Carvalho, A. M. J. A. Labeling Student Behavior Faster and More Precisely with Text Replays. *Proceedings of the 1st International Conference on Educational Data Mining*. p. 38-47.
- [7] Baker, R.S.J.d., Corbett, A.T., Koedinger, K.R., Evenson, E., Roll, I., Wagner, A.Z., Naim, M., Raspat, J., Baker, D.J., Beck, J. (2006) Adapting to When Students Game an Intelligent Tutoring System. *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, 392-401.[8] Beck, J. Engagement tracing: using response times to model student disengagement. *Proceedings of the 12th International Conference on Artificial Intelligence*, 2005. p. 88-95.
- [9] Cetinas, S., Si, L., Xin, Y. P., Hord, C. Automatic Detection of Off-Task Behaviors in Intelligent Tutoring Systems with Machine Learning Techniques. *IEEE Transactions on Learning Technologies*, in press.
- [10] Chen, Z., Klahr, D.: All Other Things Being Equal: Acquisition and Transfer of the Control of Variables Strategy. *Child Development*, 70(5), 1098-1120 (1999)
- [11] de Jong, T. Computer simulations - Technological advances in inquiry learning. *Science*, 312, 2006, , p. 532-533.
- [12] Dignath, C., Buttner, G. Components of fostering self-regulated learning among students: A meta-analysis on intervention studies at primary and secondary school level. , 2008, 3, p. 231-264.
- [13] Gobert, J., Heffernan, N., Baker, R., Ruiz, C.: AMI: ASSISTments Meets Inquiry (NSF-DRL #0733286)., Awarded September 2007 from National Science Foundation
- [14] Gobert, J., Heffernan, N., Koedinger, K., Beck, J.: ASSISTments Meets Science Learning (AMSL; R305A090170)., Awarded February 2009 from the U.S. Dept. of Education
- [15] Hadwin, A. F., Nesbit, J. C., Jamieson-Noel, D., Code, J., Winne, P. H.: The use of computer-based environments for understanding and improving self-regulation. *Metacognition Learning* 2, p. 107-124, 2007
- [16] Hanley, J. A., McNeil, B. J. The Meaning and Use of the Area under a Receiver Operating Characteristic (ROC) Curve. *Radiology*, vol. 143, 1982. p. 29-36.

- [17] Koedinger, K. R., Corbett, A. T.: Cognitive Tutors: Technology bringing learning sciences to the classroom. In (Ed.), R. K. (Eds.) *The Cambridge handbook of the learning sciences*, 2006.: Cambridge University Press. p. 61-77.
- [18] Manlove, S., Lazonder, A. W., Dejong, T.: Software scaffolds to promote regulation during scientific inquiry learning. *Metacognition Learning* 2, p. 141-155, 2007
- [19] Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T. YALE: Rapid Prototyping for Complex Data Mining Tasks. *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006)*, 2006. p. 935-940.
- [20] Mitrovic, A., Mayo, M., Suraweera, P., Martin, B. Constraint-based tutors: a success story. Springer-Verlag (Eds.) *Proceedings of the Industrial & Engineering Application of Artificial Intelligence & Expert Systems Conference IEA/AIE-2001*, 2001. p. 931-940.
- [21] NSES: National Committee on Science Education Standards and Assessment. (1996)., National Science Education Standards, Washington, D.C., National Academy Press, 1996
- [22] Roll, I., Aleven, V., McLaren, B., Koedinger, K.: Designing for metacognition--applying cognitive tutor principles to the tutoring of help seeking. *Metacognition Learning* 2, p. 125-140, 2007
- [23] Schraw, G.: A conceptual analysis of five measures of metacognitive monitoring. *Metacognition Learning* 4, p. 33-45, 2009
- [24] Shih, B., Koedinger, K., Scheines, R. A Response Time Model for Bottom-Out Hints as Worked Examples. *Proceedings of the 1st International Conference on Educational Data Mining*, 2008. p. 117-126.
- [25] Veenman, M. V. J., Van Hout-Worters, B. H. A. M., Afflerback, P.: Metacognition and learning: conception and methodological considerations. *Metacognition Learning* 1, p. 3-14, 2006
- [26] Walonoski, J. A., Heffernan, N. T. Detection and Analysis of Off-Task Gaming Behavior in Intelligent Tutoring Systems. *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, 2006. p. 382-391.
- [27] Winne, P. H., Nesbit, J. C., Kumar, V., Hadwin, A. F., Lajoie, S. P., Azevedo, R., Perry, N. E. Supporting Self-Regulated Learning with gStudy Software: The Learning Kit Project. *Tech., Inst., Cognitive Learning*, 2006. p. 105-113.

Skill Set Profile Clustering: The Empty K-Means Algorithm with Automatic Specification of Starting Cluster Centers

Rebecca Nugent¹, Nema Dean², and Elizabeth Ayers³
rnugent@stat.cmu.edu, nema@stats.gla.ac.uk, eayers@berkeley.edu

¹Department of Statistics, Carnegie Mellon University

²Department of Statistics, University of Glasgow

³Graduate School of Education, University of California, Berkeley

Abstract. While students' skill set profiles can be estimated with formal cognitive diagnosis models [8], their computational complexity makes simpler proxy skill estimates attractive [1, 4, 6]. These estimates can be clustered to generate groups of similar students. Often hierarchical agglomerative clustering or k-means clustering is utilized, requiring, for K skills, the specification of 2^K clusters. The number of skill set profiles/clusters can quickly become computationally intractable. Moreover, not all profiles may be present in the population. We present a flexible version of k-means that allows for empty clusters. We also specify a method to determine efficient starting centers based on the Q -matrix. Combining the two substantially improves the clustering results and allows for analysis of data sets previously thought impossible.

1 Introduction

A common objective in educational research is the identification of students' current skill set profiles. That is, which skills do they have? Which skills do they not have? Which skills are they in the process of learning? A variety of cognitive diagnosis models (e.g. DINA, NIDA, RUM) estimate these latent profiles using information from a student item response matrix and an expert-elicited assignment matrix of the skills required for each item [8, 10]. However, even simple models become difficult to estimate and computationally infeasible as the number of skills, items, and students grow [8].

Recent work has proposed using computationally simpler skill set estimates, e.g. capability scores and sum scores, as proxies for the cognitive diagnosis model estimates [1, 2, 4, 6]. These estimates are then clustered using common methods such as k-means and hierarchical linkage clustering to generate groups of students with similar skill set profiles. A common assumption is that all possible (combinations of complete/zero skill mastery) profiles exist in the population, a restriction that prevents us from being able to work with small samples or large numbers of skills. In addition, both capability scores and sum scores suffer from a strong dependency on a conjunctive assumption, namely that to answer an item correctly, the student must have completely mastered all necessary skills. This assumption effectively (and possibly erroneously) attenuates the individual skill set estimates in the presence of multiple skill items and relies heavily on the presence of large numbers of single skill items for reasonable estimates (which in our view is in most cases impractical).

In this paper, we propose a flexible version of k-means that utilizes more appropriate starting centers given the conjunctive assumption (conditioning on the items themselves) and allows for empty clusters, removing the restriction that each possible skill set profile will have a corresponding cluster. In our work thus far, this version outperforms both traditional k-means and hierarchical clustering in almost all situations. In Section 2, we review two skill mastery estimates and hierarchical agglomerative and k-means clustering. Details of our more flexible version of k-means are provided in Section 3; selection of sensible starting values and an illustrative example follow in Section 4. Further simulation results are in Section 5. In Section 6, we finish with concluding remarks and other possible applications.

2 Skill Set Profile Clustering

In general, the goal of cognitive diagnosis models (CDMs) is to estimate the true skill set profile for each student. Given K skills, the true skill set profile for student i is denoted α_i where $\alpha_{ik} \in \{0, 1\}$ for $k = 1, 2, \dots, K$. A student that has mastered skills 1, 3 but not skill 2 would have the profile $\alpha_i = \{1, 0, 1\}$. There are 2^K possible skill set profiles for K skills; this collection of profiles is the set of corners of a K -dimensional unit hyper-cube. For example, if $K = 2$, the four possible profiles are: $\{0, 0\}, \{1, 0\}, \{0, 1\}, \{1, 1\}$.

Estimation of the α_i is done using a student response matrix Y and an item-skill dependency matrix Q . Student responses are assembled in a $N \times J$ matrix Y where y_{ij} indicates both if student i attempted item j and whether or not they answered it correctly. N is the total number of students, J the number of items. If student i did not answer item j , then $y_{ij} = NA$ (i.e. the indicator $I_{y_{ij} \neq NA} = 0$). If student i attempted item j ($I_{y_{ij} \neq NA} = 1$), then $y_{ij} = 1$ if they answered correctly (0 if not). The Q -matrix, also referred to as a skill coding or transfer model [3, 11], is a $J \times K$ matrix where $q_{jk} = 1$ if item j requires skill k and 0 if not. The Q -matrix is usually an expert-elicited assignment matrix (here assumed to be known/correct).

2.1 Skill Mastery Estimates

Here we briefly describe two proxy estimates for the CDM estimates, $\hat{\alpha}_i$: sum scores and the capability matrix. Both estimates are easily derived from the response matrix Y and the transfer model Q and have been shown to give comparable results to CDMs [2].

First, we present the *sum score* method of [4, 6]. Here W_i is defined as the vector of sum scores where, for $k = 1, 2, \dots, K$,

$$W_{ik} = \sum_{j=1}^J y_{ij} q_{jk}.$$

The W_{ik} are simply the number of items student i answered correctly for each skill k , assuming that all students answered all items. When an item requires more than one skill, i.e., a *multiple skill item*, it contributes to more than one W_{ik} . The W_i map the students into a K -dimensional hyper-rectangle where the range of the k th dimension is $[0, J_k]$ and J_k is the total number of items that require skill k .

In [1, 2], we define an $N \times K$ *capability matrix* B , where B_{ik} is the proportion of correctly answered items involving skill k that student i attempted. That is,

$$B_{ik} = \frac{\sum_{j=1}^J I_{y_{ij} \neq NA} \cdot y_{ij} \cdot q_{jk}}{\sum_{j=1}^J I_{y_{ij} \neq NA} \cdot q_{jk}}.$$

The vector B_i estimates student i 's skill set knowledge and maps student i into the same K -dimensional unit hypercube as defined by the true α_i . For each B_{ik} , zero indicates no skill mastery, one is complete mastery, and values in between are less certain. This skill knowledge estimate accounts for the number of items in which the skill appears as well as for items not answered. If $B_{ik} = NA$, we could impute an uninformative value (e.g., 0.5, mean, median). The examples presented here do not have any missing values.

In this paper, we use the capability matrix as our skill mastery estimate; however, the presented work could easily incorporate the sum score. (Comments are made where appropriate to indicate any needed changes for the use of sum scores.) In addition, estimates derived from the CDMs could similarly be analyzed.

Regardless of estimate choice, similarly to [4, 6], we find groups of students with similar skill set profiles by clustering the B_i vectors. The algorithm returns a set of cluster centers and a cluster assignment vector. The cluster center represents the skill set profile for that subset of students. Note that cluster centers are not restricted to be in the neighborhood of a hypercube corner (although they could be assigned to one). Returning cluster centers rather than their closest corners gives more conservative estimates of skill mastery (rather than zero/complete mastery). Briefly we describe two commonly used clustering methods.

2.2 Hierarchical Agglomerative Clustering

Hierarchical agglomerative clustering (HC) “links up” groups in order of closeness to form a dendrogram from which a cluster solution can be extracted [5]. The user-defined distance measure is most commonly Euclidean distance. Briefly, all observations begin as their own group. The distances between all pairs of groups are found (initially just the distance between all pairs of observations). The closest two groups are merged; the inter-group distances are then updated. We alternate the merging and updating operations until we have one group containing all observations. All merging steps are represented in a tree structure where two groups are linked at the height equal to their inter-group distance at the time of merging. The algorithm requires *a priori* the definition of the distance between two groups containing multiple observations. Here we use the complete linkage method. Complete linkage defines the distance between two groups as the largest distance between all pairs of observations, one per pair from each group, e.g., for Euclidean distance, $d(C_k, C_l) = \max_{i \in C_k, j \in C_l} \|\underline{x}_i - \underline{x}_j\|$. It tends to partition the data into spherical shapes.

Once constructed, we extract G clusters by cutting the tree at the height corresponding to G branches; any cluster solution with $G = 1, 2, \dots, N$ is possible. In [4], extraction of

$G = 2^K$ clusters is suggested. This choice may not always be wise. First, if not all skill set profiles are present in the population, we may split some profile clusters incorrectly into two or more clusters. Moreover, if $N < 2^K$ (a reasonable scenario for many end-of-year assessment exams), we will be unable to extract the desired number of skill set profiles. [4] has shown that in the presence of single skill items, hierarchical clustering will find the correct clusters under some long test theory conditions (as $N, J \rightarrow \infty$). However, this again relies on the assumption that all possible profiles are present.

2.3 K-means

K-means is a popular iterative algorithm for data $X = \{\underline{x}_1, \underline{x}_2, \dots, \underline{x}_n\} \in R^K$ [9]. It uses squared Euclidean distance as a dissimilarity measure and tries to minimize within-cluster distance and maximize between-cluster distance. For a given number of clusters G , k-means searches for cluster centers m_g and assignments A that minimize the criterion $WC = \sum_{g=1}^G \sum_{A(i)=g} \|\underline{x}_i - m_g\|^2$. The algorithm alternates between optimizing the cluster centers for the current assignment (by the current cluster means) and optimizing the cluster assignment for a given set of cluster centers (by assigning to the closest current center) until convergence (i.e. cluster assignments do not change). It tends to find roughly equal-sized, spherical clusters and requires the number of clusters G and a starting set of cluster centers. A common method for initializing k-means is to choose a random set of G observations as the starting set of centers. In this application, the suggested number of clusters is 2^K , the number of possible skill set profiles for K skills [4]. However, similarly to hierarchical clustering, if we are missing representatives from one or more skill set profiles in our population, forcing 2^K clusters may split some clusters into sub-clusters unnecessarily.

3 Empty K-Means

A traditional problem in k-means is the choice of G . A common approach is to create an “elbow graph” that plots the WC criterion against a range of proposed numbers of clusters. As increasing G almost always corresponds to a decrease in the criterion (depending on the set of starting centers), we subjectively identify the number of clusters that corresponds to the end of the large decreases in the WC value as our choice for G .

However, in this application (and others), we may have a natural number of clusters. While it may seem that we should just search for the 2^K different profiles, this number is likely just an upper bound. All profiles might not be present in the population. Moreover, without careful prior examination of the data, we will not know which profiles might be missing. Ideally, we would like a flexible approach that searches for 2^K possible clusters but is not forced to find them.

We modify the k-means algorithm to allow for empty clusters (or absent skill set profiles) in the following way:

1. Set the 2^K starting cluster centers m_g appropriately in the K -dim hyper-cube (Sec. 4).
2. Create the cluster assignment vector A by assigning each B_i to the closest m_g .
3. For all g , if no B_i is assigned to m_g , i.e. $\sum_i I_{A(i)=g} = 0$, then m_g remains the same. Else, $m_g = \frac{1}{n_g} \sum_{A(i)=g} B_i$.
4. Alternate between 2) and 3) until the cluster assignment vector A does not change.

This algorithm continues to minimize the WC criterion with each step; the empty clusters make no contribution to the criterion value. We discuss the choice of appropriate starting centers in Section 4.

Our k-means variation allows for empty clusters or fewer clusters than originally requested. This flexibility removes the constraint that there be one cluster per skill set profile. Early work in this area has relied heavily on small examples with $K = 2, 3, 4$ skills. With the advent of online tutoring systems and end-of-year assessment exams, the number of skills has grown considerably. It is not uncommon to be interested in $K = 10, 15, 20$, etc. For $K = 10$, say, we would have $2^{10} = 1024$ different skill set profiles. In practice, it would be extremely uncommon to see a sample with all 1024 different subgroups. Moreover, the large number of profiles computationally prohibits clustering of samples where $N < 2^K$. Our k-means variation allows for the identification of the clusters/profiles that we do have; any computational constraints (e.g. memory, storage) are limited and are a characteristic of the operating system/platform and not of the algorithm.

4 Choosing Starting Centers

It is well-known that k-means can be dependent on the set of starting centers [9]. Given our goal of identification of the true skill set profiles in the population, a natural set of starting centers might be the hypercube corners $\alpha_i = \{\alpha_{i1}, \alpha_{i2}, \dots, \alpha_{iK}\}$ where $\alpha_{ik} \in \{0, 1\}$. If students map closely to their profile corners, k-means should locate the groups affiliated with the corners very quickly.

However, even if all profiles are present, the students may not be near their profile corner due to attenuation of our skill estimates in the presence of multiple skill items. Below are two possible Q matrices for $J = 24$ items. In Q_1 , items 1-8 only require skill 1, items 9-16 only skill 2, and items 17-24 only skill 3 (all single skill items). In Q_2 , the first 12 items are single skill; the remaining items require multiple skills. If a student's true skill set profile is $\{0, 1, 0\}$, (s)he should miss items 1-8, 17-24 in Q_1 but receive a B_{i2} of 1. In Q_2 , (s)he should miss items 1-4, 9-24 which correspondingly drops B_{i2} from $\frac{13}{13}$ to $\frac{4}{13}$. Similarly, a student with profile $\{1, 0, 1\}$ will have $B_{i1} = B_{i3} = 1$ for Q_1 but see a drop in capability from $\frac{13}{13}$ to $\frac{7}{13}$ using Q_2 . (Analogous drops are seen in sum scores.) These attenuated estimates are not reflective of the true profiles.

$$(Q_1)^T = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

$$(Q_2)^T = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

4.1 Generating Response Data

To illustrate, we generate response data for $N = 250$ students for both Q -matrices from the deterministic inputs, noisy “and” gate (DINA) model, a common educational research conjunctive cognitive diagnosis model [8]. The DINA item response form is

$$P(y_{ij} = 1 \mid \eta_{ij}, s_j, g_j) = (1 - s_j)^{\eta_{ij}} g_j^{1-\eta_{ij}}$$

where α_i is the true skill set profile and $\eta_{ij} = \prod_{k=1}^K \alpha_{ik}^{q_{jk}}$ indicates whether student i has all skills needed for item j . The slip parameter s_j is $P(y_{ij} = 0 \mid \eta_{ij} = 1)$; the guess parameter g_j is $P(y_{ij} = 1 \mid \eta_{ij} = 0)$. Similar to the capability matrix (and the sum score), if student i is missing any skills required for item j , $P(y_{ij} = 1)$ decreases due to the conjunctive assumption. Prior to simulating the y_{ij} , we fix the skills to be of equal medium difficulty with an inter-skill correlation of either 0 or 0.25 and generate true skill set profiles α_i for each student. (In our work thus far, only a perfect inter-skill correlation has a non-negligible effect on the results.) These choices spread students among the 2^K true skill set profiles. We randomly draw our slip and guess parameters ($s_j \sim \text{Unif}(0, 0.30)$; $g_j \sim \text{Unif}(0, 0.15)$). Given the true skill set profiles and slip/guess parameters, we then generate the student response matrix Y and estimate their corresponding capabilities.

Figure 1a below contains the capabilities estimated from the Q_1 matrix, numbered by their true profile (slightly jittered for visualization purposes). The absence of multiple skills allows the mapping of the students to (near) their profile corners. Figure 1b contains the capabilities estimated via the Q_2 matrix, also jittered, numbered by the true profile. The presence of multiple skills has pulled the non- $\{1, 1, 1\}$ profiles toward the profile $\{0, 0, 0\}$. Using the hypercube corners as the starting centers for empty k-means in the second data set will make it more difficult to find the true groups. In fact, if there are no students within a corner’s octant (0.5 as the cutoff), that profile will not be found. When multiple skill items are included, the hypercube corners are no longer representative of the true profiles. We would expect their locations to be attenuated as well. Given the Q matrix, we map the true skill set profiles to their corresponding rescaled locations in the hypercube.

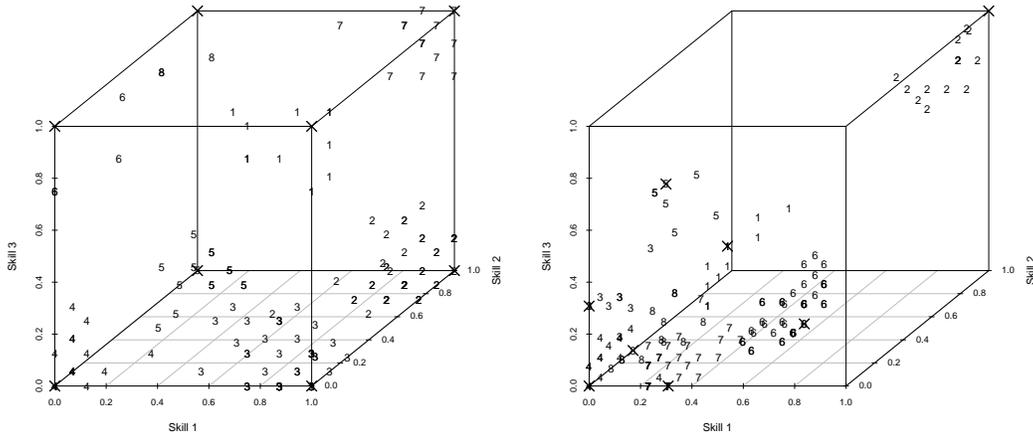


Figure 1: a) B_i for Q_1 ; b) B_i for Q_2 ; Starting centers indicated with X's

4.2 Rescaling the Starting Centers

Let α_p be the possible true skill set profiles where $p = 1, 2, \dots, 2^K$ (e.g. $\{0, 0\}, \{1, 0\}, \{0, 1\}, \{1, 1\}$ for $K = 2$). Let $A_{pj} = \prod_{k=1}^K \alpha_{pk}^{q_{jk}}$. Then A_{pj} indicates whether or not a student with true skill set profile p has all the skills necessary to answer item j . If yes, $A_{pj} = 1$, 0 otherwise. Our starting centers C_p^* are then, for $k = 1, 2, \dots, K$,

$$C_{pk}^* = \frac{\sum_{j=1}^J I_{q_{jk}=1} \cdot A_{pj}}{\sum_{j=1}^J q_{jk}}$$

The numerator is counting the number of items with skill k that the skill set profile p could answer. The denominator is the number of items requiring skill k . (Note that $\sum_{j=1}^J q_{jk} = J_k$. If we were using sum scores, we would not scale C_{pk}^* by the denominator.) If the Q matrix contains only single skill items, the starting centers return to the hypercube corners α_i .

In our example, the starting centers for Q_2 would be, (as indicated by X's in Figure 1b): $(0, 0, 0)$; $(4/13, 0, 0)$; $(0, 4/13, 0)$; $(0, 0, 4/13)$; $(7/13, 7/13, 0)$; $(7/13, 0, 7/13)$; $(0, 7/13, 7/13)$; $(1, 1, 1)$.

These values are representative of the true profile locations given the Q matrix if all students answered items according to their true profiles. They are derived with respect to the conjunctive assumption made by the capability matrix (and the sum score). In practice, we would expect students to slip up or make some lucky guesses; however, setting the starting centers to these rescaled profile locations will allow the empty k-means (or even traditional k-means) to easily find the groups. With respect to missing profiles, we still use the full set of C_p^* as our starting centers and allow the algorithm to discard the unnecessary ones.

Note that A_{pj} is similar in form to η_{ij} in the DINA model. Although they serve a similar function, our approach is not unique to clustering DINA-generated data. The capability score (and the sum score) are reasonable estimates for any conjunctive CDM. As we will see in Section 5, we can similarly rescale the centers for use with other CDMs.

4.3 Performance

After calculating the corresponding B matrix, we cluster the students using hierarchical clustering (complete linkage) and traditional k-means, both asking for $2^3 = 8$ clusters. We then re-cluster with traditional k-means and the empty k-means variation using the rescaled starting centers. (Note that the symmetry of the rescaled starting centers is a direct result of the balanced Q matrix; an unbalanced Q matrix will give asymmetric starting centers.)

To gauge performance, we calculate percent correct as the correct classification rate based on the best one-to-one mapping of clusters to true skill set profiles. We also quantify the clusters' agreement to the true profiles using the Adjusted Rand Index (ARI), a common measure of agreement between two partitions [7]. Under random partitioning, the expected ARI value is zero. Larger values indicate better agreement; the maximum value is one.

Table 1: Comparing Clustering Methods with the True Skill Set Profiles via % Correct, ARIs

	HC: Complete (2^3)	k-means (2^3 , random)	k-means (2^3 , rescaled)	k-means ($\leq 2^3$, rescaled)
% Correct	0.940	0.847	0.973	0.980
ARI	0.952	0.745	0.947	0.971

All methods performed well; the rescaled starting centers resulted in the highest percents correct and ARIs. Our k-means variation (correctly) found 8 clusters. In order to assess the performance when not all possible skill set profiles are present, we then removed the three smallest profiles $\{(0, 0, 1); (0, 1, 1); (1, 0, 1)\}$ (which is the most favorable situation for the other methods) and re-clustered.

Table 2: Comparing Clustering Methods with a Subset of the True Skill Set Profiles via % Correct, ARIs

	HC: Complete (2^3)	k-means (2^3 , random)	k-means (2^3 , rescaled)	k-means ($\leq 2^3$, rescaled)
% Correct	0.756	0.732	—	0.984
ARI	0.759	0.678	—	0.940

Again, all methods performed fairly well. Random starting centers for k-means showed a decrease in performance when clustering a subset of the profiles. Traditional k-means returned an error when using the rescaled starting centers since the initial cluster assignment returned empty clusters (as expected). Our k-means variation, however, found five clusters and had almost perfect agreement with the true skill set profiles. Even if we knew the true number of clusters (5), it is not a guarantee of superior performance. The five-cluster complete linkage solution was 93.5% percent correct with an ARI of 0.937. The traditional k-means (5 random centers) was 80.5% correct with an ARI of 0.679. Even when using only the five rescaled starting centers corresponding to the present profiles, the traditional k-means performance was comparable (97.6%, ARI = 0.946) to using our k-means variation which used the rescaled centers but required only an upper bound on the number of clusters.

5 Simulations

We explore the performance of our approach using two conjunctive CDMs while varying N, J , and K . For each simulation, the Q -matrix is randomly generated with a parameter dictating the percentage of single skill questions. We initially cluster all generated students and then remove a random number of profiles and re-cluster (the notation “—” corresponds to errors in standard k-means). We first simulate from the DINA model (Section 4.1).

Table 3: Performance with DINA-generated Responses: % Correct (ARIs)

K	N	J	Q % S/% M	Profiles Removed	HC: Complete (2^K)	k-means (2^K , random)	k-means (2^K , rescaled)	k-means ($\leq 2^K$, rescaled)
3	200	50	54/46	(4 removed)	0.980 (0.978) 0.640 (0.589)	0.840 (0.776) 0.620 (0.554)	0.975 (0.981) —	0.975 (0.981) 0.900 (0.933)
3	200	50	16/84	(1 removed)	0.615 (0.416) 0.775 (0.515)	0.660 (0.352) 0.639 (0.443)	0.800 (0.778) —	0.820 (0.814) 0.835 (0.838)
8	500	60	45/55	(6 removed)	0.410 (0.197) 0.393 (0.173)	0.414 (0.166) 0.380 (0.129)	— —	0.764 (0.655) 0.236 (0.659)
8	500	60	5/95	(8 removed)	0.332 (0.058) 0.328 (0.048)	0.356 (0.074) 0.343 (0.057)	— —	0.482 (0.199) 0.468 (0.159)
4	30	40	80/20	(2 removed)	0.800 (0.581) 0.741 (0.447)	0.700 (0.390) 0.741 (0.544)	— —	1.000 (1.000) 1.000 (1.000)

In all cases, the k-means variation with attenuated starting centers outperforms the other methods (via ARIs). We also noted in our simulations (not all presented here) that increasing the percentage of multiple skill items decreases the other methods’ performance while our k-means variation remains fairly steady. Moreover, in “classroom” size data sets, this variation identified the profiles present while other methods unnecessarily split the clusters.

We also present results using responses generated from the noisy input, deterministic output “and” gate (NIDA) model, another common conjunctive CDM. The item response form is

$$P(y_{ij} = 1 \mid \alpha_i, s_k, g_k) = \prod_{k=1}^K [(1 - s_k)^{\alpha_{ik}} g_k^{1-\alpha_{ik}}]^{q_{jk}}$$

where s_k, g_k are slip, guess parameters indexed on skill (rather than item); see [8] for further details. Responses are similarly generated; the results are comparable.

Table 4: Performance with NIDA-generated Responses: % Correct (ARIs)

K	N	J	Q % S/% M	Profiles Removed	HC: Complete (2^K)	k-means (2^K , random)	k-means (2^K , rescaled)	k-means ($\leq 2^K$, rescaled)
3	200	50	36/64	(4 removed)	0.935 (0.941) 0.604 (0.432)	0.705 (0.622) 0.549 (0.342)	0.965 (0.942) —	0.965 (0.942) 0.549 (0.408)
3	200	50	18/82	(3 removed)	0.760 (0.552) 0.838 (0.808)	0.830 (0.688) 0.738 (0.649)	0.895 (0.787) —	0.895 (0.787) 0.900 (0.922)
8	500	60	63/37	(7 removed)	0.450 (0.225) 0.429 (0.212)	0.420 (0.163) 0.404 (0.155)	— —	0.734 (0.663) 0.753 (0.680)
4	30	40	54/46	(2 removed)	0.700 (0.357) 0.615 (0.250)	0.633 (0.321) 0.538 (0.202)	— —	0.867 (0.661) 0.846 (0.600)

6 Conclusions

The modified k-means algorithm presented here, called “empty k-means”, allows a more flexible approach to clustering for use in applications such as skill set profile clustering where the true number of clusters is not known, but may be bounded. It allows the user to specify a maximum number of possible clusters which removes the need to make a subjective decision on the number of clusters. We define our attenuated starting cluster centers by the Q-matrix (giving us the hypercube corners in the case of all single skill items). As seen in the simulated results, in cases where all natural clusters were present, such starting values gave superior clustering results (compared with both k-means with random starts and hierarchical clustering). In cases where some natural clusters were not present, the empty k-means algorithm with the defined starting values again had superior performance, while commonly traditional k-means would report an error due to empty clusters. Other applications might fit this framework as well. For example, compositional data on the simplex would have natural cluster centers on the corners of hyper-triangle. Empty k-means could also be used to investigate both the validity of theorized cluster centers and the believed number of clusters. Further exploration of this approach is ongoing.

References

- [1] Ayers, E, Nugent, R, Dean, N. “Skill Set Profile Clustering Based on Student Capability Vectors Computed from Online Tutoring Data”. *Educational Data Mining 2008: 1st International Conference on Educational Data Mining, Proceedings* (refereed). R.S.J.d. Baker, T. Barnes, and J.E. Beck (Eds), Montreal, Quebec, Canada, June 20-21, 2008. p.210-217.
- [2] Ayers, E, Nugent, R, Dean, N. (2009) “A Comparison of Student Skill Knowledge Estimates”. *Educational Data Mining 2009: 2nd International Conference on Educational Data Mining, Proceedings*. Barnes, T., Desmarais, M., Romero, C., and Ventura, S. (Eds), Cordoba, Spain, p.101-110.
- [3] Barnes, T.M. (2003). *The Q-matrix Method of Fault-tolerant Teaching in Knowledge Assessment and Data Mining*. Ph.D. Dissertation, Department of Computer Science, NCSU.
- [4] Chiu, C, Douglas, J. A, Li, X. (2009) Cluster Analysis for Cognitive Diagnosis: Theory and Applications. *Psychometrika*, 74 (4), p.633-665.
- [5] Hartigan, J.A. *Clustering Algorithms*. Wiley. 1975.
- [6] Henson, J., Templin, R., and Douglas, J. (2007) Using efficient model based sum-scores for conducting skill diagnoses. *Journal of Education Measurement*, 44, p.361-376.
- [7] Hubert, L. and Arabie, P. (1985) Comparing partitions. *Journal of Classification*, 2, p.193-218.
- [8] Junker, B.W., Sijtsma K. (2001) Cognitive Assessment Models with Few Assumptions and Connections with Nonparametric Item Response Theory. *Applied Psych Measurement*, 25, p.258-272.
- [9] Steinley, D. (2008). Stability analysis in K-means clustering. *British Journal of Mathematical and Statistical Psychology*, 61, p.255-273.
- [10] Tatsuoka, K.K. (1983). Rule Space: An Approach for Dealing with Misconceptions Based on Item Response Theory. *Journal of Educational Measurement*. 20 (4), p.345-354.

Navigating the parameter space of Bayesian Knowledge Tracing models: Visualizations of the convergence of the Expectation Maximization algorithm

Zachary A. Pardos¹ and Neil T. Heffernan
{zpardos,nth}@wpi.edu
Worcester Polytechnic Institute

Abstract. Bayesian Knowledge Tracing (KT) models are employed by the cognitive tutors in order to determine student knowledge based on four parameters: learn rate, prior, guess and slip. A commonly used algorithm for learning these parameter values from data is the Expectation Maximization (EM) algorithm. Past work, however, has suggested that with four free parameters the standard KT model is prone to converging to erroneous degenerate states depending on the initial values of these four parameters. In this work we simulate data from a model with known parameter values and then run a grid search over the parameter initialization space of KT to map out which initial values lead to erroneous learned parameters. Through analysis of convergence and error surface visualizations we found that the initial parameter values leading to a degenerate state are not scattered randomly throughout the parameter space but instead exist on a surface with predictable boundaries. A recently introduced extension to KT that individualizes the prior parameter is also explored and compared to standard KT with regard to parameter convergence. We found that the individualization model has unique properties which allow it to avoid the local maxima problem.

1 Introduction

Knowledge Tracing (KT) models [1] are employed by the cognitive tutors [2], used by over 500,000 students, in order to determine when a student has acquired the knowledge being taught. The KT model is based on two knowledge parameters: learn rate and prior and two performance parameters: guess and slip. A commonly used algorithm for learning these parameter values from data is the Expectation Maximization (EM) algorithm. Past work [3,4,5], however, has suggested that with four free parameters the standard KT model is prone to converging to erroneous degenerate states depending on the initialized values of these four parameters. In this work we simulate data from a model with known parameter values and then run a grid search over the parameter initialization space of KT to map out which initial values lead to erroneous learned parameters. Through analysis of convergence and error surface visualizations we found that the initial parameter values leading to a degenerate state are not scattered randomly throughout the parameter space but instead exist on a surface within predictable boundaries. A recently introduced extension to KT that individualizes the prior parameter is also explored and compared to standard KT with regard to parameter convergence. We found that the individualization model has unique properties which allow for a greater number of initial states to converge to the true parameter values.

¹ National Science Foundation funded GK-12 Fellow

1.1 Expectation Maximization algorithm

The Expectation Maximization (EM) algorithm is a commonly used algorithm used for learning the parameters of a model from data. EM can learn parameters from incomplete data as well as from a model with unobserved nodes such as the KT model. In the cognitive tutors, EM is used to learn the KT prior, learn rate, guess and slip parameters for each skill, or production rule. One requirement of the EM parameter learning procedure is that initial values for the parameters be specified. With each iteration the EM algorithm will try to find parameters that improve fit to the data by maximizing the log likelihood function, a measure of model fit. There are two conditions that determine when EM stops its search and returns learned parameter results: 1) if the specified maximum number of iterations is exceeded or 2) if the difference in log likelihood between iterations is less than a specified threshold. Meeting condition 2, given a low enough threshold, is indicative of algorithm parameter convergence, however, given a low enough threshold, EM will continue to try to maximize log likelihood, learning the parameters to a greater precision. In our work we use a threshold value of $1e-4$, which is the default for the software package used, and a maximum iteration count of 15. The max iteration value used is lower than typical, however, we found that in the average case our EM runs did not exceed more than 7 iterations before reaching the convergence threshold. The value of 15 was chosen to limit the maximum computation time since our methodology requires that EM be run thousands of times in order to achieve our goal.

1.2 Past work in the area of KT parameter learning

Beck & Chang [3] explained that multiple sets of KT parameters could lead to identical predictions of student performance. One set of parameters was described as the plausible set, or the set that was in line with the authors' knowledge of the domain. The other set was described as the degenerate set, or the set with implausible values such as values that specify that a student is more likely to get a question wrong if they know the skill. The author's proposed solution was to use a Dirichlet distribution to constrain the values of the parameters based on knowledge of the domain.

Corbett & Anderson's [1] approach to the problem of implausible learned parameters was to impose a maximum value that the learned parameters could reach, such as a maximum guess limit of 0.30 which was used in Corbett & Anderson's original parameter fitting code. This method of constraining parameters is still being employed by researchers such as Baker et al. [4] and Ritter et al [5] in their more recent models.

Alternatives to EM for fitting parameters were explored by Pavlik et al. [5], such as using unpublished code by Baker to brute force parameters that minimize an error function. Pavlik also introduced an alternative to KT, named PFA [5] and reported an increase in performance compared to the KT results. Gong, Beck and Heffernan [6] however are in the process of challenging PFA by using KT with EM which they report provides improved prediction performance over PFA with their dataset.

While past works have made strides in learning plausible parameters they lack the benefit of knowing the true model parameters of their data. Because of this, none of past work

has been able to report the accuracy of their learned parameters. One of the contributions of our work is to provide a closer look at the behavior and accuracy of EM in fitting KT models by using synthesized data that comes from a known set of parameter values. This enables us to analyze the learned parameters in terms of exact error instead of just plausibility. To our knowledge this is something that has not been previously attempted.

2 Methodology

Our methodology involves first synthesizing response data from a model with a known set of parameter values. After creating the synthesized dataset we can then train a KT model with EM using different initial parameter values and then measure how far from the true values the learned values are. This section describes the details of this procedure.

2.1 Synthesized dataset procedure

To synthesize a dataset with known parameter values we run a simulation to generate student responses based on those known ground truth parameter values. These values will later be compared to the values that EM learns from the synthesized data. To generate the synthetic student data we defined a KT model using functions from MATLAB's Bayes Net Toolbox (BNT) [7]. We set the known parameters of the KT model based on the mean values learned across skills in a web based math tutor called ASSISTments [9]. These values which represent the ground truth parameters are shown in Table 1.

Table 1. Ground truth parameters used for student simulation

Prior	Learn rate	Guess	Slip
Uniform random dist	0.09	0.14	0.09

Since knowledge is modeled dichotomously, as either learned or unlearned, the prior represents the Bayesian network's confidence that a student is in the learned state. The simulation procedure makes the assumption that confidence of prior knowledge is evenly distributed. 100 users and four question opportunities are simulated, representing a problem set of length four. Each doubling of the number of users also doubles the EM computation time. We found that 100 users were sufficient to achieve parameter convergence with the simulated data. Figure 1 shows pseudo code of the simulation.

```

KTmodel.lrate = 0.09
KTmodel.guess = 0.14
KTmodel.slip = 0.09
KTmodel.num_questions = 4
For user 1 to 100
    prior(user) = rand()
    KTmodel.prior = prior(user)
    sim_responses(user) = sample.KTmodel
End For

```

Figure 1. Pseudo code for generating synthetic student data from known KT parameter values

Student responses are generated probabilistically based on the parameter values. For instance, the Bayesian network will roll a die to determine if a student is in the learned state based on the student’s prior and the learn rate. The network will then again role a die based on guess and slip and learned state to determine if the student answers a question correct or incorrect at that opportunity. After the simulation procedure is finished, the end result is a datafile consisting of 100 rows, one for each user, and five columns; user id followed by the four incorrect/correct responses for each user.

2.2 Analysis procedure

With the dataset now generated, the next step was to start EM at different initial parameter values and observe how far the learned values were from the true values. A feature of BNT is the ability to specify which parameters are fixed and which EM should try to learn. In order to gain some intuition on the behavior of EM we decided to start simple by fixing the prior and learn rate parameters to their true values and focusing on learning the guess and slip parameters only. An example of one EM run and calculation of mean absolute error is shown in the table below.

Table 3. Example run of EM learning the Guess and Slip parameters of the KT model

Parameter	True value	EM initial value	EM learned value
Guess	0.14	0.36	0.23
Slip	0.09	0.40	0.11
Error = $[\text{abs}(\text{Guess}_{\text{True}} - \text{Guess}_{\text{Learned}}) + \text{abs}(\text{Slip}_{\text{True}} - \text{Slip}_{\text{Learned}})] / 2$ = 0.11			

The true prior parameter value was set to the mean of the simulated priors (In our simulated dataset of 100 the mean prior was 0.49). Having only two free parameters allows us to represent the parameter space in a two dimensional graph with guess representing the X axis value and slip representing the Y axis value. After this exploration of the 2D guess/slip space we will explore the more complex four free parameter space.

2.2.1 Grid search mapping of the EM initial parameter convergence space

One of the research questions we wanted to answer was if the initial EM values leading to a degenerate state are scattered randomly throughout the parameter space or if they exist within a defined surface or boundary. If the degenerate initial values are scattered randomly through the space then EM may not be a reliable method for fitting KT models. If the degenerate states are confined to a predictable boundary then true parameter convergence can be achieved by restricting initial parameter values to within a certain boundary. In order to map out the convergence of each initial parameter we iterated over the entire initial guess/slip parameter space with a 0.02 interval. Figure 2 shows how this grid search exploration of the space was conducted.

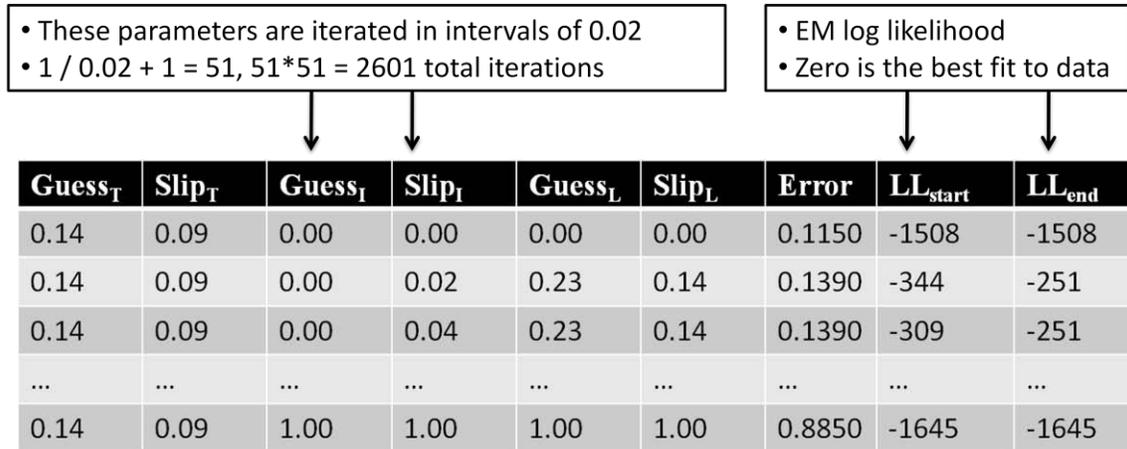


Figure 3. Output of the grid search procedure exploring the initial EM guess/slip parameter space of KT

We started with an initial guess and slip of 0 and ran EM to learn the guess and slip values of our synthesized dataset. When an EM run finished, either because it reached the convergence threshold or the maximum iteration, it returned the learned guess and slip values as well as the log likelihood fit to the data of the initial parameters and the learned parameters (represented by LL_{start} and LL_{end} in the figure). We calculated the mean absolute error between the learned and true values using the formula in Table 3. We then increased the initial slip value by 0.02 and ran EM again and repeated this procedure for every guess and slip value from 0 to 1 with an interval of 0.02.

3 Results

The analysis procedure produced an error and log likelihood value for each guess/slip pair in the parameter space. This allowed for visualization of the parameter space using Guess_{initial} as the X coordinate, Slip_{initial} as the Y coordinate and either log likelihood or mean absolute error as the error function.

3.1 Tracing EM iterations across the KT log likelihood space

The calculation of error is made possible only by knowing the true parameters that generated the synthesized dataset. EM does not have access to these true parameters but instead must use log likelihood to guide its search. In order to view the model fit surface and how EM traverses across it from a variety of initial positions, we set the Z-coordinate (background color) to the LL_{start} value and logged the parameter values learned at each iteration step of EM. We overlaid a plot of these EM iteration step points on the graph of model fit. This combined graph is shown below in figure 4 which depicts the nature of EM's convergence with KT. For the EM iteration plot we tracked the convergence of EM starting positions in 0.10 intervals to reduce clutter instead of 0.02 intervals which were used to created the model fit plot. No EM runs reached their iteration max for this visualization. Starting values of 0 or 1 (on the borders of the graph) do not converge from the borders because of how BNT fixes parameters with 0 or 1 as their initial value.

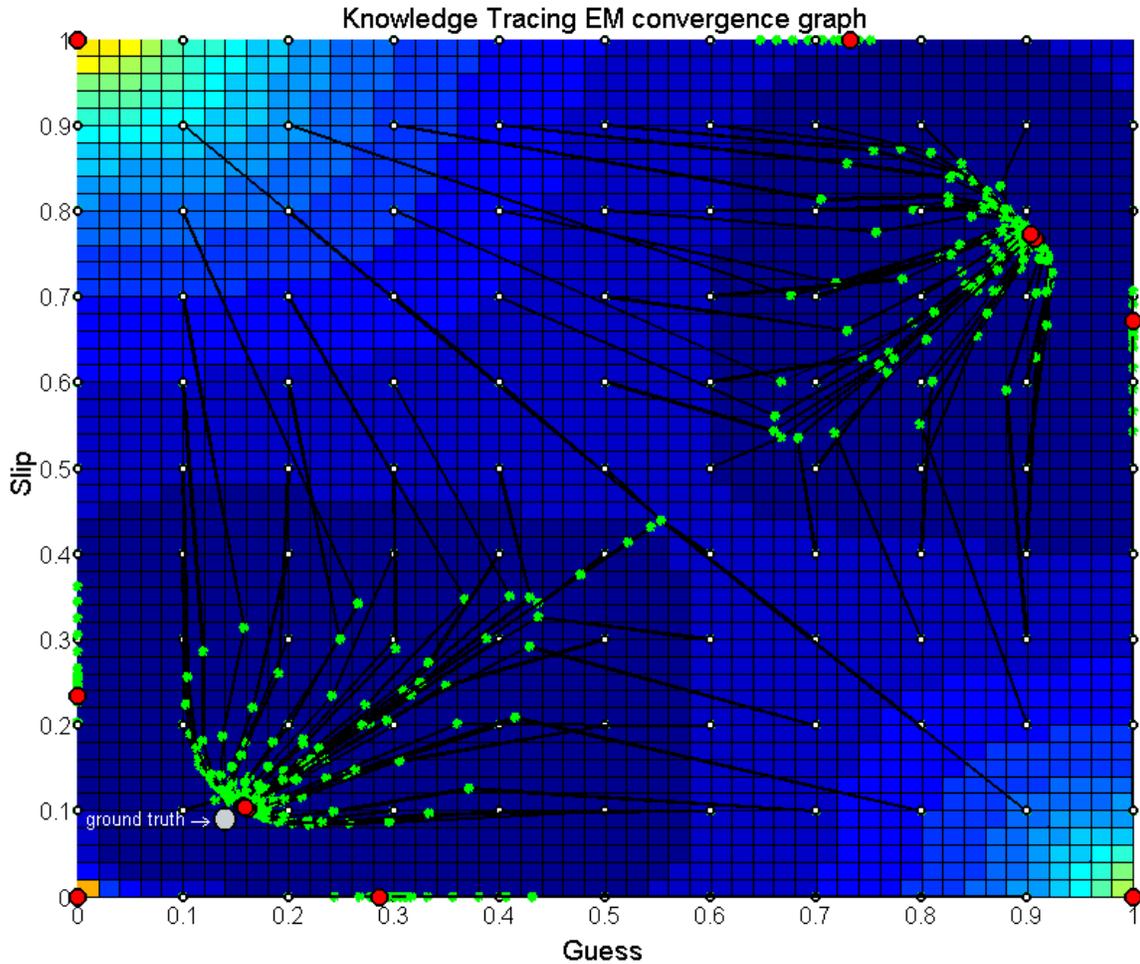


Figure 4. Model fit and EM iteration convergence graph of Bayesian Knowledge Tracing. Small white dots represent parameter starting values. Green dots represent the parameter values at each EM iteration. The red dots represent the resulting learned parameter values and the large white dot is ground truth. The background color is the log likelihood (LL_{start}) of the parameter space. Dark blue represent better fit.

This visualization depicts the multiple global maxima problem of Knowledge Tracing. There are two distinct regions of best fit (dark blue); one existing in the lower left quadrant which contains the true parameter values (indicated by the white “ground truth” dot), the other existing in the upper right quadrant representing the degenerate learned values. We can observe that all the green dots lie within one of the two global maxima regions, indicating that EM makes a jump to an area of good fit after the first iteration. The graph shows that there are two primary points that EM converges to with this dataset; one centered around guess/slip = 0.15/0.10, the other around 0.89/0.76. We can also observe that initial parameter values that satisfy the equation: $guess + slip \leq 1$, such as guess/slip = 0.90/0.10 and 0.50/0.50, successfully converge to the true parameter area while initial values that satisfy: $guess + slip > 1$, converge to the degenerate point.

3.2 *KT convergence when learning all four parameters*

For the full four parameter case we iterated through initial values of the prior, learn rate, guess and slip parameters from 0 to 1 with a 0.05 interval. This totaled 194,481 EM runs

(21^4) to traverse the entire parameter space. For each set of initial positions we logged the converged learned parameter values. In order to evaluate this data we looked at the distribution of converged values for each parameter across all EM runs.

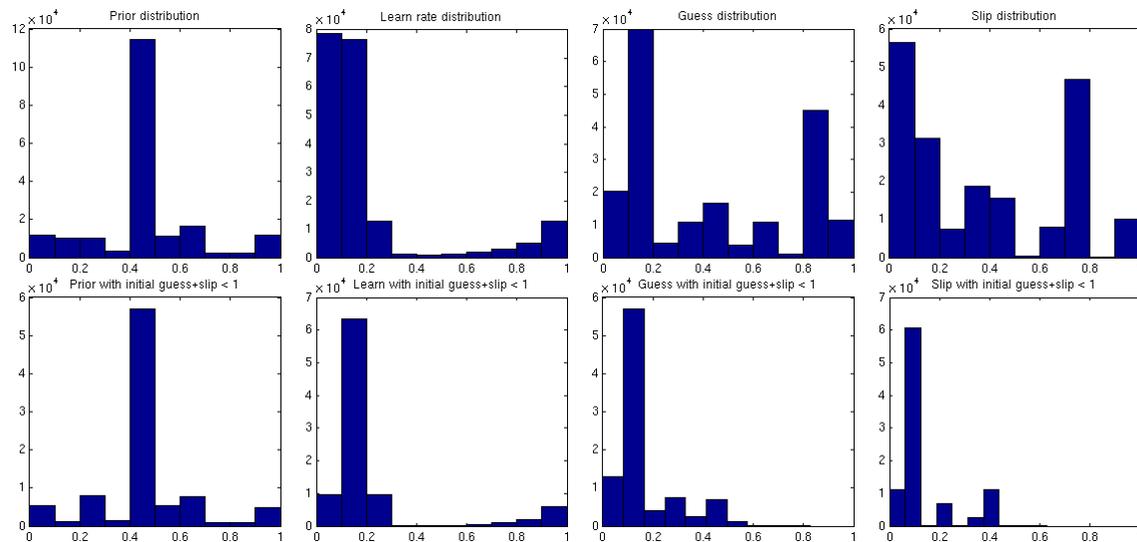


Figure 5. Histograms showing the distribution of learned parameter values for each of the four Knowledge Tracing parameters. The first row shows the parameter distributions across all the EM runs. The second row shows the parameter distributions for the EM runs where initial guess and slip summed to less than 1.

The first row of histograms in Figure 5 shows the distribution of learned parameter values across all EM runs. Generally, we can observe that all parameters have multiple points of convergence; however, each histogram shows a clear single or bi-modal distribution. The prior and learn rate appear to be the parameters that are easiest to learn since the majority of EM runs lead to values near their true values. The guess and slip histograms exhibit more of the bi-modal behavior seen in the two parameter case, with points of convergence at opposite ends of the parameter space. In the two parameter case, initial guess and slip values that summed to less than one converged towards the ground truth coordinate. To see if this trend generalized with four free parameters we generated another set of histograms but only included EM runs where the initial guess and slip parameters summed to less than one. These histograms are shown in the second row.

3.3 Evaluating an extension to KT called the *Prior Per Student model*

We evaluated a model [9], recently introduced by the authors, that allows for individualization of the prior parameter. By only modeling a single prior, Knowledge tracing makes the assumption that all students have the same level of knowledge of a particular skill before using the tutor. The Prior Per Student (PPS) model challenges that assumption by allowing each student to have a separate prior while keeping the learn, guess and slip as parameters of the skill. The individualization is modeled completely within a Bayesian model and is accomplished with the addition of just a single node, representing student id, and a single arc, connecting the student node to the first opportunity knowledge node. We evaluated this model using the two-parameter case, where guess and slip are learned and learn rate and prior are fixed to their true values.

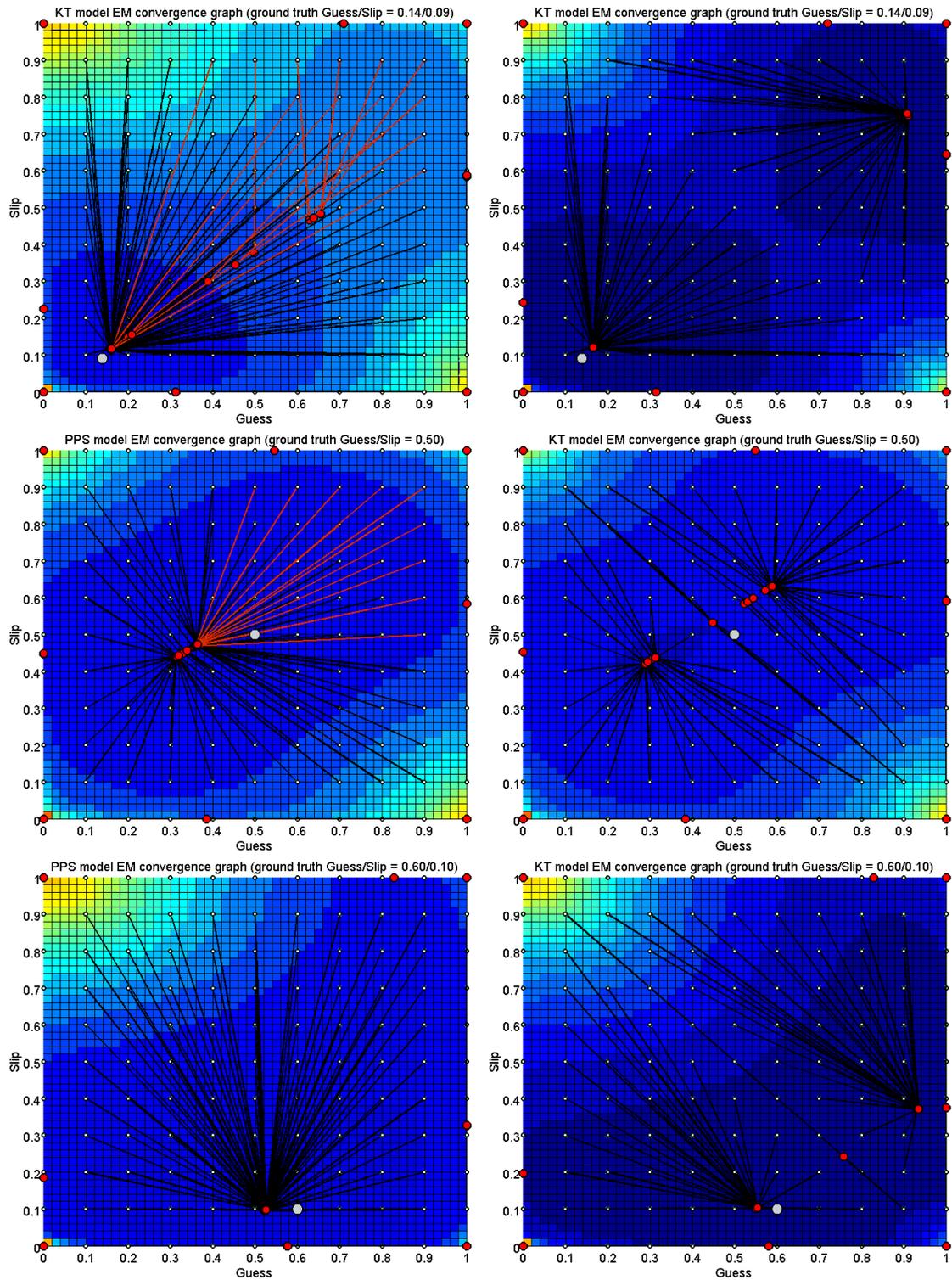


Figure 6. EM convergence graphs of the Prior Per Student (PPS) model (left) and KT model (right). Results are shown with ground truth datasets with guess/slip of 0.14/0.09, 0.50/0.50 and 0.60/0.10

The KT models, in the right column of figure 6, all show multiple points of convergence with only one of the points near the ground truth coordinate (white dot). Unlike KT, the

PPS models, in the left column, have a single point of convergence regardless of the starting position and that single point is near the ground truth value. The red lines in two of the PPS models indicate that the maximum iteration count was reached. In the 0.14/0.09 model it appears that PPS with starting parameters in the upper right region were converging towards the true values but hit the max iteration count before arriving. The PPS model was shown [9] to provide improved prediction over standard knowledge tracing with real world datasets. The visualizations shown in figure 6 suggest that this improved prediction accuracy is likely due in part to the PPS model's improved parameter learning accuracy from a wider variety of initial parameter locations.

In the case of the PPS models show above there were as many prior parameters as there were students and these parameters were all set to the values that were generated for each simulated student as seen in the line "KTmodel.prior = prior(user)" in figure 1. Accurately inferring many initial prior values would be difficult in practice; however, a heuristic is described in Pardos et al [9] that seeds each individual prior based on the student's first response. Applying this same heuristic to our synthesized dataset with ground truth guess/slip values of 0.14/0.09 we found that all points converged to the true parameter location without EM reaching its maximum iteration count. This performance suggests that single point convergence to the true parameters is possible with the PPS model without the benefit of individual student prior knowledge estimates. A more detailed description and analysis of this technique is in work that is in preparation.

4 Discussion and Future Work

An argument can be made that if two sets of parameters fit the data equally well then it makes no difference if the parameters used are the true parameters. This is true when prediction of responses is the only goal. However, when inferences about knowledge and learning are being made, parameter plausibility and accuracy is crucial. It is therefore important to understand how our student models and fitting procedures behave if we are to draw valid conclusions from them. In this work we have depicted how KT exhibits multi-modal convergence properties due to its multi-modal log likelihood parameter space. We demonstrated that with our simulated dataset, choosing initial guess and slip values that summed to less than one allowed for convergence towards the ground truth values in the two parameter case and in the four parameter case, applying this same rule resulted in a convergence distribution with a single mode close to the ground truth value. Lastly, we found that use of the Prior Per Student model eliminated the multiple maxima dilemma in the two parameter case for our synthesized datasets and use of a prior seeding heuristic for PPS resulted in performance comparable to having perfect knowledge of the individual prior confidence probabilities.

This research raises a number of questions such as how KT models behave with a different assumption about the distribution of prior knowledge. What is the effect of increased number of students or increased number of question responses per student on parameter learning accuracy? How does PPS converge with four parameters and what does the model fit parameter convergence space of real world datasets look like? These are questions that are still left to be explored by the EDM community.

Acknowledgements

We would like to thank all of the people associated with creating ASSISTments listed at www.ASSISTments.org. We would also like to acknowledge funding from the US Department of Education, the National Science Foundation, the Office of Naval Research and the Spencer Foundation. All of the opinions expressed in this paper are those of the authors and do not necessarily reflect the views of our funders.

References

- [1] Corbett, A. T., & Anderson, J. R. (1995). Knowledge tracing: modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4, 253–278.
- [2] 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.
- [3] Beck, J. E. and Chang, K. M. Identifiability: A fundamental problem of student modeling. In *Proceedings of the 11th International Conference on User Modeling*, 2007, pp. 137-146.
- [4] Baker, R.S.J.d., Corbett, A.T., Alevan, 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*, 406-415.
- [5] Pavlik, P.I., Cen, H., Koedinger, K.R. (2009). Performance Factors Analysis - A New Alternative to Knowledge Tracing. In *Proceedings of the 14th International Conference on Artificial Intelligence in Education*. Brighton, UK, 531-538
- [6] Gong, Y, Beck, J. E., Heffernan, N. T. In Press (2010) Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting. In *Proc. of The 10th International Conference on Intelligent Tutoring Systems*, Pittsburgh.
- [7] Kevin Murphy. The bayes net toolbox for matlab. *Computing Science and Statistics*, 33, 2001.
- [9] Pardos, Z. A., Heffernan, N. T. In Press (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.
- [10] Ritter, S., Harris, T., Nixon, T., Dickison, D., Murray, C., Towle, B. (2009) Reducing the knowledge tracing space. In Barnes, Desmarais, Romero, & Ventura (Eds.). In *Proceedings of the 2nd International Conference on Educational Data Mining*. pp. 151-160. Cordoba, Spain.

Mining Rare Association Rules from e-Learning Data

Cristóbal Romero, José Raúl Romero, Jose María Luna, Sebastián Ventura
cromero@uco.es, jrromero@uco.es, i32luarj@uco.es, sventura@uco.es
Dept. of Computer Science, University of Córdoba, Spain

Abstract. Rare association rules are those that only appear infrequently even though they are highly associated with very specific data. In consequence, these rules can be very appropriate for using with educational datasets since they are usually imbalanced. In this paper, we explore the extraction of rare association rules when gathering student usage data from a Moodle system. This type of rule is more difficult to find when applying traditional data mining algorithms. Thus we show some relevant results obtained when comparing several frequent and rare association rule mining algorithms. We also offer some illustrative examples of the rules discovered in order to demonstrate both their performance and their usefulness in educational environments.

1 Introduction

Nowadays most research on Association Rule Mining (ARM) has been focused on discovering common patterns and rules in large datasets. In fact, ARM is widely and successfully used in many different areas, such as telecommunication networks, market and risk management, inventory control, mobile mining, graph mining, educational mining, etc. The patterns and rules discovered are based on the majority of commonly repeated items in the dataset, though some of these data can be either obvious or irrelevant [5]. Unfortunately, not enough attention has been paid to the extraction process of rare association rules, also known as non-frequent, unusual, exceptional or sporadic rules, which provide valuable knowledge about non-frequent patterns. The aim of Rare Association Rule Mining (RARM) is to discover rare and low-rank itemsets to generate meaningful rules from these items. Notice that this specific type of rule cannot be revealed easily using traditional association mining algorithms.

In previous works, other authors have applied ARM to e-learning systems extensively to discover frequent student-behavior patterns [13], [7]. However, RARM has been hardly applied to educational data, despite the fact that infrequent associations can be of great interest since they are related to rare but crucial cases. For instance, they might allow the instructor to verify a set of rules concerning certain infrequent/abnormal learning problems that should be taken into account when dealing with students with special needs. Thus, this information could help the instructor to discover a minority of students who may need specific support with their learning process. From the perspective of knowledge discovery, the greatest reason for applying RARM in the field of education is the imbalanced nature of data in education, as in other real-world tasks, i.e., some classes have many more instances than others. Furthermore, in applications like education, the minor parts of an attribute can be more interesting than the major parts; for example, students who fail or drop out are usually less frequent than those students who fare well. In the field of association rule mining, the rare item problem [6] is essentially considered to be a data imbalance problem which may, on either side of the association rule, give rise to severe problems. The problem of imbalance has only been dealt with in one educational data mining study [9]. However, in this work, data was firstly modified/preprocessed to solve the problem of imbalance and then several different classification algorithms were applied instead of specific association rule algorithms.

In this paper, we explore the application of RARM to student data stored in a large Moodle repository to discover information about infrequent student behavior. This paper is organized as follows. Section 2 presents some background on frequent and infrequent association rule mining while Section 3 describes the experiments carried out and the analysis of the most relevant results obtained, as well as including a description of the most accurate rules mined by applying both ARM and RARM to a Moodle dataset containing real information. Finally, conclusions are outlined in Section 4.

2 Background

Association Rule Mining is one of the most popular and well-known data mining methods for discovering interesting relationships between variables in transaction databases or other data repositories [2]. An association rule is an implication $X \Rightarrow Y$, where X and Y are disjoint itemsets (i.e., sets with no items in common). The intuitive meaning of such a rule is that when X appears, Y also tends to appear. The two traditional measures for evaluating association rules are support and confidence. The confidence of an association rule $X \Rightarrow Y$ is the proportion of the transactions containing X which also contain Y . The support of the rule is the fraction of the database that contains both X and Y . The problem of association rule mining is usually broken down into two subtasks. The first one is to discover those itemsets whose occurrences exceed a predefined support threshold, and which are called frequent itemsets. A second task is to generate association rules from those large items constrained by minimal confidence. Nowadays, the problem of frequent itemset mining has been studied widely and many algorithms have already been proposed [3], mainly variations or improvements of the Apriori algorithm [2] which is the first, simplest and most common ARM algorithm. Most research in the area of ARM is focused on the sub-problem of efficient frequent rule generation. However in some data mining applications relatively infrequent associations are likely to be of great interest, too. Though these algorithms are theoretically expected to be capable of finding rare association rules, they actually become intractable if the minimum level of support is set low enough to find rare rules [5].

The problem of discovering rare items has recently captured the interest of the data mining community [1]. As previously explained, rare itemsets are those that only appear together in very few transactions or some very small percentage of transactions in the database [5]. Rare association rules have low support and high confidence in contrast to general association rules which are determined by high support and a high confidence level. Figure 1 illustrates how the support measure behaves in relation to the two types of rules.

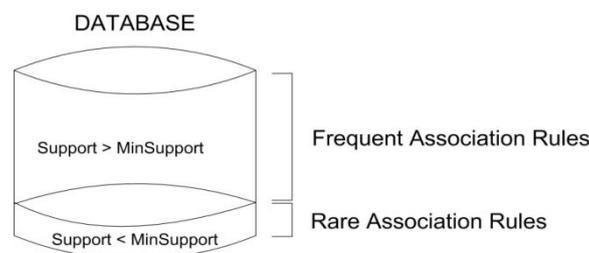


Figure 1. Rules in a database.

There are several different approaches to discover rare association rules. The simplest way is to directly apply the Apriori algorithm [2] by simply setting the minimum support threshold to a low value. However, this leads to a combinatorial explosion, which could produce a huge number of patterns, most of them frequent with only a small number of them actually rare. A different proposal, known as Apriori-Infrequent, involves the modification of the Apriori algorithm to use only the above-mentioned infrequent itemsets during rule generation. This simple change makes use of the maximum support measure, instead of the usual minimum support, to generate candidate itemsets, i.e., only items with a lower support than a given threshold are considered. Next, rules are yielded as generated by the Apriori algorithm. A totally different perspective consists of developing a new algorithm to tackle these new challenges. A first proposal is Apriori-Inverse [4], which can be seen as a more intricate variation of the traditional Apriori algorithm. It also uses the maximum support but proposes three different kinds of additions: fixed threshold, adaptive threshold and hill climbing. The main idea is that given a user-specified maximum support threshold, $MaxSup$, and a derived $MinAbsSup$ value, a rule X is rare if $Sup(X) < MaxSup$ and $Sup(X) > MinAbsSup$. A second proposal is the Apriori-Rare algorithm [11], also known as Arima, which is another variation of the Apriori approach. Arima is actually composed of two different algorithms: a naïve one, which relies on Apriori and hence enumerates all frequent itemsets; and MRG-Exp, which limits the considerations to frequent itemsets generators only. Finally, please notice that the first two approaches (Apriori-Frequent and Apriori-Infrequent) are taken to ensure that rare items are also considered during itemset generation, although the two latter approaches (Apriori-Inverse and Apriori-Rare) try to encourage low-support items to take part in candidate rule generation by imposing structural constraints.

The algorithms aforementioned are the most important RARM proposals. Next, we will explore how these approaches can be applied over educational data in such a way that their usefulness in this research area is shown.

3 Experimentation and Results

In order to test the performance and usefulness of applying RARM to e-learning data, we have used student data gathered from the Moodle system to compare several ARM and RARM algorithms and show examples of discovered rules.

3.1 Experimentation

The experiments were performed using data from 230 students in 5 Moodle courses on computer science at the University of Córdoba. Moodle (<http://moodle.org>) is one of the most frequently used free Learning Content Management Systems (LCMS) and keeps detailed logs of all activities that students perform (e.g., assignments, forums and quizzes). This student usage data has been preprocessed in order to be transformed into a suitable format to be used by our data mining algorithms [10]. First, a summary table (see Table 1) has been created, which integrates the most important information about the activities and the final marks obtained by students in the courses. Notice that we have transformed all the continuous attributes into discrete attributes that can be treated as categorical attributes. Discretization allows the numerical data to be divided into categorical classes that are easier for the instructor to understand.

Name	Description	Values
course	Identification number of the course.	C218, C94, C110, C111, C46
n_assignment	Number of assignments done.	ZERO, LOW, MEDIUM, HIGH
n_quiz	Number of quizzes taken.	ZERO, LOW, MEDIUM, HIGH
n_quiz_a	Number of quizzes passed.	ZERO, LOW, MEDIUM, HIGH
n_quiz_s	Number of quizzes failed.	ZERO, LOW, MEDIUM, HIGH
n_posts	Number of messages sent to the forum.	ZERO, LOW, MEDIUM, HIGH
n_read	Number or messages read on the forum.	ZERO, LOW, MEDIUM, HIGH
total_time_assignment	Total time spent on assignments.	ZERO, LOW, MEDIUM, HIGH
total_time_quiz	Total time spent on quizzes.	ZERO, LOW, MEDIUM, HIGH
total_time_forum	Total time spent on forum.	ZERO, LOW, MEDIUM, HIGH
mark	Final mark obtained by the student in the course.	ABSENT, FAIL, PASS, EXCELLENT

Table 1. Attributes used for each student instance

Due to the way their values are distributed, the course and mark attributes are clearly imbalanced, i.e., they have one or many values with a very low percentage of appearance:

- **Course:** From a total of 230 students, 80 took course 218 (34.78%), 66 students did course 94 (28.69%), 62 students did 110 (26.95%), 13 students took course 111 (5.65%) and 9 students took course 46 (3.91%). Thus, there are three predominant courses (C218, C94 and C110) and two minority courses (C111 and C46).
- **Mark:** From among 230 students, 116 students PASS the final exam with a normal/medium score (50.43%), 87 students FAIL the exam (38.82%), 15 students obtain an EXCELLENT or very good/high score in the exam (6.52%) and 12 students were ABSENT from the exam (5.21%). So, there are two majority marks (PASS and FAIL) and two minority marks (EXCELLENT and ABSENT).

A better view of such imbalanced value distribution for these two attributes (mark and course) can be seen graphically in Figure 2.

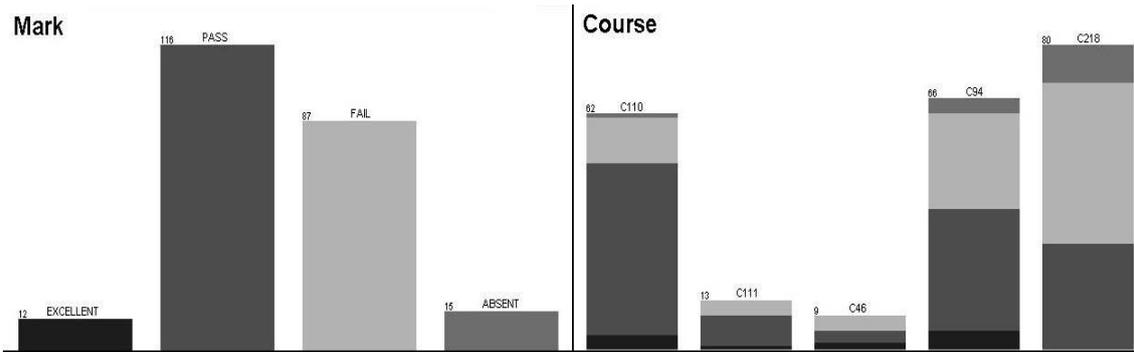


Figure 2. Value distribution for the attributes Mark and Course. The different colours on the right image correspond to the different Marks.

We performed a comparison between ARM and different RARM algorithms to discover rare class association rules [12] from the aforementioned data. A class association rule is a special subset of association rules with the consequent of the rule limited to a target class label (only one predefined item), whereas the left-hand may contain one or more attributes. It is represented as $A \rightarrow C$, where A is the antecedent (in our case, the course and activity attributes) and C is the class (in our case, the mark attribute). This type of rule is more easily understood than general association rules, since it only comprises one element in the consequent and usually represents discovered knowledge at a high level of abstraction, and so can be used directly in the decision making process [8]. In the context of EDM, class association rules can be very useful for educational purposes, since they show any existing relationships between the activities that students perform using Moodle and their final exam marks. To obtain class association rules we need to filter the resulting rules from the ARM or RARM algorithms, so we only select those rules that have a single attribute (i.e., the mark attribute) in their consequent.

We evaluated the four different Apriori proposals following the configuration parameters stated below:

- Apriori-Frequent [2], setting the minimum support threshold at a very low value (0.05);
- Apriori-Infrequent, setting the maximum support at 0.1;
- Apriori-Inverse and Apriori-Rare, using the same support threshold set at 0.1.

We also assigned the value 0.7 as the confidence threshold for all these algorithms.

3.2 Evaluation of results

Table 2 summarizes the results obtained from the four Apriori proposals, and shows the number of frequent and infrequent itemsets mined, the number of rules discovered, and their average support and confidence.

Algorithm	# Freq. Itemsets	# UnFreq. Itemsets	# Rules	Avg Support/ ± Std Deviation	Avg Confidence/ ± Std Deviation
Apriori-Frequent	11562	--	788	0.162±0.090	0.717±0.211
Apriori-Infrequent	--	1067	388	0.058±0.060	0.863±0.226
Apriori-Inverse	--	3491	46	0.056±0.070	0.883±0.120
Apriori-Rare	--	5750	44	0.050±0.080	0.885±0.108

Table 2. Comparison of ARM and RARM proposals.

Notice that the Apriori-Frequent is the only algorithm that uses frequent itemsets. Therefore, it discovers the greatest number of rules (both frequent and rare) with the highest average support but not the highest confidence. This means that the instructor needs to search manually for the rare rules. On the other hand, Apriori-Infrequent mines the smallest number of infrequent itemsets. Though it discovers a great number of rare rules, most of them are redundant. Finally, Apriori-Inverse and Apriori-Rare behave in a very similar fashion and are the best at discovering rare association rules, since they use a higher number of infrequent items than Apriori-Infrequent and discover a lower number of rare rules. A lower number of rules is easier than a higher number of rules for the instructor to use and understand. Furthermore, the standard deviation shows that both Apriori-Inverse and Apriori-Rare tend to be very close to the average, so one expects to obtain rules that will not vary much from these values.

3.3 Examples of discovered rules

Due to the imbalanced nature of the data source, different versions of the conditional support were defined. Conditional support is a well-known measure for the processing of imbalanced data using class association rules [12]. Thus, three different measures are considered to evaluate rule support, as defined in continuation:

- The traditional support of a rule $A \rightarrow C$, with A as the antecedent and C as the consequent, is defined as $Sup(A \rightarrow C) = \frac{n(A \cap C)}{N}$, where $n(A \cap C)$ is the number of instances that matches both the antecedent and consequent, and N is the total number of instances.

However, the support of rules that contain course and mark attributes (imbalanced attributes) must be defined as follows:

- The conditional support with respect to the mark of a class association rule $A \rightarrow Mark$, where $Mark$ stands for the imbalanced attribute mark, and is defined as $SupM(A \rightarrow Mark) = \frac{n(A \cap Mark)}{n(Mark)}$, where $n(A \cap Mark)$

is the number of instances that matches both the antecedent and consequent and $n(Mark)$ is the number of instances that matches the "mark" attribute.

- The conditional support with respect to the course of a class association rule $A \cap Course \rightarrow Mark$, where $Course$ stands for the imbalanced attribute course and $Mark$ for the class attribute, is defined as

$$SupC(A \cap Course \rightarrow Mark) = \frac{n(A \cap Course \cap Mark)}{n(Course)}, \text{ where } n(Course)$$

is the number of instances that matches the “course” attribute.

Next, there are some examples of rules that were mined using both the ARM and the different RARM algorithms. For each rule, we show the antecedent and the consequent constructed, as well as the support and confidence measures. Firstly, Table 3 shows some representative association rules mined using the Apriori-Frequent algorithm. A further description is detailed below.

Rule	Antecedent	Consequent	Sup	SupC/SupM	Conf
1	total_time_forum=HIGH	mark=PASS	0.24	--/0.47	0.82
2	n_posts=MEDIUM AND n_read=MEDIUM AND n_quiz_a=MEDIUM	mark=PASS	0.13	--/0.25	0.71
3	course=C110 AND n_assignment=HIGH	mark=PASS	0.14	0.52/0.27	0.89
4	total_time_quiz=LOW	mark=FAIL	0.21	--/0.55	0.78
5	n_assignment=LOW	mark=FAIL	0.23	--/0.60	0.70
6	n_quiz_a=LOW AND course=C218	mark=FAIL	0.18	0.51/0.47	0.83

Table 3. Rules extracted using the Apriori-Frequent algorithm.

As can be seen, all the rules discovered (not only the 6 rules shown in Table 3 but also the other 788 rules discovered) contain only frequent itemsets, such as mark=PASS (students who passed the exam), mark=FAIL (students who failed), course = 119 (students who took the course 119), course=218 and course=94. Secondly, we can see that these rules have low support (but not very low), a medium value in the two conditional supports and are of high confidence (but not very high). Finally, to explain the usefulness of these rules for the instructor, we are going to describe their meaning. Rule 1 shows that if students spend a lot of time in the forum (a high value) then they pass the final exam. It provides information to the instructor about how the forum has been a good activity for students with a confidence of 0.82. Rule 2 shows that if students have submitted and read messages to/from the forum, and they have passed quizzes, then they have passed the exam. The information provided is similar to the previous data but adds the quizzes as another determining factor in the final mark (as is logical). Rule 3 shows that students in course 110 who sent in many assignments then passed the final exam (rule 5 is the opposite version but for any course). So, the number of assignments is directly related to the final mark. Rule 4 and 5 show that if the total time in quizzes is low or the number of passed quizzes is low (and the course is 218), then students obtain a bad mark. So, quizzes are also directly related to the mark and can be used to detect in time students at risk of failing the final exam.

Next, Table 4 shows some representative rare association rules obtained using the Apriori-Rare algorithm. Please notice that due to the Apriori-Inverse approach obtains almost the same set of rules, so we don't present another analysis similar to the following table. A detailed description of this rule set is presented below.

Rule	Antecedent	Consequent	Sup	SupC/SupM	Conf
1	n_quiz=HIGH AND n_quiz_a=HIGH	mark=EXCELLENT	0.045	--/0.69	0.86
2	total_time_assignment=HIGH	mark=EXCELLENT	0.045	--/0.69	0.86
3	n_posts=HIGH AND course=C46	mark=EXCELLENT	0.045	1.00/0.69	1.00
4	total_time_assignment=ZERO AND total_time_forum=ZERO AND total_time_quiz=ZERO]	mark=ABSENT	0.050	--/0.76	0.78
5	n_posts=ZERO AND n_read=ZERO	mark=ABSENT	0.050	--/0.76	0.78
6	n_quiz=ZERO AND course=C111	mark=ABSENT	0.050	0.88/0.76	1.00

Table 4. Rules extracted using the Apriori-Rare algorithm.

As can be seen, all the rules that are discovered (not only the 6 rules shown in Table 4 but also the other 44 rules discovered) contain only infrequent itemsets, such as mark=EXCELLENT (students who passed the exam with a very high score), mark=ABSENT (students who did not take the exam), course = 46 and course = 111 (students who did courses 46 and 111 respectively). We can see that these rules have a very low support, a very high confidence level (the maximum value) and also a high value for the conditional supports; indicating that they are rare/infrequent rules and their data is imbalanced with respect to the course and mark attributes. To explain the usefulness of these rules for the instructor, we are going to describe their meaning. Rule 1 shows that if students execute all the quizzes and pass them, then they obtain an excellent score in the final exam. It could be an expected rule that shows the instructor that quizzes can be used in order to predict very good student results. Rule 2 shows that if students spend a lot of time on assignments, they obtain an excellent score. This is the opposite of rule 5 in Table 3 and so it proves again that the number of assignments is directly related to the final mark. Rule 3 shows that if students in course 46 send a lot of messages to the forum, they obtain an excellent score. The instructor can use this information to detect very good students in course 46 depending on the number of messages they send to the forum. The last three rules are about students who have been absent for the exam. They show the instructor that if students do not spend time on assignments, forum participation and quizzes, then they do not take the exam. The instructor can detect this type of student in time to help him/her to take part in course activities and also do the final exam.

Finally, we have also compared the values of the evaluation measures, shown in Table 3 and Table 4. Firstly, we can see that confidence values (*Conf*) are normally higher in Table 4 (rare association rules) than in Table 3 (frequent association rules). Secondly, the support values (*Sup*) of rare association rules are much lower in Table 4 than the support of frequent association rules in Table 3. Then, we can see that relative support

values ($SupC$ and $SupM$) of rare association rules are higher in Table 4 than the relative support of frequent ones in Table 3. It proves that rare association rules have high confidence levels, and although they have very low values of support with respect to all the data, these support values are high with respect to imbalanced attributes (as show the relative support measure).

4 Concluding remarks and future work

In this paper we have explored the use of RARM over educational data gathered from the Moodle system installed at the University of Córdoba. The use of this approach has shown to be an interesting research line in the context of EDM, since most real-world data are usually imbalanced. Rare-association rules are more difficult to mine using traditional data mining algorithms, since they do not usually consider class-imbalance and tend to be overwhelmed by the major class, leaving the minor class to be ignored. In fact, we have shown that the regular Apriori algorithm [2] (known as Apriori-Frequent) discovers a huge number of rules with frequent items. Hence we explored how some specific algorithms, such as Apriori-Inverse and Apriori-Rare, are better at discovering rare-association rules than other non-specific algorithms, such as Apriori-Frequent and Apriori-Infrequent. In fact, the set of rules discovered by Apriori-Rare are included into the set of rules discovered by Apriori-Inverse but they are included neither into the set of rules discovered by Apriori-Infrequent nor Apriori.

Finally, we have shown how the rules discovered by RARM algorithms can help the instructor to detect infrequent student behavior/activities in an e-learning environment such as Moodle. In fact, we have evaluated the relation/influence between the on-line activities and the final mark obtained by the students.

In the future, we would like to develop a new algorithm specifically to discover RARM using evolutionary algorithms, and to compare its performance and usefulness in e-learning data versus the previous algorithms. We also plan to explore the use of other different rule evaluation measures for rare association rule mining.

Acknowledgment

This research is supported by projects of the Regional Government of Andalusia and the Ministry of Science and Technology, P08-TIC-3720 and TIN2008-06681-C06-03 respectively, and FEDER funds.

References

- [1] Adda, M., Wu, L, Feng, Y. Rare itemset mining. In *Sixth Conference on Machine Learning and Applications*. 2007, Cincinnati, Ohio, USA, pp-73-80.
- [2] Agrawal, R., Imielinski, T., Swami, A. Mining association rules between sets of items in large databases. *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Washington, DC, 1993, pp. 207–216.
- [3] Kotsiantis, S., Kanellopoulos, D. Association rules mining: A recent overview. *International Transactions on Computer Science and Engineering Journal*, 2006. 32, 1, pp. 71-82.

- [4] Koh, Y., Rountree, N. Finding sporadic rules using Apriori-Inverse. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. 2005. Berlin, pp. 97-106.
- [5] Koh, Y., Rountree, N. Rare association rule mining and knowledge discovery. 2009. *Information Science Reference*.
- [6] Liu, B., Hsu, W., Ma, Y. Mining association rules with multiple minimum supports. In *Proceedings of fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 1999, New York, USA, pp. 337-341.
- [7] Merceron, A., and Yacef, K, Interestingness Measures for Association Rules in Educational Data. In *International Conference on Educational Data Mining*. 2008. Montreal, Canada, pp. 57-66.
- [8] Romero, C., Ventura, S., Bra, P. D. Knowledge discovery with genetic programming for providing feedback to courseware author. *User Modeling and User-adapted Interaction: The Journal of Personalization Research*, 2004, 14 (5), pp. 425-464.
- [9] Romero, C., Ventura, S., Espejo, P., Hervás, C. Data Mining Algorithms to Classify Students. In *International Conference on Educational Data Mining*. 2008, Montreal, Canada, pp. 8-17.
- [10] Romero, C., Ventura, S., Salcines, E. Data mining in course management systems: Moodle case study and tutorial. *Computer & Education*, 2008, 51(1), pp. 368-384.
- [11] Szathmary, L., Napoli, A., Valtchev, P. Towards rare itemset mining. In *International Conference on Tools with Artificial Intelligence*, Washington, DC. 2007, pp. 305-312.
- [12] Zhang, H., Zhao, Y., Cao, L., Zhang, C., Bohoscheid, H. Rare class association rule mining with multiple imbalanced attributes. *Rare Association Rule Mining and Knowledge Discovery*. 2009. Information Science Reference.
- [13] Zaïane, O., Building a Recommender Agent for e-Learning Systems. *Proceedings of the International Conference in Education*. 2002, New Zealand, pp. 55-59.

Using Text Replay Tagging to Produce Detectors of Systematic Experimentation Behavior Patterns

Michael A. Sao Pedro¹, Ryan S.J.d. Baker^{2,1}, Orlando Montalvo², Adam Nakama²,
and Janice D. Gobert^{2,1}

{mikesp, rsbaker, amontalvo, nakama, jgobert}@wpi.edu

¹Computer Science Department, Worcester Polytechnic Institute

²Social Science and Policy Studies Department, Worcester Polytechnic Institute

Abstract. We present machine-learned models that detect two forms of middle school students' systematic data collection behavior, designing controlled experiments and testing the stated hypothesis, within a virtual phase change inquiry learning environment. To generate these models, we manually coded a proportion of the student activity sequence clips using "text replay tagging" of log files, an extension of the text replay method presented in Baker, Corbett & Wagner (2006). We found that feature sets based on cumulative attributes, attributes computed over all predecessor clips, yielded better detectors of CVS-compliant and hypothesis-testing behavior than more local representations of student behavior. Furthermore, our detectors classify behaviors well enough to use them in our learning environment to determine which students require scaffolding on these skills.

1 Introduction

While inquiry learning is generally considered an important part of science education at all levels, many students lack inquiry skills [14]. Students have difficulties focusing on relevant variables, stating testable hypotheses, drawing correct conclusions from experiments, and linking hypotheses and data. They also struggle with basic experimental processes such as designing effective experiments, translating theoretical variables from their hypotheses into manipulable variables, and adequately monitoring their progress [10]. To that end, we are developing a learning environment, Science Assistments (http://users.wpi.edu/~sci_assistments), with the goal of assessing and scaffolding students' scientific inquiry as they engage in inquiry using interactive microworlds [13]. We place special emphasis on students' development of skills for conducting experiments since it has been argued that learning to correctly plan and execute controlled experiments is necessary to the development of other scientific inquiry skills [13]. In order to properly identify students needing inquiry support, we must be able to distinguish students who exhibit appropriate systematic behaviors from those who do not.

In this paper, we present machine-learned models for detecting two forms of systematic data collection behavior exhibited as students conduct experiments in a phase change microworld. The first behavior we detected was designing controlled experiments using the Control of Variables Strategy (CVS) [9], a strategy stating that one should change only the variable to be tested, the target variable, while keeping all extraneous variables constant to test the effects of that variable on an outcome. The second was collecting data to test a stated hypothesis, as opposed to collecting data that does not pertain to the stated hypothesis. To train our behavior detectors we generated training instances by manually inspecting and coding a proportion of student activity sequences using "text replay tagging" of log files. Similar to a video replay or screen replay, a text replay [2] is a pre-specified chunk of student actions presented in text that captures information such as

each action's time, type, widget selection, and input selection. Our approach leverages the success of [4, 6] in using text replays to provide training instances for machine-learned detectors of gaming the system within intelligent tutors. We are building detectors of these constructs, as opposed to detectors that identify specific kinds of unsystematic behavior, with the eventual goal of auto-scoring students' systematicity. Despite the ill defined nature of science inquiry, we can tutor students' inquiry skills with this approach, an approach similar to model tracing (cf. [16]).

This approach differs from previous text replays in two ways. First, whereas text replays allow for the classification of a replay clip as a single category out of a set of categories, text replay tagging allows multiple tags to be associated with one clip. For example, a clip may be tagged as CVS-compliant, hypothesis testing-compliant, both, or neither. Second, the behaviors we are studying are temporally more coarse-grained than in [4] or [6], requiring the display of the entire sequence of work on part of a problem rather than specific attempts to answer a problem or problem step. This permits coders to obtain a more comprehensive view of students' inquiry processes necessary for labeling processes like these that unfold over time. After producing these "gold standard" classifications, we summarized each student's activity sequences by creating a feature set from the data and used classification methods to find models that predict the labels from the data. In accordance with our data, we considered problem-level features of the student data rather than step or transaction-level data, unlike in many prior EDM models of student behavior (e.g. [1, 4, 6, 8, 20]).

Past research has attempted to model and analyze inquiry behavior using knowledge engineering approaches. In [7], the authors defined rules that encapsulated behaviors for differing levels of systematic experimentation skill when solving problems with interactive genetics simulations. They defined their rules over a set of domain-specific features extrapolated from student interactions, such as the types of genetic crosses made and if crosses were repeated, and domain-general features, such as time spent solving a problem. In [19], the authors constructed an ACT-R model based on an assessment of skill differences in novices and experts that designed and interpreted psychological experiments within a computerized environment. They developed a detailed set of rules and hierarchical high and low-level goals and actions to represent the cognitive processes of how an expert hypothesized, explored, analyzed and concluded about two competing theories. Finally, they tested the efficacy of their model by adding and removing key productions and comparing the model's simulated performance to experts and novices. Like both these approaches, we use low-level student actions as a basis for creating our behavioral models and, particularly like [19], we are interested in quantifying how well our detectors predict behaviors. However, our approach is different in that it does not prescribe rules for systematicity; instead, given data, human classified labels, and a feature set, we use machine learning techniques to discover rules. This approach has several advantages. The resulting models capture relationships that humans cannot easily intuit. They also identify boundary conditions more precisely than knowledge engineering approaches. Finally, unlike knowledge engineering approaches, they are easier to verify, since cross-validation is possible.

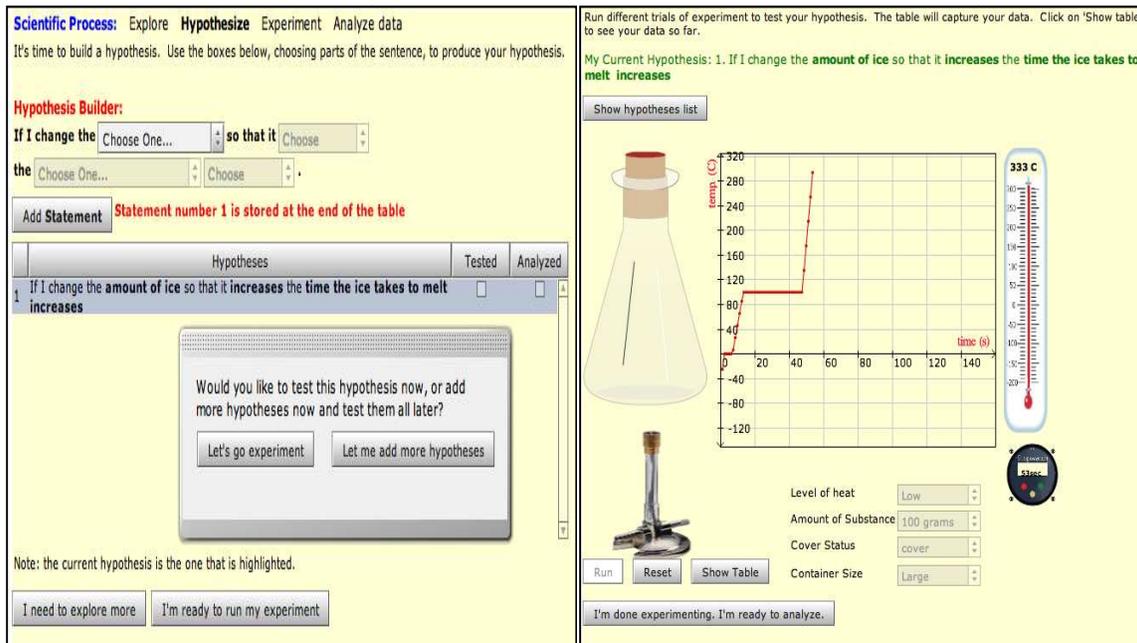


Figure 1. Hypothesizing widget (left) and data collection panel (right) for the phase change microworld.

2 Learning Environment

Our phase change environment (Figure 1), hosted by Science Assistments [13], enabled students to engage in authentic inquiry using a microworld and inquiry support tools. Each problem in our learning environment required students to conduct full experiments to determine if a particular independent variable, e.g., container size, affected various outcomes like the melting point or boiling point of a substance. For a given variable, students demonstrated proficiency by hypothesizing, collecting data, reasoning with tables and graphs, analyzing data, and communicating their findings about how that variable affected the outcomes. We helped organize students' inquiry processes by arranging these tasks into different inquiry phases: "observe", "hypothesize", "experiment", and "analyze data". Students start in the hypothesizing phase and move between phases in a suggested order but can navigate back and forth between some of the inquiry phases. For example, from the "analysis" panel students could collect more data by returning to the "experiment" phases, they could create new hypotheses by returning to the "hypothesize" phase (starting a new inquiry loop), or could submit their final experimentation procedures and analyses and begin the next problem. While in the hypothesizing phase (left side of Figure 1), they could either explore the microworld or begin collecting data in the experiment phase (right side of Figure 1). Finally, within the experiment phase, students can only move to the analysis phase.

This learning environment has a moderate degree of learner control, less than in purely exploratory learning environments [1], but more than in classic model-tracing tutors [16] or constraint-based tutors [18]. Though our scaffolding restricts when students can switch

inquiry phases, there is enough freedom such that students could approach these inquiry tasks in many ways. For example, a student could choose to specify only one hypothesis like, “If I change the container size so that increases, the melting point stays the same” (left side of Figure 1), and then test that single hypothesis. Alternately, they could generate several and test them all sequentially. While experimenting (right side of Figure 1), a student could set up and run as many different experiments as they desired, including repeating the same trial multiple times. A table tool was provided within the learning environment to display the results of the student’s previous experiments and to display their hypothesis list to determine which experiments to run next.

As students engage in inquiry using our tools and microworld, they can exhibit several different inquiry behaviors. Students acting in a systematic manner [7] collect data by designing and running controlled experiments that test their hypotheses. Also, students acting systematically use the table tool and hypothesis viewer in order to reflect and plan for additional experiments. Students who are unsystematic, by contrast, may exhibit haphazard behaviors such as: constructing experiments that do not test their hypotheses, not collecting enough data to support or refute their hypotheses, not following CVS, running the same experimental setup multiple times, or failing to use the inquiry support tools to analyze their results and plan additional trials [10].

3 Dataset

Participants were 148 eighth grade students, ranging in age from 12-14 years, from a public middle school in Central Massachusetts. These students used the phase change microworld as part of a broader study to determine if inquiry skills learned in one domain will transfer to inquiry skill in other domains [13]. Students engaged in authentic inquiry problems using the phase change and density microworlds within the Science Assistments learning environment. Students were randomly assigned to one of two conditions that counterbalanced the order in which students engaged in a science domain: phase change followed by density vs. density followed by phase change. In this paper, we discuss detectors of systematic data collection for student actions within the phase change microworld only, as the version of the density microworld used lacked the hypothesizing scaffold used in the phase change microworld. In building these detectors, we look specifically at what students did in the “hypothesizing” and “experimenting” phases of inquiry. As part of the phase change activities, students attempted to complete four tasks using our interactive tools.

Each of these students completed at least one data collection activity in the phase change environment (two other students did not use the microworld, and were excluded from analysis). As students solved these tasks, we recorded fine-grained actions within the inquiry support tools and microworlds. The set of actions logged included creating hypotheses, setting up experiments, showing or hiding support tools, running experiments, creating interpretations of data, and transitioning between inquiry activities (i.e. moving from hypothesizing to data collection). Each action’s type, current and previous values (where applicable – for instance, a variable’s value), and timestamp were recorded. In all, 27,257 student actions for phase change were logged. These served as

the basis for generating text replay clips consisting of contiguous sequences of actions specific to experimenting.

4 Text Replay Tagging Methodology

In designing our text replays, it was necessary to use a coarser grain-size than in prior versions of this method (e.g. [4, 6]). In particular, it was necessary to show significant periods of experimentation so that coders could precisely evaluate experimentation behavior relative to stated hypotheses. We decided our text replays should include actions from only the hypothesis and the experimenting phases. Another important issue was that trial run data from one hypothesis test could be used in another hypothesis test to make inferences about the hypothesis at hand (i.e. comparing a current trial to one conducted earlier). To compensate for this, we code using both the actions in testing the current hypothesis, and cumulative measures that include actions performed when testing previous hypotheses. Hence, each tagged clip focuses on actions in the current part of the inquiry process, but may take into account the context of the cumulative section.

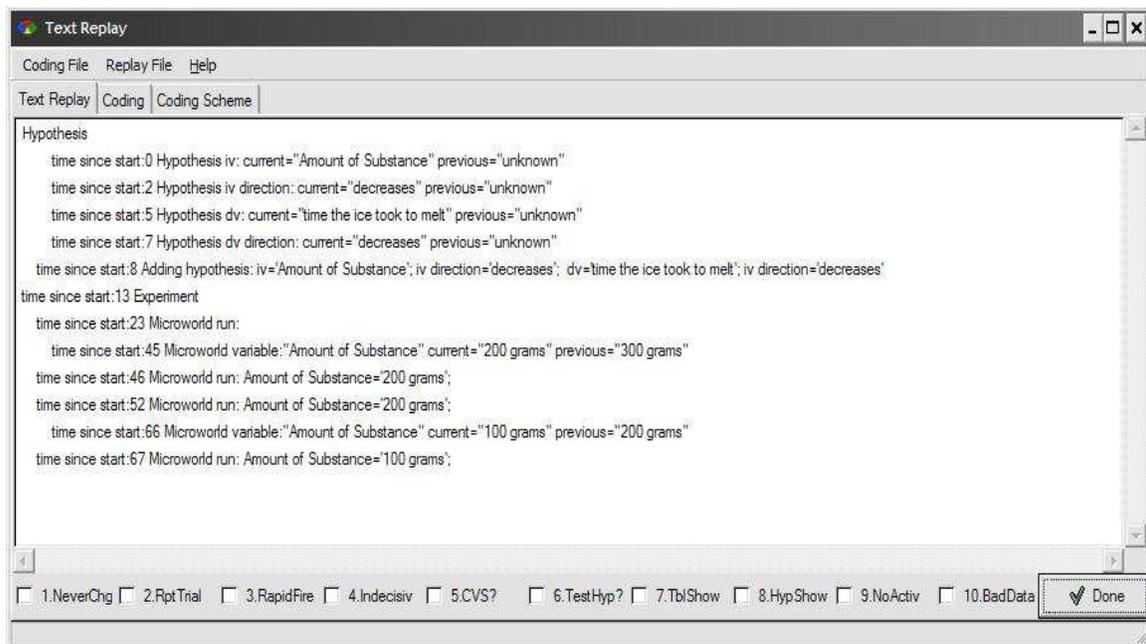


Figure 2. Text Replay Tagging Tool with an example student’s clip coded as running repeated trials, following CVS, and testing stated hypotheses.

To support coding in this fashion, a new tool for text replay tagging was developed in Ruby, shown in Figure 2. The start of the clip is triggered by a hypothesis variable change after the beginning of a new problem. The tool displays all student actions (hypothesis and experiment) until the student transitions to the analysis stage. Subsequent clips include previous clips and any single new cycle which includes the Hypothesis and Experiment stage. A clip could be tagged with one of 9 tags corresponding to data collection behaviors: “Never Change Variables”, “Repeat Trials”, “Non-Interpretable Action Sequence”, “Indecisiveness”, “Used CVS”, “Tested Hypothesis”, “Used Table to Plan”, “Used Hypothesis Viewer to Plan”, “No Activity”,

and one extra category for unclassifiable clips, “Bad Data”, for a total of 10 coding categories. Specifically for the analyses in this paper, we tagged a clip as “Used CVS” if the clip contained actions indicative of designing and running controlled experiments. “Tested Hypothesis” was chosen if the clip had actions indicating attempts to test the *stated* hypotheses, regardless of whether or not proper CVS procedure was used.

4.1 Clip Tagging Procedure

Two coders (the first and fourth authors) tagged the data collection clips using at least one of the ten tags. To ensure that a representative range of student clips were coded, we stratified our sample of the clips on condition, student, problem, and within-problem clip order (e.g. first clip, second clip, etc.) The corpus of hand-coded clips contained exactly one randomly selected clip from each problem each student encountered, resulting in 571 clips. Each coder tagged the first 50 clips; the remaining clips were split between the coders. For the 50 clips tagged by each coder, there was high overall tagging agreement, average $\kappa=0.86$. Of particular relevance to this study, there was also better agreement on the CVS and testing hypotheses tags, $\kappa=.69$ and $\kappa=1.00$ respectively, than has been seen for previous text replay approaches that led to successful behavior detectors (e.g. [4, 6]).

4.2 Feature Distillation

Features were extracted relevant to the 10 categories of behavior within the microworld. These included: all actions, total trial runs, incomplete trial runs, complete trial runs, pauses, data table display, hypothesis list display, field changes in hypothesis builder (left side of Figure 1), hypotheses made, and microworld variable changes. For each category, we traced the number of times the action occurred and the time taken for each action. For timing values, we also computed the minimum, maximum, standard deviation, mean and mode for each student and compared these values relative to all other students. We also included the number of pairwise trials where only one independent variable differed between them and a count for repeated trials, trials with the same independent variable selections. These last two had no time associated with them.

We extracted feature values from student actions as follows. As stated in Section 2, student microworld activity was divided into tasks, each focusing on a specific independent variable. Also, within a task, the student could make and test several hypotheses. For each of the categories, we extracted data for each hypothesis the student tested (local data), and across all hypotheses in the set (cumulative data). We did this because within each set, the data table accumulated the trial run data across hypotheses, enabling students to compare trial runs testing previous hypotheses with the runs made in the current hypothesis.

4.3 Machine Learning Algorithms

Machine-learned detectors of the two behavioral patterns of interest, CVS and hypothesis testing, were developed within RapidMiner 4.6 [17]. Detectors were built using J48 decision trees, with automated pruning to control for over-fitting, the same technique used in [4, 20]. Before running the decision tree algorithm, we filtered redundant features

correlated at or above 0.6. Six-fold cross-validation was conducted at the student level (e.g. 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 groups of students. We assessed the classifiers using two metrics. First, we used A' [15]. A' is the probability that if the detector is comparing two clips, one involving the category of interest (CVS or Hypothesis Testing) and one not involving that category, it will correctly identify which clip is which. A' is equivalent to both the area under the ROC curve in signal detection theory, and to W , the Wilcoxon statistic [15]. A model with an A' of 0.5 performs at chance, and a model with an A' of 1.0 performs perfectly. In these analyses, A' was used at the level of clips, rather than students. Statistical tests for A' are not presented in this paper. The most appropriate statistical test for A' in data across students is to calculate A' and standard error for each student for each model, compare using Z tests, and then aggregate across students using Stouffer's method (cf. [3]) – however, the standard error formula for A' [15] requires multiple examples from each category for each student, which is infeasible in the small samples obtained for each student in our text replay tagging. Another possible method, ignoring student-level differences to increase example counts, biases undesirably in favor of statistical significance.

Second, we used Kappa (κ), which assesses whether the detector identifies is better than chance at identifying the correct action sequences as involving the category of interest. A Kappa of 0 indicates that the detector performs at chance, and a Kappa of 1 indicates that the detector performs perfectly. As Kappa looks only at the final label, whereas A' looks at the classifier's degree of confidence, A' can be more sensitive to uncertainty in classification than Kappa.

5 Results

We constructed and tested detectors using our corpus of hand-coded clips. The CVS and hypothesis testing detectors were constructed from a combination of the subset of the first 50 clips that the two coders agreed on, and the remaining clips, tagged separately by the two coders. Of all clips, 31.2% were tagged as showing evidence of CVS and 34.4% were tagged as showing evidence of collecting data to test specified hypotheses.

Detectors were generated for each behavior using J48 decision trees and two sets of attributes, cumulative and non-cumulative attributes. As a reminder, non-cumulative attributes were tallied over a single clip, irrespective of other clips, whereas cumulative attributes included data from earlier clips from the same problem. Thus, four different detectors were constructed. The CVS detector using cumulative attributes ($A'=.85$, $\kappa=.47$) appeared to perform better than the detector built with non-cumulative attributes ($A'=.81$, $\kappa=.42$). Likewise, the hypothesis testing detector built with cumulative attributes ($A'=.86$, $\kappa=.46$) scored higher on our metrics than the non-cumulative detector ($A'=.84$, $\kappa=.44$). We believe the detectors built from cumulative attributes perform better because students may perform actions within particular clips that, when taken in conjunction with actions from previous clips, represent a more complete picture of student behavior. For example, while analyzing results a student may realize they need to run one more

experiment to correctly test their hypothesis (which would start a new clip). The human coders would correctly label this as CVS and testing a hypothesis *in reference to the previous context*, but values for noncumulative attributes would most likely indicate that the student was not systematic because these attributes' values are not computed based on previous clips.

6 Discussion and Conclusions

The goal of this research was to develop machine-learned models that can automatically detect if a student is systematic in their inquiry, particularly in their data collection actions, using text replay tagging. This work showed that combining text replay clip tagging of low-level student actions and machine learning can lead to the successful development of behavior detectors in an ill-defined domain such as scientific experimentation. This work also presents a contribution to the text replay process since it is more efficient to code a clip with multiple tags. Our results were promising; using cumulative attributes, we can distinguish students who are successfully applying the Control of Variables Strategy (CVS) in the phase change environment from students not applying CVS 85% of the time and can distinguish students testing their hypotheses 86% of the time. Furthermore, the Kappa values indicate that each of these detectors are substantially better than chance. In other words, though these detectors are not perfect, they can be used to select students for scaffolding. Since they are not perfect, some students may receive help when they do not need it and vice versa. Hence, interventions used should be fail-soft, relatively non-harmful when given incorrectly. As such, we aim to use these detectors to determine which students will receive scaffolding.

An important area of future work will be to improve our detectors' A' and Kappa. To this end, we plan to add lesson-wide attributes, learner attributes, and data on the other tags used to critique a clip. Lesson-wide attributes, such as task attempt number, that can benchmark a students' experience within our environment may aid in predicting systematicity, in coordination with other features. Additionally, rather than treating learner characteristics, such as prior knowledge, as external predictors of systematicity, we could incorporate those measurements into the detectors themselves. Similar to computing average attribute differences between clips (i.e. computing the difference in number of trials run for the given clip and the average number of trials run for all clips), we could compute differences between students with similar learner characteristics. Similarly, rather than using systematicity to predict content knowledge, we could incorporate student prior knowledge of content and inquiry using our standardized-test style questions [13]. Another important area of future work will be to generalize and train our detectors across different microworlds (cf. [5]) to increase their applicability across middle school science learning.

This approach also enables us to research the interactions between content knowledge and authentic inquiry performance within our learning environments. Being able to classify students as systematic according to different skills, e.g. testing hypotheses and CVS, will enable us to determine if skill proficiency in solving authentic inquiry problems will predict skill proficiency in solving standardized test-style inquiry questions. We can also determine the degree to which systematic behavior predicts robust

content knowledge. Finally, by developing and generalizing detectors across domains, we can determine the degree to which authentic inquiry skill transfers between domains. As such, these models have considerable potential to enable future “discovery with models” analyses that can shed light on the relationship between a student’s mastery of systematic experimentation strategies and their domain learning.

Acknowledgements

This research is funded by the National Science Foundation (NSF-DRL#0733286) and the U.S. Department of Education (R305A090170). Any opinions expressed are those of the authors and do not necessarily reflect those of the funding agencies.

References

- [1] Amershi, S., Conati, C. Combining Unsupervised and Supervised Machine Learning to Build User Models for Exploratory Learning Environments. *Journal of Educational Data Mining*, 2009, 1(1).
- [2] Baker, R. S. J. d., Corbett, A. T., Wagner, A. Z. Human Classification of Low-Fidelity Replays of Student Actions. *Proceedings of the Educational Data Mining Workshop at the 8th International Conference on Intelligent Tutoring Systems*, 2006. p. 29-36.
- [3] Baker, R. S. J. d., Corbett, A. T., Aleven, V. Improving Contextual Models of Guessing and Slipping with a Truncated Training Set. *Proceedings of the 1st International Conference on Educational Data Mining*, 2008. p. 67-76.
- [4] Baker, R. S. J. d., de Carvalho, A. M. J. A. Labeling Student Behavior Faster and More Precisely with Text Replays. *Proceedings of the 1st International Conference on Educational Data Mining*, 2008. p. 38-47.
- [5] Baker, R., Corbett, A., Roll, I., Koedinger, K. Developing a generalizable detector of when students game the system. *User Modeling and User-Adapted Interaction*, 2008, 18(3), p. 287-314.
- [6] Baker, R., Mitrovic, A., Mathews, M. Detecting Gaming the System in Constraint-Based Tutors. *Proceedings of the 3rd International Conference on User Modeling and Personalization*, in press.
- [7] Buckley, B. C., Gobert, J., Horwitz, P. Using log files to track students' model-based inquiry. *Proceedings of the 7th International Conference on Learning Sciences*, 2006. p. 57-63.
- [8] Cetintas, S., Si,, Xin, Y. P., Hord, C. Automatic Detection of Off-Task Behaviors in Intelligent Tutoring Systems with Machine Learning Techniques. *IEEE Transactions on Learning Technologies*, in press.

- [9] Chen, Z., Klahr, D. All Other Things Being Equal: Acquisition and Transfer of the Control of Variables Strategy. *Child Development*, 1999, 70(5), p. 1098-1120.
- [10] de Jong, T. Computer Simulations - Technological Advances in Inquiry Learning. *Science*, 2006, 312, p. 532-533.
- [11] Gobert, Janice (Principal Investigator); Heffernan, Neil; Koedinger, Ken; Beck, Joseph (Co-Principal Investigators): ASSISTments Meets Science Learning (AMSL; R305A090170). Awarded February 1, 2009 from the U.S. Dept. of Education, 2009
- [12] Gobert, Janice (Principal Investigator); Heffernan, Neil; Ruiz, Carolina; Kim, Ryung (Co-Principal Investigators): AMI: ASSISTments Meets Inquiry (NSF-DRL# 0733286). Awarded September 2007 from the National Science Foundation, 2007
- [13] Gobert, J., Heffernan, N., Feng, M., Sao Pedro, M., Beck, J. ASSSTments for Science and Math: Assessing and Assisting. *Presented at the Annual Meeting of the American Educational Research Association. San Diego, CA, April 13-17, 2009.*
- [14] Gobert, J., Schunn, C. Supporting Inquiry Learning: A Comparative Look at What Matters. *A symposium presented at the Annual Meeting of the American Educational Research Association. Chicago, IL, April 9-13, 2007.*
- [15] Hanley, J. A., McNeil, B. J. The Meaning and Use of the Area under a Receiver Operating Characteristic (ROC) Curve. *Radiology*, 1982, 143, p. 29-36.
- [16] Koedinger, K. R., Corbett, A. T.: Cognitive Tutors: Technology Bringing Learning Sciences to the Classroom. In Sawyer, R. K. (Eds.) *The Cambridge Handbook of the Learning Sciences*, 2006. New York: Cambridge University Press. p. 61-77.
- [17] Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., Euler, T. YALE: Rapid Prototyping for Complex Data Mining Tasks. *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006)*, 2006. p. 935-940.
- [18] Mitrovic, A., Mayo, M., Suraweera, P., Martin, B. Constraint-Based Tutors: A Success Story. Springer-Verlag (Eds.) *Proceedings of the Industrial & Engineering Application of Artificial Intelligence & Expert Systems Conference IEA/AIE-2001*, 2001. p. 931-940.
- [19] Schunn, C., Anderson, J.: Scientific Discovery. In Anderson, J. (Eds.) *The Atomic Components of Thought*, 1998. Mahwah: Lawrence Erlbaum Associates, Inc. p. 385-428.
- [20] Walonoski, J. A., Heffernan, N. T. Detection and Analysis of Off-Task Gaming Behavior in Intelligent Tutoring Systems. *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, 2006. p. 382-391.

Identifying High-Level Student Behavior Using Sequence-based Motif Discovery

David H. Shanabrook¹, David G. Cooper², Beverly Park Woolf², and Ivon Arroyo²
dhshanab@acad.umass.edu

¹Department of Education, University of Massachusetts, Amherst

²Computer Science Department, University of Massachusetts, Amherst

Abstract. We describe a data mining technique for the discovery of student behavior patterns while using a tutoring system. Student actions are logged during tutor sessions. The actions are categorized, binned and symbolized. The resulting symbols are arranged sequentially, and examined by a motif discovery algorithm to detect repetitive patterns, or motifs, that describe frequent tutor events. These motifs are examined and categorized as student behaviors. The categorized motifs can be used in real-time detection of student behaviors in the tutor system.

1 Introduction

Tutoring systems have demonstrated effective learning in the classroom [11]. However, even the most effective tutoring system will fail if the student's behavior is not receptive to the material being presented. For example, lack of motivation has been shown empirically to correlate with a decrease in learning rate [2]. In addition, students often use tutors ineffectively and adopt behavioral strategies that allow them to avoid learning, e.g., deliberately entering incorrect answers to elicit hints and, eventually, the correct answer from the tutor [2]. Although tutor instruction is beneficial, its effectiveness might be increased if maladaptive student behaviors could be identified [4].

Intelligent tutoring systems can identify student behaviors that are likely to be unproductive in the long term. The potential to help students is much greater if their keystroke information is logged to provide teachers with real-time data on students' performance, formative assessments, instead of summative assessments [8]. Recent research has utilized dynamic assessment of students' performance to enhance the effectiveness of their tutor sessions [4].

Previous methods for examining behavioral trends have focused on behaviors that are correlated to specific outcomes. One example is correlating engagement (e.g. fluctuations in attention, willingness to engage in effortful cognition, etc.) with successful and unsuccessful problem solving [3]. A potential drawback of this traditional approach is that behavior patterns during tutor usage can be masked in the overall data.

The work presented here approaches the goal of understanding student use/misuse of tutoring systems in a different way. Instead of correlating to specific outcomes, we data-mine patterns of student behavior that are frequent. Interesting frequent patterns over series of student data emerge. This process is a variation of time-based motif discovery [7]. Continuous variables of student tutor progress are binned into discrete categories represented by "a, b, c ..." These letters chronologically form a single string that represents one student's actions over hours of tutor use. Concatenating these strings we can create a single string representing the work of hundreds of students in several schools over several work sessions. Motif discovery is applied and the discovered patterns are examined and categorized qualitatively. The discovered patterns are a potential input for tutor interventions.

2 Relevant Literature

Several models have been proposed to infer high-level student behavior, such as student motivation from behavioral measures. A latent response model [2] was learned to classify student actions as either gaming or not gaming the system. Two cases of gaming were identified: gaming with no impact on pretest-posttest gain and gaming with a negative impact on pretest-posttest gain. The latent response model features consisted of a student's actions in the tutor, such as response time, and probabilistic information regarding a student's latent skills. Beck [4] proposed a function relating response time to the probability of a correct response to model student disengagement in a reading tutor. He adapted the item characteristic curve from Item Response Theory (IRT) to include a student's speed, proficiency, response time, and other problem-specific parameters. The learned model showed that disengagement negatively correlated with performance gain. These models embody different assumptions about the variables required to estimate student motivation (e.g. static versus dynamic models, complex versus simple features, user specified versus learned model parameters, generic versus domain specific models).

A dynamic mixture model was proposed that used a student's behavior to disambiguate between proficiency, modeled as a static, continuous variable and motivation, modeled as a dynamic, discrete variable [6]. These assumptions were based on a student's tendency to exhibit different behavioral patterns over the course of a tutoring session. The model suggested four novel principles: the model should estimate both student motivation and proficiency, run in real time, be able to easily include other forms of unmotivated behavior, and motivation should be treated as a dynamic variable. Empirical evidence suggests that a student's motivation level tends to ebb and flow in spurts [6].

Following the above literature, our method examines student interaction with the tutor during problem solving. However, rather than correlating behaviors with outcomes, we examine frequent behaviors to find meaning in the behaviors on their own. This involves four steps: 1) The raw tutor data is binned and categorized both by hand and statistically. This results in a string of symbols representing the tutor interactions of all students concatenated. 2) The discovery algorithm searches for reoccurring motifs representing student actions over 10 problem intervals. 3) The discovered motifs are consolidated (combined) and categorized by hand, so that they describe high-level student behaviors. 4) These motifs are applied to the original student log and predict student behaviors during tutor usage. In the future, these will be utilized within the tutor to predict real-time student behaviors and correlated to performance outcomes.

3 Tutor description and method

Wayang Outpost is an adaptive tutoring system that helps students learn to solve standardized-test type of questions, in particular state-based exams taken at the end of high school in the USA. This multimedia tutoring system teaches students how to solve geometry, statistics and algebra problems of the type that commonly appear on standardized tests. To answer problems in the Wayang interface, students choose a solution from a list of multiple choice options. Students are provided immediate feedback when they click on an answer (a check for correct or a cross for incorrect). Students may click on a help button for hints, and

teachers/researchers encourage them to do so as many times as necessary, as hints are displayed in a progression from general suggestions to bottom-out solution.

Despite efforts to emphasize the importance of going slowly through problems, students tend to “game” the system by using a variety of speeding strategies. Students either click through answers fast in order to get the immediate reward of a correct answer (a green check); or they skip problems without making any attempts; or they click through hints to get to the “bottom-out” hint that reveals the correct choice. Decisions about content sequencing are based on a model of student effort, used to assess the degree of cognitive effort a student invests to develop a problem solution [1].

The data involved in this paper comes from 250 high school students from a variety of math classes in public high schools during Spring 2009. Students came to the computer lab to use Wayang Outpost during that spring semester for about a week, one-hour periods approximately, instead of their regular math class. Students went through various topics such as perimeter problems, area problems, angles, triangles, Pythagorean theorem, etc. The first day and the last day students took a mathematics pretest and posttest. For some students, the topics in Wayang were a review to their math class, to others it was a way to encounter new concepts (there is a short tutorial at the beginning of each topic) and for others it was a way to practice strategies for their upcoming standardized state-wide exams.

4 Data binning and categorization

During interaction with our tutoring system, each student’s actions are logged in a central database. We focus on the data collected during the course of a problem. We utilize four metrics: hints seen (hints), seconds to first attempt (secFirst), seconds between subsequent attempts (secOther), and incorrect attempts (numIncorrect). These metrics are manually binned based on the meaning of the value. For example, there are two indicators for hints seen in our database. There is the count of hints, and there is an indicator that the last hint was seen. From this we have three bins. No hints seen, some hints seen, and last hint solved (because the last hint reveals the answer).

Once the metrics are binned, each problem is represented as a four character *problem string*. For any given student, the tutor interaction can be summarized by the sequence ordered concatenation of problem strings that we will call a *student string*. The problem string construction and meaning is discussed in the following paragraphs.

In motif discovery, data is sometimes binned into discrete categories to significantly reduce the data footprint in memory. Our reasons for binning are to increase the clarity and meaning of the data; with our understanding of the tutor we categorize each metric so that the bins more clearly describe student behavior. Each metric is categorized in three or five categories represented by a single character from 'a' through 'q', as follows:

hints (a, b, c) – Hints is a measure of the number of hints viewed for this problem. Although each problem has a maximum number of hints, the hint count does not have an upper bound because students can repeat hints and the count will increase at each repeated view. The three categories for hints are: (a) no hints, meaning that the

student did not use the hint facility for that problem, (b) meaning the student used the hint facility, but was not given the solution, and (c) last hint solved, meaning that the student was given the solution to the problem by the last hint. As described above, this metric combines two values logged by the tutor: the count of hints seen, and an indicator that the final hint giving the answer was seen. The data could have been simply binned low, medium, high hints; however, this would have missed the significance of zero hints and using hints to reveal the problem solution.

secFirst (d, e, f) – The seconds to first attempt is an important measure as it is during this time that the student is reading the problem and formulating their response. In previous research [6], five seconds was determined to be a threshold for this metric representing gaming: students who make a first attempt in less than five seconds are considered not working on-task. We divide secFirst into three bins: (d) less than 5 sec, (e) 5 to 30 sec, (f) greater than 30 sec. (d) represents students who are gaming the system, (e) represents a moderate time to the first attempt, (f) represents a long time to the first attempt. The cut at 30 seconds was chosen because it equalizes the distribution of bins (e and f), representing a division between a moderate and a long time to the first attempt.

secOther (g, h, i, j, k) – This variable represents actions related to answering the problem after the first attempt was made. While the first attempt includes the problem reading and solution time, subsequent solution attempts could be much quicker and the student could still be making good effort. secOther is categorized in five bins: (g) skip, (h) solved on first, (i) 0 to 1.2 sec, (j) 1.2 to 2.9 sec, (k) greater than 2.9 sec. First, there are two categorical bins, skip and solve on first attempt. These are each determined from an indicator in the log data for that problem. Skipping a problem implies only that students never clicked on a correct answer; they could have worked on the problem and then given up, or immediately skipped to the next problem with only a quick look. Solved on first attempt indicates correctly solving the problem. If neither of the first two bins are indicated in the logs, then the secOther metric measures the mean time for all attempts after the first. The divisions of 1.2 sec and 2.9 sec for the latter three bins were obtained using the mean and one standard deviation above the mean for all tutor usage; (i) less than 1.2 seconds would indicate guessing, (j) would indicate normal attempts, and (k) would indicate a long time between attempts.

numIncorrect – (o, p, q) - Each problem has four or five possible answer choices, that we divide into three groups: (o) zero incorrect attempts, indicates either solved on first attempt, skipped problem, or last hint solves problem (defined by the other metrics); (p) indicates choosing the correct answer in the second or third attempt, and (q) obtaining the answer by default in a four answer problem or possibly guessing when there is five answer problem.

Some of the bins have dependencies that effect motif discovery and analysis. For example, last hint solves (c) precludes solved on first (h) and skipped (g); solved on first (h) requires zero incorrect attempts (o). In addition, by binning skip (g) in the secOther group, the timing of incorrect attempts is lost when the problem is skipped.

479 student strings representing 3762 total problems were constructed and concatenated into a 15048 character input string for motif discovery. The first 160 characters of the string (40 words/problems, separated at problems for clarity) are:

*afkq bfho cekp bfho aehe cekq bfiq bfkq bfip aehe aehe aeip
 aehe aeip afip cekp aehe afip cfho aehe cfho bfkp bekp bekp
 aehe aekp beho bekp aehe cfho bfkq aekp cehe cfkp bfjp aeip
 bfkp aehe afip afho*

In the first problem, afkq indicates no hints used, greater than 30 seconds to first attempt, over 2.9 mean seconds in other attempts, and most or all choices were made to find the solution. In the next problem, coded bfho, the student asks for one or two hints, greater than 30 seconds to first attempt, then solves the problem on first attempt. The sequence based motif discovery algorithm searches the input string in step two of the process. A detailed description follows.

5 Word-skipping sequence-based motif discovery algorithm

Our sequence-based motif discovery algorithm is a modification of the PROJECTION algorithm [10]. It is similar to the Chiu et al. [5] projection algorithm, except that our sliding window moves per word rather than per character. The PROJECTION algorithm is an efficient way to find planted strings in a long sequence of characters, and our modification to the algorithm allows us to apply it in a multivariate fashion where each character of the word represents a variable. In order to illustrate the algorithm, we first present an example, and then present a formal description of the algorithm.

5.1 Example word skipping projection

Take as input a string words, four characters each, with no separation. Construct a matrix S as a 40 character sliding window that slides 4 characters (1 word) per row. The below 6 x 40 character matrix (Figure 1a) represents the first 64 characters or 16 words. Randomly select 10 columns in the S matrix as the projection highlighted below. Project the selected columns to a new matrix (Figure 1b). If there is a string match between any pair of rows, then add a collision to the collision matrix (Figure 1c). The highlighted rows (3 and 4) match and thus the 3rd row and the 4th column has a collision.

5.2 Random projection multivariate motif discovery algorithm

Function Name: wordMotif

Inputs: word sequence (t), motif size in characters (n), word size (w), projection length (p), number of iterations (m), max character distance (d), number of motifs to find (c).

Outputs: c motif lists with start index and strings of length n for each motif example.

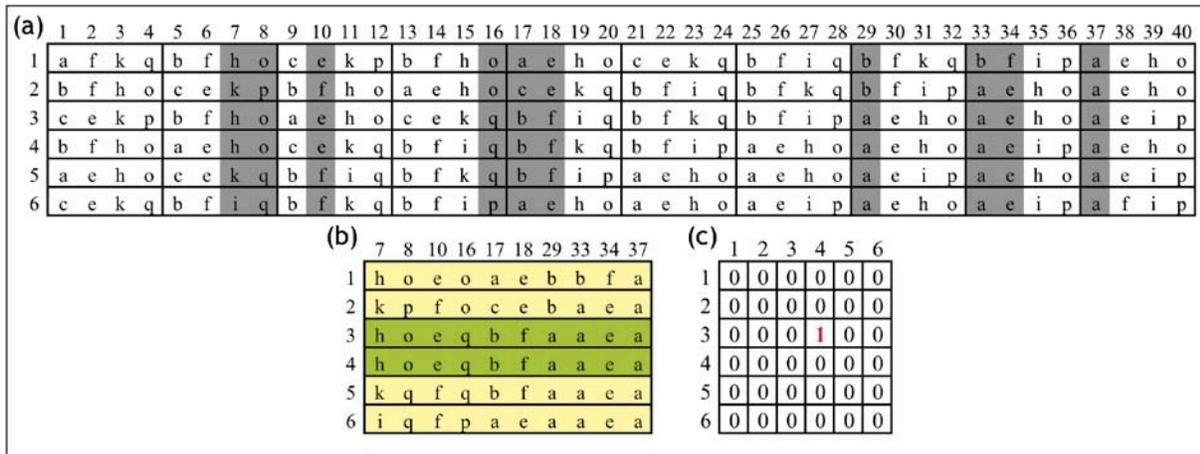


Figure 1. Steps of word skipping projection. (a) the S matrix of 40 character substrings highlighted with a random projection (b) the projected matrix with 2 matching substrings (c) the resulting collision added to the collision matrix.

wordMotif(t,n,w,p,m,d,c)

1. Construct S matrix of size $(t/w * n)$ by sliding an n sized window w characters at a time.
2. Create collision matrix using **project**(S,p,m)
3. Compare pair of examples (A,B) with highest collision value
4. If (A,B) do not overlap and are within a hamming distance of d characters, then test them against other members of the S matrix, adding all members that are within hamming distance d to the motif set, and removing the collision values from the collision matrix.
5. Repeat 3 and 4 until c motifs are found or the collision matrix is exhausted.
6. Return the lists of up to c motifs

project(S,p,m)

1. let k be the number of rows in S
2. construct an empty $k \times k$ collision matrix
3. repeat m times
 - a. make a $k \times p$ matrix based on a random mask of p columns of the S matrix
 - b. add a collision for each pair whose string is equivalent.
4. Return collision matrix

The algorithm outputs a matrix of indices into the string, with a column for each discovered motif. For this study we found thirty motifs. The first two indices of the first 10 motifs are shown in Table 1.

Table 1. The first 2 indices of the first 10 motifs.

M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
2305	6789	8989	8993	9485	2301	18181	29469	29301	48953
6533	7717	9101	9105	11525	18061	19557	49577	58825	67561

Each value in the matrix is an index to the first character of the motif. We can index into the original data to determine the symbols in each of these motifs. The first instance of the motif (in row 1) is representative of the pattern, and the other instances (in row 2, etc.) will be identical or within 10 characters of both the first and the second instance. Table 2 shows the first instance of the first ten motifs.

Table 2. Ten of the motifs The index of the first character is followed by the 40 character motif with a separation each word to see problem characteristics.

M1	2305	adgo									
M2	6789	adgo									
M3	8989	adiq	adgo								
M4	8993	adgo									
M5	9485	adgo									
M6	2301	cdgo	adgo								
M7	18181	adgo	adgo	adgo	adgo	adgo	bdgo	adgo	adgo	adgo	adgo
M8	29469	afho	aeho	aeho	aeho	aeho	aeho	aeho	afho	aeho	aeho
M9	29301	aeho	afho	afho	aeho						
M10	48953	aeho	afho	afho	afho	aeho	aeho	aeho	aeho	afho	aeho

6 Discovered motifs

In the third step, we examine and analyze the discovered motifs to determine the high level behavior that each motif discovered. We group motifs with similar behaviors. For this study we used the algorithm to detect thirty motifs. Chiu et al. [5] suggest a need to eliminate degenerate motifs, which are motifs that have no informational content, such as a sequence of repeated characters. In the case of the tutor data, repeated words are not degenerate because they inform us that the student is repeating a particular behavior. Repetition of undesired behavior is an important feature to capture, so we do not remove such motifs as degenerate.

In the 30 discovered motifs, there were a number of repeated motifs. This is because a number of motifs were essentially straight, i.e. repetitions of a motif word, so there are cases of motifs that have essentially the same structure; these would overlap with each other, so they are considered distinct patterns. By grouping these exact match motifs, and also by comparing the different motifs by eye, we grouped the 30 motifs into 7 distinct *meaning groups* coded g, f, F, k, r, z, n:

Game-like (g). adgo (10), adip (10), or adiq (10) – Student is not reading the problems and either skipping or making quick guesses.

Frustration (guess) (f). adiq (1) adgo (9) – Guessing to find solution, then skipping the next 9 problems. This could be an indication of frustration.

Frustration (hints) (F). cdgo (1) adgo (9) – Using the hints to find solution, then skipping the next 9 problems. This also could indicate frustration, and was grouped together with the previous motif.

Not challenged (k). a[ef]ho (10) – Solving the problem on first attempt, not using hints, not guessing. This student is using the tutor appropriately, but not being challenged.

Too difficult (r). ceho (10) - student is taking time to read the problem, then **using hints** to find the answer. These students could be working but the material is too difficult for them to solve the problem themselves.

Skipping (z). adgo (5) aeho (5) – Student skips 5 problems, then solves 5 normally on first attempt. Taking time to read all problems, then answering or skipping depending on whether the answer is known.

On-task(n) aeiq aeho aeho aekp aeho aeiq aeho aeip aeho aeip - This is the most complex motif found and seems to indicate “on task learning” tutor usage; the student is always reading the problem and making a good first attempt, with a mixture of solving on first attempt (aeho), solving after some attempts (aekp, aeip) and guessing (aeiq). The student is not using the hints nor skipping.

With these groups coded as a single character, we can look at the progress of one student by converting the string of words to a string of motif groups for quicker interpretation. For example, in the **too difficult (r)** grouping, these problems show the student is taking 5 to 30 seconds before taking any action. However, if the student initially chose the wrong answer, subsequent attempts were quick guesses or gaming hints. In contrast, in the **game-like (g)** grouping the student was not reading the problem, simply skipping, or making quick guesses. In the **frustration (guess) (f)** grouping the student was skipping after an attempt. This could just be an indication of skipping problems, or possibly the error in the first attempt triggering a frustration response to skip the next 9 problems. And the **on-task(n)** grouping appears to be a mixture of responses for on task tutor use.

7 Evaluating student interaction

The final step in the process is to apply the 7 meaning groups to individual students. To do this we can convert each student string into a *student motif string*. This is done by scanning the student string and replacing each problem string with a dash (-) until a motif is detected; at this point, the problem string is replaced with the meaning group code associated with the motif. The meaning group code is only placed at the final problem of each motif (the last four characters). The conversion is shown below for two students.

Using the meaning group code, we generated the student number followed by patterns for each student representing their tutor interaction. We examine two students below:

4362,"-----k-kkkkkk--k-----k-kkkkk-----kkkk-kkkkkkkk-----kkkkk-kkkk-k---k--kk-----"

4363,"-----g-----k-----g-----g-----rrrrr---rr-----g-----"

For student 4362 the *not challenged* (*k*) motif indicates solving of problems on the first attempt without using hints. (The (-) patterns are those which did not match any motif.) The behavior is a lagging indicator, representing the current and previous 9 problem.

Student 4363 started by gaming the tutor, *game-like* (*g*) either skipping or guessing quickly to find answers. This is followed by *not challenged* (*k*), a string problems solved on first attempt followed by more tutor gaming. The *too difficult* (*r*) string of r's are where the student began using the hint facility but in a manner to find answers. After about 20 of these too difficult problems he/she returned to skipping or guessing.

This conversion process illustrates how our discovered motifs can be used in a real-time application. A tutor can detect these patterns and respond based on the meaning group in order to have a more personalized interaction with the student. With student 4362 a tutor could increase problem difficulty. For student 4363, a tutor could intervene at problem 15 where gaming was detected, perhaps by introducing the hint system, or directing the student to a teaching video.

8 Discussion and future work

This paper describes a novel method for determining student behavior without linking the behavior to performance outcomes. We have shown a case study where a number of meaningful behaviors were discovered using a combination of hand chosen features and automatic pattern discovery. The features that have been found can be used to detect student behaviors so that the tutor can react in real time. However, these outcome of our study needs to be verified in future work. A number of future directions are discussed below.

We will validate that the discovered motifs accurately represent student behavior by implementing them in the tutor. Upon motif detection, the corresponding meaning group will be verified by a person in real time. The student or a teacher observer will verify the meaning group by responding to a query during the tutoring session, e.g.. "Are you skipping problems because..." These responses will be compared with the predicted behaviors for validation.

We will study automatic data categorization as modifications to our process. The data binning is the most user intensive part of this process. Finding methods to automate it would allow for broader use of these methods.

We will compare a number of different motif window sizes in order to understand the time scale of problem behavior patterns. The value used in this paper, ten problems, is sufficient to describe behavior, and it yielded a manageable number of informative motifs. However, other window levels may yield motifs of different quality and quantity.

References

- [1] Arroyo, I., Mehranian, H., Woolf, B.P.: Effort-based Tutoring: An empirical approach to Intelligent Tutoring. *EDM 2010, this volume*.
- [2] Baker, R.S., Corbett, A.T., Koedinger, K.R., Roll, I. Detecting when students game the system, across tutor subjects and classroom cohorts. In Ardissono, L., Brna, P., Mitrovic, A. (Eds.) *UM 2005. LNCS (LNAI)*, 2005. 3538, p. 220–224. Heidelberg: Springer.
- [3] Beal, C. R., Cohen, P.R. Temporal Data Mining for Educational Applications. In Ho, T.-B. & Zhou, Z.-H. (Eds.) *Trends in Artificial Intelligence: 10th Pacific Rim International Conference on Artificial Intelligence, PRICAI 2008, LNAI*, 2008. p. 66–77. Springer
- [4] Beck, J. Engagement tracing: Using response times to model student disengagement. In Looi, C., McCalla, G., Bredeweg, B., Breuker, J. (Eds.) *Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology*, 2005. p. 88–95. Amsterdam: IOS Press.
- [5] Chui B., Keogh E., Lonardi S. Probabilistic discovery of time series motifs. *Proceedings of Knowledge Discovery in Data*, 2003. p. 493–498.
- [6] Johns, J., Woolf, B.P. A Dynamic Mixture Model to Detect Student Motivation and Proficiency, *Proceedings of the National Conference on Artificial Intelligence*, 2006. p. 163.
- [7] Lin, J., Keogh, E., Lonardi, S., Patel, P. Finding motifs in time series. *Proceedings of the 2nd Workshop on Temporal Data Mining*, 2002. p. 53-68.
- [8] Stevens, R., Johnson, D., Soller, A. Probabilities and prediction: Modeling the development of scientific problem solving skills. *Life Sciences Education*, 2005, 4(1), p. 42–57.
- [9] Stevens, R., Soller, A., Cooper, M., Sprang, M. Modeling the Development of Problem-Solving Skills in Chemistry with a Web-Based Tutor, *7th International Conference on Intelligent Tutoring Systems*, 2004. p. 580-591.
- [10] Tompa, M., Buhler, J.. Finding motifs using random projections. *Proceedings of the 5th Int'l Conference on Computational Molecular Biology*. 2001. p 67-74.
- [11] VanLehn, K., Lynch, C., Schulze, K., Shapiro, J., Shelby, R., Taylor, L., Treacy, D., Weinstein, A., Wintersgill, M. The Andes Physics Tutoring System: Lessons Learned. *International Journal of Artificial Intelligence in Education*, 2005, 15(3), p. 147–204.

Unsupervised Discovery of Student Learning Tactics

Benjamin Shih, Kenneth R. Koedinger, and Richard Scheines
{shih, koedinger, scheines}@cmu.edu
Carnegie Mellon University

Abstract. Unsupervised learning algorithms can discover models of student behavior without any initial work by domain experts, but they also tend to produce complicated, uninterpretable models that may not predict student learning. We propose a simple, unsupervised clustering algorithm for hidden Markov models that can discover student learning tactics while incorporating student-level outcome data, constraining the results to interpretable models that also predict student learning. This approach is robust, domain-independent, and does not require domain experts. The models have text-set correlations with learning gain as high as 0.5 and the findings suggest possible improvements to the scaffolding used by many software tutors.

1 Introduction

Since its inception as a field, educational data mining has consisted mostly of domain experts who use machine learning rather than machine learning experts who study education. The most commonly used methods are thus highly dependent on domain expertise. Examples include domain experts constructing data features [3], generating priors [5], and developing initial seed models [4]. An expertise-based approach is highly effective for educational data, but a reliance on domain experts has risks: if the domain expert's prior beliefs are wrong then the results will tend to be biased. The process can also be time-consuming and difficult for other researchers to replicate.

Alternatively, educational data mining without domain experts often results in uninterpretable or ungeneralizable models. Our solution is a novel unsupervised algorithm that incorporates student-level educational measures directly into the learning process, biasing the model search towards models that predict learning gain. Domain experts are only involved with the post-hoc interpretation of results. Further, in addition to predicting learning gain, the algorithm's models of student behavior suggest that the students who learn best tend to make persistent attempts rather than using software help.

2 Definitions

The data for this study comes from the Geometry Cognitive Tutor. A screenshot from a version of the tutor used in this study is shown in Figure 1. We will postpone most discussion of the data for later, but the tutor's representation of geometry problems is especially important. Each problem is shown on a separate page along with a geometry diagram. Students are expected to enter values, such as angle magnitudes, into answer cells. Solving

This work was supported in part by a Graduate Training Grant awarded to Carnegie Mellon University by the Department of Education (#R305B040063)

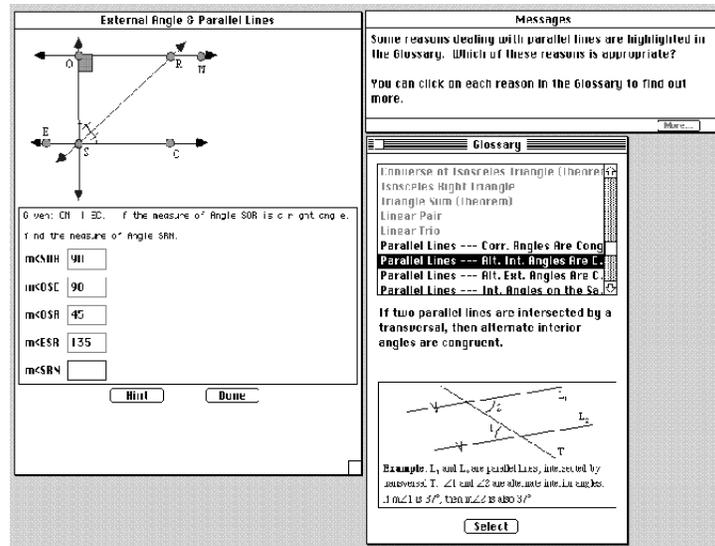


Figure 1: Geometry Cognitive Tutor, circa 1998

Table 1: Mapping from $\langle \text{Action, Duration} \rangle$ to a single variable

	Attempt/Try	Hint
Fast	a	h
Slow	A	H

for and filling in one of these cells is called a “step”. The first steps on a problem tends to involve givens; the later steps require values from the previous steps. Students can switch between steps at will, but can only switch problems by finishing them. Students can also request hints (fairly common) or read the glossary (very uncommon).

Within each step of each problem, a student performs actions (also called transactions in other literature). An action could be entering an answer (right or wrong), requesting a hint, or reading a definition from the glossary. For simplicity, we will group hints and the rare glossary request together, labeling them both as hints. The definition of an action has one additional wrinkle: each action has a corresponding duration. For example, a student might take 10 seconds to type an answer or 4 seconds to read a hint. Thus, actions are divided into two categories, long and short, using a threshold to be discussed later. Table 1 shows this mapping. For example, Aaaaaa denotes one long attempt followed by many short attempts and might be considered a guessing tactic.

In sum, the data for this study consists of students working on geometry problems with each problem split into steps. The problems and steps are broken down into actions (a, A, h, or H). These rudimentary features are much simpler than human-constructed features. Beal et. al., for example, work from researcher-constructed features like “independent-inaccurate problem solving” [3]. Their approach, while successful, is dependent on the quality of the constructed features. Similarly, Baker et. al. use a set of human-constructed features as inputs to a feature construction algorithm [2]. Their approach can generate new

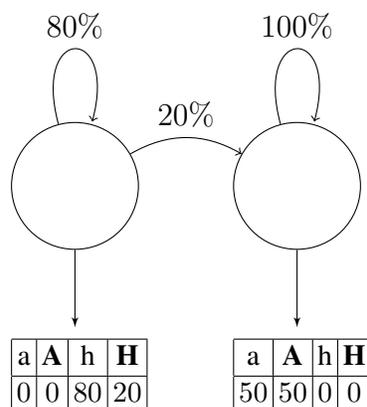


Figure 2: Example HMM

composite features, but the initial features must be expert-defined. Another distinction is that the actions used in this study are atomic: there is no lower granularity of data available. In contrast, Baker computes aggregate features over a roaming window of actions.

3 Hidden Markov Models

The goal of this study is to build models for student learning tactics. An example of a learning tactic, as defined in this paper, is, “The student requests hints quickly, over and over, until the tutor provides the solution. The student then enters the solution.” From this example, a learning tactic can be generalized to be an observable, predictable, and repeated *pattern* of behavior that is sufficiently abstract to include multiple *observed* instantiations. We implement learning tactics using hidden Markov models. A hidden Markov model (HMM) is a set of unobserved states, each state related to observations through a probability distribution. Here, the observations are student actions. Figure 2 shows an example HMM. Each unobserved state is represented by a circle; each arrow between states or looping back to a state represents a transition; the number above the arrow is a transition probability. The tables below the states show the probabilities of observing an action. When an HMM generates an action symbol, we say it *emits* the symbol.

Let a series of observed actions be a sequence. Sequences can be defined for either all actions in a problem or all actions in a step. Given a set of student sequences associated with an HMM, the Baum-Welch algorithm can relearn the parameters of that HMM to better fit the observed data. This is a standard method for learning a single HMM.

Single HMMs, as described above, have been used in many studies to model student behavioral traces. In a particularly relevant study, Beal et. al. used tutoring system log data to learn HMMs modeling patterns of student behavior [3]. Their study differs from this one in several ways: they define the structure of the HMMs by hand, they use outputs from another algorithm as inputs to their HMMs, they learn one HMM per student, and they perform clustering of students (not tactics) only after learning the HMMs. However, their key result is very relevant: HMMs work as both descriptive and predictive models for student

learning behaviors and can find patterns without using cognitive models or domain content knowledge.

3.1 HMM Clustering

Let each individual HMM represent a single learning tactic. Discovering learning tactics requires discovering sets of HMMs. Let a set of HMMs be called a collection. In a collection, an observed sequence of actions is classified by whichever HMM is most likely to generate it. This results in a partitioning of the set of sequences, with each partition corresponding to one HMM. Each partition thus includes all observed examples of a given tactic.

The Baum-Welch algorithm can only learn parameters for a single HMM, but clustering algorithms can learn sets of HMMs, and thus sets of tactics. The usual objective of an HMM clustering algorithm is to maximize the total likelihood of generating the observed sequences. This type of problem has historically been tackled with Expectation-Maximization (E-M) algorithms and, for HMM clustering, given an initial set of HMMs, one iteration of the E-M algorithm is:

- Assign each sequence to the HMM most likely to generate it.
- For each HMM, relearn its parameters with Baum-Welch using the sequences in its partition.

This process begins with initial seed HMMs and repeats until a termination criterion is met, such as when an iteration results in fewer than 10 sequences being reclassified. A collection learned by this algorithm fits the data well if the likelihood of generating the observed sequences is high. This algorithm, here forth called *HMM-Cluster*, is provably guaranteed to converge to a local maximum. Further, *HMM-Cluster* will never change the number of HMMs in the collection (k) or the number of states per HMM (n); only the parameters and partitions will change.

There have been many prior uses of similar E-M HMM clustering algorithms, beginning with Rabiner et. al. for word recognition [6]. While there are newer variants, most HMM clustering is still done with Rabiner's original algorithm. A particularly illustrative study was done by Schliep et. al. to analyze gene expression data[9]. The paper discusses, amongst other things, the expressiveness of the models, the interpretation of results (for genetics), the inclusion of human labels, and the comparison of HMM clusters to other time series models.

3.2 Stepwise-HMM-Cluster

Unfortunately, naive *HMM-Cluster* has issues from both machine learning and educational perspectives:

- Like most E-M algorithms, *HMM-Cluster* gets trapped in local maxima.
- The choice of k and n determines the effectiveness of *HMM-Cluster*. If they are too

X:			Y:
	HMM 1	HMM 2	Gain
Student 1	80%	20%	73%
Student 2	30%	70%	9%
Student 3	25%	75%	5%

Figure 3: Example Stepwise Regression Inputs

large, the collection will overfit; if they are too small, no collection will fit the data.

- Collections that fit the data may not actually predict learning.

In principle, a better algorithm would search over values of k and n with a bias towards fewer, smaller HMMs, leading to better generalization and easier interpretation of the final collection. One such algorithm is *Stepwise-HMM-Cluster*, which is to *HMM-Cluster* what stepwise regression is to normal regression. An iteration of *Stepwise-HMM-Cluster*, for k HMMs and n states per HMM, proceeds:

- Begin with a collection of HMMs C .
- If $|C| < k$, generate $(k - |C|)$ new HMMs with n states per HMM.
- Run *HMM-Cluster* on C .
- Pick the “good” HMMs from C and use them for the next iteration.

A critical step in *Stepwise-HMM-Cluster* is the selection of “good models” from a collection C . This step allows *Stepwise-HMM-Cluster* to incorporate external data and iteratively improve its fit across iterations of the algorithm. For this study, HMMs are selected using forward stepwise linear regression: the total number of sequences classified by each HMM for each student is used as the independent variable and the pre-test to post-test learning gain is used as the dependent variable. A toy example is shown in Figure 3.

Stepwise-HMM-Cluster serves two goals at once: it tries to build a collection of HMMs to fit the observed sequences of actions, but also requires that the collection predict student learning gain. The incorporation of external data, such as pre-post learning gain, has been traditionally difficult when applying machine learning algorithms to educational data. This selection step addresses that issue, allowing student-level measures to influence the learning of much lower-level HMMs. In this case, learning gain is used to constrain the search for problem- and step-level HMMs. In future work, other data sources could be added, such as grade-point averages, survey information, or expert labels.

For this study, the parameters are restricted to $2 \leq k \leq 8$ and $2 \leq n \leq 8$. The limits of 8 HMMs and 8 states per HMM were chosen to maximize interpretability, but both limits exceed the complexity of any optimal collections actually found.

4 Data

This study uses two data sets, 02 and 06. Both data sets in this study originate in previous experiments, so only the control groups for each study are used. Both data sets involve

Table 2: Correlations with Learning, Best Collections, Test Data

τ	02 Problem	02 Step	06 Problem	06 Step
6	0.57	0.50	0.3	0.53
8	0.71	0.54	0.50	0.39
10	0.54	0.60	0.31	0.39

geometry tutoring systems that use the same general interface. However, the 02 data is from the angles unit, while the 06 data is from the circles unit. The 06 tutor also has some interface differences, including a minimum time per hint request. In the 06 data, students do fewer actions per step, complicating direct comparisons between the two data sets. Also, the 06 post-test used counter-balanced hint conditions between problems, e.g., sometimes students could get a hint at the cost of partial credit. This makes the test scores noisier and harder to predict.

- 02 data - First published in 2002, includes 21 students and 57204 actions [1].
- 06 data - First published in 2006, includes 16 students and 7429 actions [7].

5 Results

Stepwise-HMM-Cluster has several parameters. First, there is a threshold value between long and short actions. Let that threshold be denoted by τ . Second, *Stepwise-HMM-Cluster* can learn either problem-level or step-level tactics. For the former, HMMs are trained on sequences that include an entire problem. In the latter case, each sequence only contains actions from one step. The software implementation was built on the GHMM package [8] and, for a given search using a fixed value of τ , approximately 100 candidate collections reach the model selection stage.

Collections are learned from the first 80% of sequences per student; the remaining 20% are saved as test data. The main measure of a “good” collection is that it provides an accurate prediction of learning when applied to test data. To apply a collection of HMMs to test data, the HMMs are first used to classify test-data sequences. The total number of sequences per HMM per student is entered into the regression as shown earlier, now using parameters learned from training data. Table 2 shows the best correlations, for both data sets, between predicted learning gain and actual learning gain, as computed on the test data. Each column contains the best results for a specific run of *Stepwise-HMM-Cluster* with the rows split by value of τ .

In practice, Table 2 can be interpreted as showing upper bounds for predictions on withheld test data. However, even in this simple table, it’s already clear that the 06 data is harder. The conclusion from Table 2 is that there are collections with a good fit to test data, if we can find them. The caveat to Table 2 is that it shows best collections picked after applying to test data; our actual goal is to find good collections using only the training data. To do so naively, however, invites overfit. This suggests the use of a selection heuristic: pick the collection with the best adjusted R^2 score on training data.

Table 3: Correlations with Learning, Selected Collections, 02 Test Data

τ	Problem-Level	Step-Level	# of HMMs	Max # of States / HMM
6	0.49	0.44	5	3
8	0.42	0.50	5	4
10	0.47	0.52	4	4

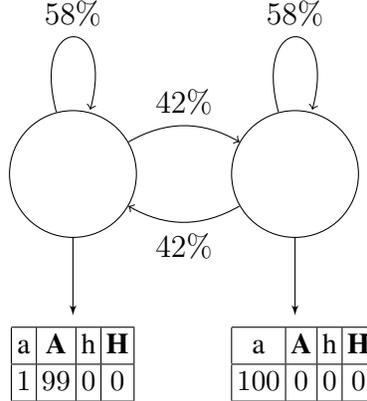


Figure 4: Dominant HMM for $\tau = 6$, 02 data

R^2 , unadjusted, is defined as the sum-of-squared-error divided by the total sum of squares. For standard linear regression, R^2 is equal to the square of the correlation coefficient. The adjusted R^2 includes an additional term that grows in the number of model parameters, penalizing complex collections. Table 3 shows, for 02 data, test-set correlations for collections selected using adjusted R^2 , the number of HMMs in the best collection (step-only), and the maximum number of states per HMM in the best collection (step-only).

The correlations in Table 3 are statistically significant ($\alpha < 0.05$) and almost as high as those in Table 2. They clearly show that, for the 02 data, it's possible to pick collections that generalize to with-held, within-student test data. However, the same table for the 06 data (not shown) is much less convincing. In 06 data, naively picking collections using the adjusted R^2 produces collections with poor predictions of learning. However, in 06, amongst the collections with the highest adjusted R^2 , some collections do have a high test-set correlation with learning gain. For example, for $\tau = 6$, the fourth-best collection has a 0.46 test-set correlation with learning gain. The problem is selecting the right collection: while the adjusted R^2 is effective for 02 collections, it selects poorly from 06 candidates.

In the end, educational data mining requires interpretable results that have educational implications. Fortunately, *Stepwise-HMM-Cluster* outputs simple, interpretable HMMs. In particular, there is one HMM that occurs, with slightly different parameters, in every 02 collection shown in Table 3. This HMM-archetype always classifies a plurality of sequences, and so will be called the “dominant” HMM. Figure 4 shows a dominant HMM for $\tau = 6$, trained on 02 data at the step level. These dominant HMMs tend to be small, often only two states and never more than four.

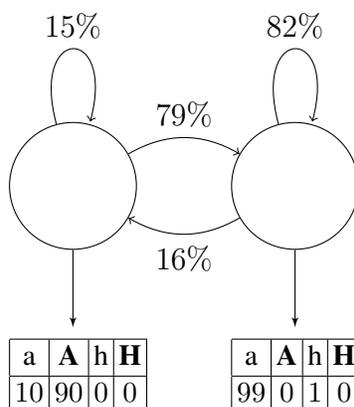


Figure 5: Repeated Guessing HMM for $\tau = 6$, 02 data

While interpreting the structure of individual HMMs is not actually meaningful (to be addressed), it is still a useful comprehension exercise. Here, the dominant HMM emits a and A with high probability, and emit both actions equally often (over the course of many sequences). One possible explanation is that the dominant HMMs select short sequences where the student already knows the answers and can solve each step in one attempt. However, the correlation between the frequency of first-try-correct sequences and learning gain is -0.24 . Instead, an alternative interpretation for these HMMs is that they represent a persistence-trait. Students that attempt to solve repeatedly are more likely to learn the material than those that rely on hints. This is borne out by the resilience of the HMM to changes in τ , and by the duration-agnostic nature of the HMM, which emits both a and A.

However, this conflicts with common sense. The HMM shown in Figure 4 has a high probability¹ of emitting a sequence of type Aaaaaa, i.e., a single long attempt followed by many short ones. This is generally considered poor learning behavior [2]. Intuitively, it represents a failed attempt followed by repeated, unthinking guessing. This paradox can be resolved by noting that no single HMM in any collection can be interpreted alone. Each HMM exists only as part of an entire collection and, thus, other HMMs in the collection can remove specific, degenerate sequences. Take the $\tau = 6$ collection as an example. It contains an HMM, shown in Figure 5, that has a high probability of emitting repeated-guessing type sequences. The repeated-guessing HMM, a highly specialized model, removes only the guessing sequences from the dominant HMM's partition. This relationship between HMMs in a collection, where a specific HMM can be tuned to special cases of a more general HMM, allows collections to be more expressive than the sum of their individual HMMs. However, this feature is also what makes the interpretation of the structure of individual HMMs meaningless, as a high probability sequence for one HMM may actually belong to another HMM's partition.

A more appropriate way of interpreting the HMM clusters is to directly examine the se-

¹A high probability as compared to any other sequence of the same length. These HMMs are not assumed to model the distribution of sequence lengths, so comparing sequence probabilities is only meaningful for a fixed length.

quences classified by a particular HMM. For example, consider the dominant HMM for $\tau = 6$, 02 data, step-level. The five most commonly observed sequences in the HMM's partition are: **A**, **AA**, **Aa**, **AAA**, **AAa**. None of these sequences are of the repeated-guessing type, yet they account for 95.5% of all sequences in the partition. Longer sequences in the partition follow the same pattern: example sequences include **AAaaAA** and **AAaAaA**. As noted above, guessing sequences, e.g., **Aaaaaa**, are about as likely to be generated by the dominant HMM as the above sequences, but are actually captured by the repeated-guessing HMM. Similar results apply to the collections discovered for other values of τ .

The general interpretation of these results is that students learn more when using persistence-type tactics, as long as they don't guess repeatedly. Interestingly, this is largely independent of the choice of threshold τ . The most likely explanation is that very short or very long actions contains the most information about the student, and thus the actions that are reclassified by small changes in τ are relatively unimportant.

Finally, across all the best collections, hint-heavy tactics are negatively associated with learning. However, many of the more complex collections (3 or 4 HMMs) contain a "noise" HMM that generates all sequences with nearly uniform probability. Thus, hint-specific HMMs are actually very specialized, usually emitting mostly h actions. This explains the negative association with learning. Some "good" HMMs do involve hints, but those HMMs are not structurally consistent enough to permit conclusions without more data or analysis.

6 Conclusions

Most educational data mining methodologies either rely on domain experts or discover uninterpretable models. In contrast, *Stepwise-HMM-Cluster*, an unsupervised algorithm, can generate collections of HMMs that predict learning, but are also interpretable. For at least some data sets, *Stepwise-HMM-Cluster* produces collections of HMMs that can provide good predictions on with-held test data. This algorithm thus satisfies multiple educational data mining goals: it produces interpretable models, the models generalize (within-student), and the models not only fit data, but also predict learning outcomes.

Additionally, *Stepwise-HMM-Cluster* produced models with potential educational implications. Our results provide an additional argument that the most common type of hint-scaffolding in software tutors may be sub-optimal and that most learning may arise from persistent attempts to solve. This suggests a paradigm for tutoring systems that emphasizes attempts and provides hints or worked examples only when strictly necessary; however, there are other feasible explanations and more extensive exploration of this issue is required. In particular, there may exist learning tactics that are both productive and involve hints, but are difficult to detect due to noise or rarity.

There are many opportunities for future work. First, the evidence for statistical generalization is still weak; collections learned on one data set should be tested on another data set entirely. Second, adjusted R^2 appears to be a poor model selection criterion in some cases and other criteria may be more successful. Third, τ is a problematic parameter: while *Stepwise-HMM-Cluster* was robust to changes in τ in this study, it may not be robust in general. For

example, some data sets may require three action strata (“Long”, “Medium”, “Short”) or require a different τ for hints versus attempts. Fourth, while there is already a procedure for interpreting clusters, there is significant room for improvement. A promising approach is to construct a visualization of the unfolding of sequences with sets of sequences with the same prefix or suffix grouped together. Finally, the algorithm itself is overly simple. The heart of *Stepwise-HMM-Cluster* is a strict assignment, E-M clustering algorithm using Baum-Welch; probabilistic mixture models or another method, such as spectral clustering, might improve the results, as might a better HMM learning algorithm.

Despite its limitations, *Stepwise-HMM-Cluster* has many potential applications. First, in its present form, the algorithm can already learn interesting models with little dependence on data. However, it is also flexible and extendible. Glossary requests could be separated from hints; new action types could be added for new data sets; tactics could be learned on the level of class sessions or curriculum units instead of problems and steps; human labels could be incorporated into the model selection step; actions could be redefined to include domain information, such as skill models. Further, there is the potential to develop a fully hierarchical algorithm that could simultaneously learn HMMs for individual step tactics, learn HMMs to classify problem tactics as a series of previously learned step tactics, and so on up, as far as sample size will permit.

However, the most important contribution of this study is neither the algorithm nor the educational implications. Rather, our results suggest an opportunity for a new paradigm where algorithms can simultaneously leverage multiple data sources at different granularities. In particular, constraining low-level models using student-level measures could potentially improve many existing algorithms and lead to important educational insights.

References

- [1] ALEVEN, V., AND KOEDINGER, K. R. An effective meta-cognitive strategy: Learning by doing and explaining with a computer-based cognitive tutor. *Cognitive Science* (2002), 147179.
- [2] BAKER, R. S., CORBETT, A. T., AND KOEDINGER, K. R. Detecting student misuse of intelligent tutoring systems. In *Proceedings of the 7th International Conference on International Tutoring Systems* (2004).
- [3] BEAL, C. R., MITRA, S., AND COHEN, P. R. Modeling learning patterns of students with a tutoring system using hidden markov models. In *Proceedings of the 13th International Conference on Artificial Intelligence in Education* (2007).
- [4] CEN, H., KOEDINGER, K. R., AND JUNKER, B. Learning factors analysis - a general method for cognitive model evaluation and improvement. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems* (2006).
- [5] CONATI, C., GERTNER, A. S., VANLEHN, K., AND DRUZDZEL, M. J. On-line student modeling for coached problem solving using bayesian networks. In *Proceedings of the Sixth International Conference on User Modeling* (1997).
- [6] RABINER, L. R., LEE, C. H., JUANG, B. H., AND WILPON, J. G. Hmm clustering for connected word recognition. In *Proceedings of the IEEE ICASSP* (1989).
- [7] ROLL, I., ALEVEN, V., MCLAREN, B. M., RYU, E., BAKER, R. S., AND KOEDINGER, K. R. The help tutor: Does metacognitive feedback improve students' help-seeking actions, skills and learning? In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems* (2006).
- [8] SCHLIEP, A., GEORGI, B., RUNGSARITYOTIN, W., COSTA, I. G., AND SCHNHUTH, A. The general hidden markov model library: Analyzing systems with unobservable states. In *Proceedings of the Heinz-Billing-Price* (2004).
- [9] SCHLIEP, A., SCHNHUTH, A., AND STEINHOFF, C. Using hidden markov models to analyze gene expression time course data. *Bioinformatics* 19 (2003), 255–263.

Assessing Reviewers' Performance Based on Mining Problem Localization in Peer-Review Data

Wenting Xiong¹ and Diane Litman^{1,2} and Christian Schunn²
{wex12, dlitman, schunn}@pitt.edu

¹Department of Computer Science, University of Pittsburgh

²Learning Research and Development Center, University of Pittsburgh

Abstract. Current peer-review software lacks intelligence for responding to students' reviewing performance. As an example of an additional intelligent assessment component to such software, we propose an evaluation system that generates assessment on reviewers' reviewing skills regarding the issue of problem localization. We take a data mining approach, using standard supervised machine learning to build classifiers based on attributes extracted from peer-review data via Natural Language Processing techniques. Our work successfully shows it is feasible to provide intelligent support for peer-review systems to assess students' reviewing performance fully automatically.

1 Introduction

Peers reviewing each other's writing is commonly used in various academic fields. A typical peer-review practice consists of three sections: first, students write essays on certain prompts; second, they provide feedback on essays of their peers; third, based on the feedback they get from their peers, they revise their initial essays. Peer-review is valuable not only because it provides learning opportunities for students, but also because it is more abundant in quantity compared with feedback from instructors. Besides, peer-review exercises also provide students the opportunity to develop their reviewing skills.

One problem with peer-review feedback is that its quality is not always guaranteed [1]. Previous studies on the nature of feedback suggest that the quality of feedback, in terms of the likelihood of its implementation, is significantly correlated with certain feedback features [2], among which problem localization is most significant. As defined in previous work, problem localization refers to pinpointing the source of the problem and/or solution. While such feedback features were used as mediators in the analysis of feedback helpfulness in [2], we believe that they could also be used as indicators in evaluating feedback quality automatically.

To date, current peer-review software facilitates the peer-review exercise with respect to document management and review assignment. However, no automatic feedback is generated for students regarding their reviewing performance. To add an automatic assessment component to peer-review software, we construct an evaluation system for reviewing performance that provides binary assessment for reviewers with respect to problem localization. Taking a data mining approach, we use standard supervised machine learning algorithms to build classifiers for identifying feedback features, based on attributes extracted from peer-review data with Natural Language Processing (NLP) techniques. Our results suggest that it is feasible to add an assessment component to peer-review software that could respond to students' reviewing performance automatically.

2 Related Work

Empirical studies of peer-review feedback based on manual coding have explored which feedback features predict whether feedback will be understood; the understanding of feedback was found to be a good predictor of whether feedback was implemented. For example, one study [2] has analyzed the rate of understanding the problem as a function of the presence/absence of feedback features, and found that feedback was more likely to be understood when the location of the problem was explicitly stated, or the solution to the specified problem was provided. This suggests those feedback features contribute to feedback implementation, which further indicates the helpfulness of feedback.

There is an increasing interest in research on computer-supported peer reviews that can bring benefits to both instructors and students. In our work, we aim to enhance the quality of feedback received by students by automatically assessing and guiding students' reviewing performance. Similarly, researchers from the data mining community have tried to predict feedback helpfulness automatically based on previous theoretical discoveries. With the help of software such as SWORD¹, peer-review corpora are being collected and can be used for data mining and machine learning. One study [3] on a corpus that SWORD collected used machine learning and classified any piece of peer-review feedback as helpful or not helpful based on tags that are automatically generated by tagging software. (In contrast, [2] took a manual-analysis approach: they require human annotators to code many feedback features that could be potentially relevant with respect to their purpose of study.) The result in [3] showed the performance of the classifier was limited by errors from the tagging software, which couldn't distinguish problem detection and solution suggestions (they are both types of criticism feedback).

In this paper we will also examine a corpus collected with SWORD. However, in contrast to [3], we first detect the criticism feedback, and then predict the helpfulness of the recognized criticism feedback only based on the issue of problem localization. By treating criticism feedback as one group, we get around the problematic identification between solution suggestion and problem detection. In contrast to both [3] and our own prior work in [8], our system also aims to assess reviewing performance at the reviewer-level, rather than predicting the helpfulness [3] or problem localization [8] of a given piece of feedback. In the area of NLP, one related work of identifying criticism feedback could be sentiment analysis [4], while problem localization often involves paraphrasing [5] partial content of the associated essay. However, in this preliminary study, we take a simple approach in addressing these problems.

3 Data

The data used for this work is from a previous study [2] of the relationship between feedback features and helpfulness. The data was collected using SWORD in a college

¹ Scaffolded Writing and Rewriting in the Discipline. <http://www.lrdc.pitt.edu/Schunn/sword/index.html>

level history introductory class. It consists of textual reviews provided by 76 reviewers referring to 24 associated student history essays.

In the previous study, all the textual reviews were manually segmented into 1405 idea-units (defined as contiguous feedback referring to a single topic).² These units were further annotated by two independent annotators for various coding categories. For the purpose of our work, we automatically predict two of the coding categories, *feedbackType* and *pLocalization*.

FeedbackType was coded with three values — criticism, praise and summary. For only criticism feedback, *pLocalization* was then coded as true or false, indicating whether the criticism feedback contains problem localization for any specified problems or suggested solutions. According to the coding scheme, *pLocalization* is not applicable to praise or summary feedback, thus its *pLocalization* was labeled as N/A. The reported agreement (Kappa) between two annotators on *FeedbackType* is 0.92, and that on *pLocalization* is 0.69. Relevant statistics are listed in Table 1. From now on, **feedback** will be used to refer to the 1405 annotated feedback idea-units.

Table 1. Descriptive statistics of annotations on history peer-review feedback data

Coding category	Value			
	feedbackType	criticism	praise	summary
875		388	142	1405
pLocalization	true	false	N/A	
	462	413	530	1405

In addition to the feedback idea-units, we also have access to the collection of 24 essays to which the feedback refers. These essays provide domain knowledge, and are a self-contained resource that will assist us in mining features from the peer-review feedback data using statistical NLP techniques.

4 System and Features for Classification

Before diving into details of feature extraction and model learning, we would like to first provide an overview of our system, which takes the annotated feedback provided by a single reviewer, identifies target features sequentially for each piece of feedback, and generates assessment on the reviewer’s reviewing performance with respect to problem localization in general (as the flow suggests in Figure 1).

² In the new version of SWORD, segmentation is handled automatically through the interface which requires users to submit comments separately by idea-unit.

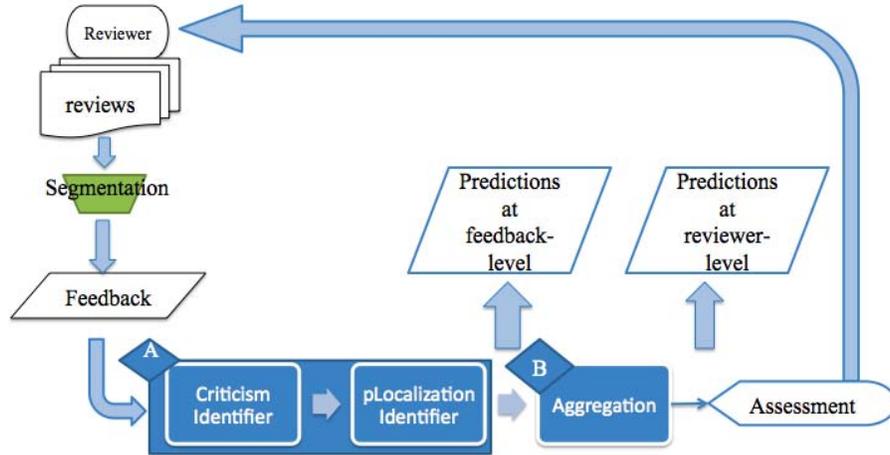


Figure 1. System overview

4.1 System overview

The system described in Figure 1 can be viewed as an assessment module compatible with peer-review software such as SWoRD. The system consists of two binary classifier components for identifying problem localization at the feedback-level (part A in Figure 1), plus an aggregation component (part B in Figure 1), which propagates the prediction of problem localization from the feedback-level up to the reviewer-level.

In pilot work, adding either annotated or predicted *feedbackType* into the feature set significantly improved the model’s performance on identifying problem localization at the feedback-level. Therefore, we decompose the task into two concatenated tasks. We first use supervised learning to train a classifier for identifying criticism feedback versus praise and summary feedback; then we use the same algorithm to train another classifier for identifying whether problems are localized (*pLocalization* = true) for a given criticism feedback. Note that although both *feedbackType* and *pLocalization* were annotated with three values (Table 1), we combined values to create two binary classification tasks. Because it is the criticism feedback that is actionable, and focused on in the next step (classification for *pLocalization*), we group the praise and summary feedback together as non-criticism. As a byproduct, this binary separation also results in a more balanced data set from the perspective of machine learning. Similarly, recall that all non-criticism feedback in the data set was labeled N/A for *pLocalization*; we group N/A with false (vs. true), which simplifies our model for handling noisy output of the *feedbackType* identifier (specifically, any non-criticism predicted as criticism and sent to the second component).

Since our goal is to generate assessment of reviewing performance for each reviewer, we add another aggregation step into the system after the two components mentioned above, in which we make a binary decision on whether the reviewer provides enough problem localization in their reviews in general. This decision is based on the predictions made by the two preceding components on problem localization for all the feedback submitted by that reviewer. Since localization at the feedback-level is relatively difficult even for

humans (recall Kappa=0.69), we expect to provide more accurate (and hence useful) feedback at the aggregate level.

4.2 Criticism Identifier

To identify criticism feedback, we develop 3 groups of attributes that are automatically derived from the surface of sentences.

Simple: This set simply contains two attributes, the *wordCount* and the *feedbackOrder* in the review. *WordCount* is the number of words in the feedback; *feedbackOrder* is the index of the feedback with respect to its original review before segmentation. Based on our brief exploration of the data, we hypothesize that negative feedback is more likely to be verbose than positive feedback, and there is a certain pattern in expressing opinions, thus the *feedbackOrder* is useful in detecting criticism feedback.

Essay: There are four attributes in this group capturing the topic information contained in the feedback. To build a domain dictionary, first we preprocess the collection of 24 essays into bigrams (two adjacent words) and unigrams (single word). In particular, using NLTK³ we extract bigrams whose term frequency–inverse document frequency (TF-IDF) is above the average TF-IDF of all bigram-collocations to form the bigram-domain dictionary. Then we gather all unigrams that constitute the bigrams in the bigram-domain dictionary for building our unigram-domain dictionary. Our final dictionary contains 291 bigrams and 402 unigrams. To capture how much content of the feedback is related to the domain, we count bigrams in the feedback that also belong to the bigram domain dictionary, and create the attribute *Collocation_d*. Similarly we create *Collouni_d* based on unigrams. Besides the domain-topics shared by all essays in general, we also considered essay-topics, referring to terms that are more frequently used in one specific essay rather than all of them. For each piece of feedback, we compute its bigrams and unigrams, and then only count (*Collocation_e* and *Collouni_e*) those that also appear in the associated essay with above-average item frequency of that essay. These four counts are normalized with the length of feedback.

Keyword: Due to the expensive computational cost for building models based on all words in the feedback corpus, we semi-automatically learned a set of Keywords (Table 2) which has categories based on the semantic and syntactic function of the words.⁴ We first manually created a list of words that are specified as signal words for annotating *feedbackType* and *pLocalization* in the coding manual; then we supplemented the list

³ Natural Language Toolkit. <http://www.nltk.org/>

⁴ We also considered Bag-of-Words as well as sentiments that could be easily extracted with available software for opinion analysis features, but Keyword turned out to perform as well as the combination of these two groups, if not better. We prefer using Keyword in our model because it involves considerably fewer attributes thus reducing the complexity of our model, and it does not require the need for sentiment analysis software.

with the words selected by a decision tree model learned using a feature vector consisting of all words in the feedback (Bag-of-Words). As shown in Table 2, we generated nine attributes counting the number of words in the feedback that belongs to each tag category, respectively.

Table 2. Keyword table

Tag	Meaning	Word list
SUG	suggestion	should, must, might, could, need, needs, maybe, try, revision, want
LOC	location	page, paragraph, sentence
ERR	problem	error, mistakes, typo, problem, difficulties, conclusion
IDE	idea verb	consider, mention
LNK	transition	however, but
NEG	negative	fail, hard, difficult, bad, short, little, bit, poor, few, unclear, only, more, stronger, careful, sure, full
POS	positive	great, good, well, clearly, easily, effective, effectively, helpful, very
SUM	summarization	main, overall, also, how, job
NOT	negation	not, doesn't, don't

4.3 PLocalization Identifier

For a given piece of criticism feedback, we developed four groups of attributes to capture different perspectives of localized expressions.

Regular Expression: Three simple regular expressions were employed to recognize common phrases of location (e.g., ~~on~~ page 5”, ~~the~~ section about”). If any regular expression is matched, the binary attribute *regTag* is true.

Domain Lexicon: Intuitively, localized feedback tends to use more domain vocabulary. Using the domain dictionary that we generated in 4.2 to calculate ESSAY attributes, we counted unigram domain-topics (*collouni_d*) contained in each piece of feedback.

Syntactic Features: Besides computing lexicon frequencies from the surface text, we also extracted information from the syntactic structure of the feedback sentences. We used MSTParser [6] to parse feedback sentences and hence generated the dependency structure of feedback sentences. Then we investigated whether there are any domain-topics between the subject and the object (*SO_domain*) in any sentence. We also counted demonstrative determiners (this, that, these and those) in the feedback (*DET_CNT*).

Overlapping-window Features: The three types of attributes above are based on our intuition about localized expressions, while the following attributes are derived from an overlapping-window algorithm that was shown to be effective in a similar task -- identifying quotations from reference works in primary materials for digital libraries [7]. To match a possible citation in a reference work, it searches for the most likely referred

window of words through all possible primary materials. We applied this algorithm for our purpose, and considered the length of the window (*windowSize*) plus the number of overlapped words in the window (*overlapNum*).

4.4 Example of Attribute Extraction

To illustrate how these attributes were extracted from the feedback, consider the following feedback as an example, which was coded as *feedbackType=criticism*, *!Localization=true*:

The section of the essay on African Americans needs more careful attention to the timing and reasons for the federal governments decision to stop protecting African American civil and political rights.

This feedback has 31 words (*wordCount* =31) and its index in the review is 2 (*feedbackOrder*=2). It has 2 bigram domain-topics (*!African American* 2), 9 unigram domain-topics (*!African* 2, *!American*, *!Americans*, “federal”, *!governments*, *!evil*, “political” and *!rights*), and 2 bigram essay-topics (*!African American* 2) plus 8 unigram essay-topics (same as the unigram domain-topics except the *!rights*). These four numbers are then normalized by the count of words in this feedback. As for Keyword, it contains 1 SUG (*!need*) and 2 NEG (*!more*, *!careful*).

The *regTag* is true because one regular expression is matched with *!the section of*; there is no demonstrative determiner, thus *DET_CNT* is zero; *!African Americans* is between the subject *!section* and the object *!attention*, so *SO_domain* is true.

4.5 Aggregation

To finally generate an overall assessment of appropriate problem localization by each reviewer, the system aggregates the relevant predictions at the feedback level, calculating a *pLocalization%* score representing the reviewer’s overall performance, which is the percentage of criticism feedback whose *pLocalization* is *!true* submitted by that reviewer. To classify reviewers into *!high* and *!low* groups regarding overall reviewing performance, we compare their *pLocalization%* score against a threshold and make a binary decision. In this work, we used an intuitive threshold that is the *pLocalization%* of criticism feedback of all reviewers (the number of *!true* *pLocalization* criticism over the number of criticism feedback in the training data). This threshold performed best among several alternatives explored in a pilot study.

5 Experimental Setup

Though the system output is the assessment of reviewer’s reviewing performance, its error could be due to any of the predictions made in the three components. To better analyze the predictive power of our system, we first evaluate each component separately (Section 5.1), then combine them and test its performance in a fully automatic version (Section 5.2). We compare our result to a Majority Class baseline for all experiments.

5.1 Component Evaluation

The *feedbackType* experiment uses all 1405 feedback from 76 reviewers. We use the Decision Tree algorithm provided by WEKA⁵ and conduct 10-fold cross validation for evaluation. We chose the Decision Tree learning algorithm not only because it worked best in a preliminary study (compared with Naïve Bayes, Logistic Regression, and SVM), but also because the learned model (decision tree) is easier to interpret and provides clearer insights into how problem localization is recognized. Results are presented in Table 3 and explained in Section 6.

Recall that *pLocalization* was only coded for criticism feedback (non-criticism feedback are directly coded as N/A), thus for the isolated evaluation of this component, we only use the 875 criticism feedbacks for training and testing. The learning algorithm and evaluation settings are the same as those used for *feedbackType* (Table 3, Section 6).

5.2 System Evaluation

Though there is no annotation of overall problem localization for each reviewer (*pLocalization* at reviewer-level) in the data set, we can generate this by aggregating the *pLocalization* (annotated values) at the feedback-level, as we described in section 4.5. Note that when generating binary labels (High and Low) for each reviewer, the aggregation is based on annotated *pLocalization* and the threshold is calculated with annotated labels. When predicting, the aggregation is based on all predicted values. Thus the threshold would be different correspondingly.

When all components work together as a system, the *pLocalization* identifier receives the output of the *feedbackType* identifier. Therefore in the combined version, the *pLocalization* identifier was trained with the 1405 feedbacks, with one new attribute: predicted *feedbackType*. We again use the Decision Tree algorithm provided by WEKA for learning, while in this case we conduct leave-one-reviewer-out cross validation for evaluation. Results are presented in Table 4.

6 Results

Table 3 presents the experimental results of the performance of each component in isolation. With respect to the accuracy of our models, both significantly ($p < 0.05$) outperform our baselines (79% vs. 62% and 78% vs. 53%) and their Kappa values are all greater than 0.5. Because we would like to provide further tutoring for the “Low” group of reviewers in the future, we are more interested in precision and recall of predicting the “Low” group. Thus we also analyze precision and recall for *feedbackType=criticism* and *pLocalization=true*, which are used to compute the *pLocalization%* scores. As listed in Table 3, both models achieve precision higher than 0.8, while for identifying criticism feedback the model’s recall is even 0.86, though the *pLocalization* model has 0.73 for

⁵ <http://www.cs.waikato.ac.nz/ml/weka/>

recall, which is still acceptable. Since the Majority class always predicts *feedbackType* as *criticism* and *pLocalization* as *true*, its recall will always be 1, thus we don't aim to beat the baseline for recall. Besides examining the quantitative results, we can also examine our results for qualitative characteristics. The learned model (decision tree) for *pLocalization* at the feedback-level is compact, using only 5 attributes (presented in [8]), and it suggests that domain-word counts plays an important role. The *feedbackType* model though more complicated, also relies on domain knowledge (*Collocation_d* and *Collouni_d* appear close to the root).

When all the components work together, the overall system can successfully predict the “Low” group of reviewers with both precision and recall higher than 0.8 (Table 4). Table 4 also presents the confusion matrix for details. Although system performance suffers from errors within each component, aggregation does alleviate it and maintains the overall performance comparable to the *pLocalization* model in isolation (M2 in Table 3).

Table 3. Performance of identification of feedbackType and pLocalization at feedback-level

	Model	Accuracy	Precision	Recall	Kappa
feedbackType (n = 1405)	Baseline	62%	0.62	1	0
	M1	79%	0.81	0.86	0.54
pLocalization (n=875)	Baseline	53%	0.53	1	0
	M2	78 %	0.82	0.73	0.55

Table 4. Performance of the overall system for identifying pLocalization at reviewer-level

Confusion matrix		
Predict\Label	High	Low
High	21	9
Low	8	38
Precision (Low)		0.81
Recall (Low)		0.83

7 Conclusion and Future Work

In this paper, we proposed a novel system for generating automatic assessments of reviewing performance with respect to problem localization at the reviewer-level. As a preliminary study in this new area of automatically assessing reviewing performance, we have demonstrated the feasibility of detecting reviewers who have low problem localization in reviewing, which is a first step for enhancing peer feedback quality. From the perspective of data mining, we successfully mine features of problem localization patterns from free form textual comments using statistical NLP techniques. Though we have started with simple methods and our classifiers are based on shallow NLP features, our system achieves comparatively high accuracy and precision for identifying reviewers who generally fail to provide localization information in their reviews.

In the future, we hope to construct a more complete dictionary of domain vocabulary, which might provide us with a better result based on our observations from this work. To improve the generalization of our system, we would also like to use a data driven approach to generate the Keyword list fully automatically (Table 2). Clearly each component in our system could be a NLP research topic, so we plan to explore the use of more sophisticated models from the NLP community as we discussed in the related work. Finally, since our ultimate goal is to help students with reviewing, we would like to perform a follow-up study to further evaluate how helpful the assessment generated by our system is in term of improving problem localization in future peer-review exercises for our “low” reviewers.

References

- [1] Kluger, A. N. & DeNisi, A. The effects of feedback interventions on performance: A historical review, a meta-analysis, and a preliminary feedback intervention theory. *Psychological Bulletin*, 1996, 119(2), 254-284.
- [2] Nelson, M.M. & Schunn, C.D. The nature of feedback: how different types of peer feedback affect writing performance. *Instructional Science*, 37, 2009, 375-401.
- [3] Cho, K. Machine classification of peer comments in physics. *Educational Data Mining*, 2008.
- [4] Wilson, T., Wiebe, J. M. & Hoffmann, Paul. Recognizing contextual polarity in phrase-level sentiment analysis. *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, 2005, 347–354.
- [5] Malakasiotis, P. Paraphrase recognition using machine learning to combine similarity measures. *Proceedings of the 47th Annual Meeting of ACL and the 4th Int. Joint Conf. on Natural Language Processing of AFNLP*, 2009.
- [6] McDonald, R., Crammer, K. & Pereira, F. Online large-margin training of dependency parsers. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, 2005.
- [7] Ernst-Gerlach, A. & Crane, G. Identifying quotations in reference works and primary materials. *Research and Advanced Technology for Digital Libraries*, 2008, 78-87.
- [8] Xiong, W. & Litman, D. Identifying problem localization in peer-review feedback. *Tenth International Conference on Intelligent Tutoring Systems*, 2010.

Using Numeric Optimization To Refine Semantic User Model Integration Of Educational Systems

Michael Yudelson¹, Peter Brusilovsky¹, Antonija Mitrovic², Moffat Mathews²
{mvy3, peterb}@pitt.edu, {tanja.mitrovic, moffat.mathews}@canterbury.ac.nz

¹University of Pittsburgh, School of Information Sciences

²University of Canterbury, Department of Computer Science and Software Engineering

Abstract. Nowadays, it is a common practice to use several educational systems in one domain. In this situation, each of the systems should be able to provide the best user modeling based on the integrated data about the user. However, differences in domain conceptualization complicate the ability of the systems to understand each other's user models and necessitate the use of labor intensive and time consuming alignment procedures that require involvement of knowledge engineers. While the latter are best at detecting associative links between the user model items, they fail to reliably identify the strengths of these associations. In this paper, we are proposing a method to improve the user model mapping by using a numerical optimization procedure. Our results show that numerical weight optimization helped to decrease the amount of manual work and improved the target model accuracy.

1 Introduction

Over the recent decade, many state-of-the-art user-adaptive systems evolved from research prototypes to real-life production applications. Social interaction, information access, and E-Learning are the areas where the user-adaptive content is booming. In some fields (educational systems for example), the concentration of user-adaptive systems is such that several are available for each topic. Despite their availability, adaptive systems do not compete but complete each other, implementing only part of an ideal fully functional system. They offer unique features unable to completely replace their rivals. This creates a problem in using them together to embrace the full range of features offered separately. To enable coherent use, systems should be able to mutually exchange and understand collected user data. In an ideal scenario, user models should be integrated.

There are two major approaches to user model integration. The most widespread and popular is translation [15] (or mediation [1]). Here, systems exchange complete snapshots of their user models and *convert* them into local representations. The operation of translation is costly. However, since translation/mediation is usually employed when the systems are used sequentially, these costs stay insignificant. When systems are used in parallel, holistic translation/mediation poses a problem when large chunks of user data need to be exchanged and converted (even after minor updates to one of the models).

In the situation of parallel systems' use, evidence-based integration is more appropriate, because user-adaptive systems do not exchange whole user models. Instead, they share results of elementary user actions. When one user model is updated, only the changes are

conveyed to the other system's user model. The cost of conversion is lower as fewer user model items require conversion. Although, in the case of the evidence-based approach, the problem of user model integration is of a seemingly smaller scale; the question of user model conversion is still quite complicated. This is especially true for the field of adaptive education systems (AES). In AES, each learning object (problem, example, test) is commonly described in terms of constructs of the domain model often called concepts. In reality, systems seldom share the representation of the models, even in closely related domains. Under these conditions, evidence integration becomes a challenging task.

Because of the differences in the domain model representations, concepts from one system could be related to several concepts from the other; they may feature different strengths of relations. To be able to convert user model data between systems, researchers often employ domain experts and knowledge engineers who manually identify links between domain models of the systems and assign strengths to them. Although human experts are known to be able to find related complex relations between domain models effectively, they do have problems assigning numerical weights for the strengths [6]. The very same expert could change his/her weighting decision if the process is repeated.

In this paper, we are addressing the problem of evidence-based user model integration for two educational systems in the area of databases: SQL-Tutor [11] and SQL KnoT [14]. Our prior work on integrating the user models of these systems has shown that expert involvement only partially solves the problem of integration [2]. While experts agree on the concept associations, assigning weights is extremely time consuming and often leads to conflicting results. To solve these problems, we are proposing an optimization procedure intended to aid experts in mapping domain models of the two systems. This procedure is designed to significantly cut the amount of efforts and time, optimize the mapping weights, and potentially refine the structure of the concept mapping links.

The rest of the paper is organized in the following way. Section 2 addresses related work. Section 3 talks about the ongoing integration of the two systems in question, identifies a problem, suggests a solution, and lists our hypotheses. Section 4 discusses the experimental work. And finally, section 5 finishes with discussion points.

2 Related Work On User Model Integration

The task of making two adaptive systems understand each other's user models is quite challenging. It requires aligning the domain vocabulary of one against the other, so that both systems can correctly interpret assumptions about the users from the partner system and employ user data to improve their own performance. Domain model alignment leading to a successful integration requires resolution of multiple inconsistencies. The less complex are the inconsistencies at the language-level. Among these are differences in syntax, differences in clarity, varying use of semantics, etc. The more complex inconsistencies are the model-level mismatches occurring due to the discrepancy in structure and/or semantics of the domain models. Resolving these kinds of inconsistencies entails dealing with: naming conflicts (the same concept termed differently in two models or the same term defining different concepts); different graph structure (relevant sets of concepts connected differently); different scopes (e.g. two

models covering different parts of the domain); different granularities (a single concept of one model covering a portion of domain knowledge represented by several concepts in another model); and adherence to different modeling paradigms and conventions.

Common ontology. User models can be exchanged and mutually interpreted when user models of two systems rely on a common domain ontology. A good example of common ontology integration is the OntoAIMS project [7]. Here, two separate systems OWL-OLM and AIMS are deployed with mutual regard for interoperability. Both systems represent their domain models in the form of ontologies and user models – as overlays of these ontologies. The shared user model is populated and read by both systems. Another approach to common ontology integration is to devise a central user modeling server (e.g. Personis [10], CUMULATE [4]). Such servers store domain models, perform centralized user modeling, and deliver user data to participating adaptive systems.

The intermediary ontology is used when systems agree on a single ontology. Regrettably, the field of user-adaptive systems is far from producing ontologies, which would be widely accepted. It is often rather difficult to make an ontological commitment, as different research teams design ontologies with different views of the conceptualization of the domain. A good example of the use of an intermediary ontology is the M-OBLIGE architecture [12]. Here, a common reference ontology of SQL works as a multi-translator and facilitates the exchange of the user model information.

Automatic ontology mapping. The approaches presented above are practical solutions for semantic integration of multiple user-adaptive systems. However, their applicability is reduced by the need to either adapt to an alternative representation of the domain or to perform time-consuming and mostly manual knowledge engineering work. Nevertheless, cases exist where the use of ontologies for domain modeling permits automatic alignment them [9]. Automation helps to find matching concepts and relies on techniques of natural language processing, graph theory, information retrieval, and statistics to discover similar lexical patterns, conceptual sub graphs, and regularities in accompanying text. Fully implemented ontology mapping solutions are not yet known. But authors of [15], when investigating the applicability of automatic ontology mapping, have shown that it has the potential to be close to the best possible translation done by human experts.

Evidence-based integration. To successfully integrate user-adaptive systems, one does not have to align entire ontologies each time. Instead, integration could be performed on single units of user information: reports of user activity, assumptions about user knowledge, preferences, and goals – all described in terms of domain model concepts. This data arrives to the user model sequentially as a series of events. Translation is applied to one or a set of events as shown in Figure 1. Model values are often exchanged as soon as they are produced by one of the systems. An example of the evidence-based integration is the work on integrating Ramapo Proplets and QuizJET systems [3]. Both systems have internal domain ontologies of Java programming and the ontologies are different in granularity and focus of modeling. Integration of user data is done on the side of QuizJET. QuizJET uses the ontology concepts in the classical descriptive metadata sense, while Ramapo Proplets uses the concepts-in-a-context paradigm, where each concept is accompanied by a descriptor of its situational application (e.g. for-loop vs. for-

loop-executing-exactly-once). This leads to the situation when each of the Ramapo Problents' concepts-in-context is related to a sizable weighted set of QuizJET concepts.

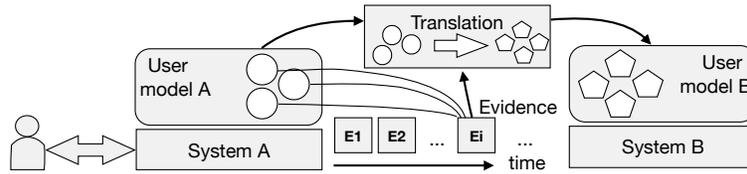


Figure 1 Evidence-based integration of two user models

The problem with all of the approaches above is that they heavily rely on time-consuming work on the part of experts to produce the mapping function. While experts are best for selecting the semantic relations between concepts of the domain models, the quality of their assignment of strengths to the inter-concept links is sub-optimal [6]. In our work, we are seeking to improve user model integration by utilizing an optimization procedure to either refine expert mapping weights or to produce them without human involvement, potentially altering the expert-suggested mapping links.

3 From Manual To Automated Student Model Mapping

3.1 Model Mapping For Evidence-Based Integration

In our previous work, we have dedicated a lot of attention to evidence-based integration of user models in the SQL domain. This work revolved around two systems: SQL Knot and SQL-Tutor. SQL KnoT [14] is an external-loop tutor serving parameterized problems testing students' knowledge SQL. SQL KnoT represents domain model in the form of an ontology that has been developed by a team of human experts. Each problem is semi-automatically indexed with a set of ontology concepts. SQL-Tutor [11] is an ITS that presents problems to students and helps them to improve their knowledge of SQL. SQL-Tutor represents a domain model in the form of *constraints*. Constraints represent the fundamental principles of SQL and must be satisfied in any correct solution. Each SQL-Tutor problem has a set of relevant constraints. If the student solution violates these constraints, the solution is incorrect. The constraint set in SQL-Tutor contains about 700 constraints that assess the syntactic and semantic correctness of the solution.

The gist of the user modeling approach in both SQL KnoT and SQL-Tutor is quite similar: they construct an overlay of the domain vocabulary of knowledge items (constraints or concepts); nevertheless, the fundamental differences between the two domain models make the alignment of their models quite challenging. The unique nature of SQL-Tutor constraints makes the use of the known automatic ontology mapping techniques impossible. A constraint is not directly related to a single concept or a sub-tree of the ontology; instead, it models the syntactic or semantic relations between various concepts. As a result, the mapping is not one-to-one, but many-to-many.

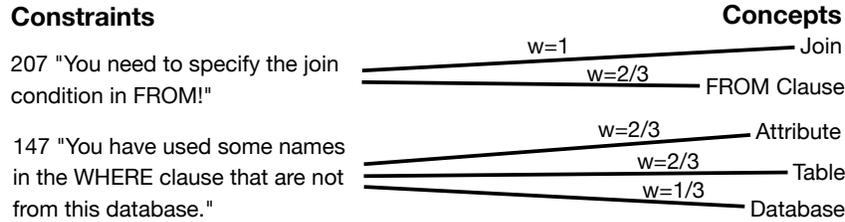


Figure 2 A fragment of constraints-to-concepts mapping with weights

We charged a number of experts with the task of mapping the two domain representations [13]. The resulting mapping contained a set of constraint-concept relations with the relevance weights: small (1/3), medium (2/3), or large (3/3=1). A fragment of the mapping is shown in Figure 2. The formula for computing the concept knowledge scores is shown in Equation (1). Namely, sum of weights between the concept and satisfied constraints minus sum of weights between the concept and broken constraints, divided by the sum of weights between the concept and all activated constraints.

$$S_{concept} = \left(\sum W_{(concept,constraint)}^+ - \sum W_{(concept,constraint)}^- \right) / \sum W_{(concept,constraint)} \quad (1)$$

Both systems were deployed in an undergraduate database course at the School of Information Sciences, University of Pittsburgh during the Fall 2008 semester. Students had access to 300 SQL-Tutor problems and over 50 parameterized SQL KnoT problems. The student's progress was stored in one single long-term user model. SQL-Tutor was responsible for nearly a quarter of the students' problem-solving activity (SQL KnoT was responsible for the rest), despite that fact that SQL-Tutor was made available only in the middle of the semester and students were already familiar with SQL KnoT.

3.2 Problems Of Manual Mapping

Despite tangible progress in integrating the two SQL problem-solving tools, there are still open questions pertinent to this particular integration effort and to user model integration in general. First, merging the user models leaves the issue of the merger quality unanswered. The fusion of the user models does not automatically improve the quality of the combined model. Second, we relied on expert opinion to come up with the mapping. From the literature, we know that the experts can reliably identify mapping relations between items of two domains [8]. However, assigning an appropriate weight – even on a simpler categorical scale – often poses a problem [6]. The same person can change his/her opinion if the weighting procedure is repeated. This means that, even if all relations between concepts and constraints were identified correctly, the assigned weights could still be suboptimal; as a result, the quality of the mapped model would be poor.

3.3 Towards Automated Model Mapping

To answer these questions, we suggest using an optimization procedure to refine the expert-assigned constraint-to-concept weights. The idea is to employ the student logs of the source system (SQL-Tutor) to create and fine-tune a custom user model using least

square fitting method with the mean squared error as the criterion. Then, using the same student logs, we utilize the experts' mapping to compute the mean squared error of the now mapped concept-based user model. Performing the search in the weight-space, we are minimizing the error of the target model, thus refining the mapping.

Our hypothesis is that the suggested procedure would be able to, first, improve user model integration by optimizing expert weights. Second, starting with constant or random weights, produce the weights, without considering expert suggestion. Here, we expect the optimization results to be not as good as the one starting with expert mapping. And third, change the mapping links structure (by setting weights =0) for a more optimal one.

4 Experimental Study Of Automated Model Mapping

To verify the validity of the approach suggested, we conducted a set of experiments. First of all, to establish the baseline for mapping, we constructed the source constraint-based user model in such a way that modeling parameters minimize the modeling error. Second, the optimization was repeated. This time, the variables were the mapping weights and the objective function was the mean squared error of the mapped concept-based model.

Logs of the three database courses offered at the University of Canterbury in the 2006-2007 term were used for the experiment. We took the logs of the first course that contained 3544 transactions of 38 students. Each log entry contained user id, problem id, time the solution was submitted, solution correctness, list of confirmed constraints, and a list of broken constraints (if the solution is not correct). At the first stage of the experiment, we used user modeling approach that was different from the one deployed with SQL-Tutor, namely – Bayesian Knowledge Tracing (BKT) [5] – an established user modeling method in the area of intelligent tutoring systems. In the second stage, we used three sets of weights: supplied by the experts, equal constant weights, and the random weights. The details and the results of these procedures are given below.

4.1 Baseline Constraint-Based User Model

As we have mentioned before, Bayesian Knowledge Tracing (BKT) was used as the baseline user modeling approach in SQL-Tutor. BKT assumes a two-state model of knowledge items (often called skills or rules) of a particular learning domain. The knowledge item (KI) is either in a learned or unlearned state. While interacting with the system, knowledge of KI can transition from the unlearned to the learned state. Even if a KI is in the learned state, a student can make a mistake. As in the unlearned state, there is a chance student can guess correctly. For each of the modeled KIs, BKT model maintains four parameter estimates: $p(L)$ – the probability that KI is in the learned state, the probability that KI is in the learned state prior to interacting with the tutor is $p(L_0)$; $p(T)$ – the probability that the KI will transfer to the learned state on next time user practices it; $p(S)$ – the probability the student will slip and apply the KI incorrectly even when it is in the learned state; and finally $p(G)$ – the probability that the student will apply the KI correctly despite it being in the unlearned state. For details on BKT models, refer to [5]

We adhered to the usual practice of BKT modeling and kept a unique set of parameters $p(L_0)$, $p(T)$, $p(S)$, and $p(G)$ for each KI and had a separate running estimate of $p(L)$ for each user-KI pair. Counter to the tradition, we didn't assume conditional independence of KIs and obtained estimates for all parameters together. Reason being the *special feature* of constraint-based model: constraints always spanned several domain concepts. One other usual BKT practice is using only the first chance to apply the knowledge item per problem attempt. This rule is often used in the so-called *inner-loop tutors* that walk students through each step of the problem solving activity. SQL-Tutor is an *outer-loop tutor*: it gives feedback only when a complete problem solution has been submitted. It is common to see same constraint both satisfied and broken in the problem several times not knowing which came first. We ignored repetitions, and when the constraint was both satisfied and broken, set the value of $p(L)$ to the arithmetic mean of $p(L)^+$ (assuming correct response came first) and $p(L)^-$ (assuming incorrect response came first).

Table 1 Results of fitting source BKT model of SQL-Tutor constraints.

Parameter	Value	Parameter	Value
p(L0) – a priori knowledge level	0.642	p(S) – probability of slip	0.206
p(T) – probability of transfer	0.523	p(G) – probability of guess	0.187
Mean Squared Error of modeling	0.248		

Out of about 700 constraints, we selected only those occurring in the user logs. In addition, we filtered out constraints that are always satisfied and never broken. This gave us 282 constraints with 4 parameters to estimate for each: a total of 1,128 parameters. Each constraint was treated as a separate KI. Mean squared error was used as an objective function of the optimization. To compute the BKT models, we used Matlab. Fitting was done using Matlab's implementation of linear square fitting procedure using trust region reflective algorithm. The parameters of the resulting source BKT model of constraints are shown in Table 1. Model parameters given are weight-averaged over all participating constraints (the number of constraint occurrences in the log weights its contribution).

4.2 Mapping Weights Optimization

To optimize the model mapping weights, we modified the code used to construct the baseline constraint-based BKT model. User model parameters were held constant and mapping weights were used as variables. Objective function remained the same – mean squared error of modeling. The original constraint-based user model was updated as before, but for the computation of the user model error it was mapped to concept-based model using Equation (1). Since in the previous stage we reduced the number of participating constraints, the original number of 1,012 constraint-concept links/weights decreased to 576. Three sets of weights were used. The first set contained weights produced by experts: low, medium, or high (1/3, 2/3, and 1=3/3 respectively). In the second set, all weights were set to 0.5. In the third set, all weights were randomly chosen from 0.0 to 1.0. As in the case with BKT modeling of constraints, we did not assume conditional independence of sets of weights and all of the 576 weights were fit together. Matlab's constrained nonlinear multivariable optimization procedure was used with an active-set algorithm. Table 2 presents the results of the optimization experiments.

Table 2 Results of weight mapping refinement

	Expert weights	Constant weight = 0.5	Random 0.0 to 1.0
Mean squared error before refinement	0.454	0.453	.463
Mean squared error after refinement	0.290	0.270	.463
Number of weights changed	44	576	0
Mean absolute weight change	0.031	0.496	0
Weights decreased / increased	44 / 0	52 / 524	0
Weights set to zero values	34	52	0

As we can see, under all three starting conditions the initial mean squared error was roughly the same: 45%-46%. However, only the first two conditions (originating from expert weights and from constant weights) lead to improved modeling error rate. Optimization of the expert-supplied weights ended with a 29% error rate, only 4% worse than the original constraint-based BKT model. The mapping originating with constant weights of 0.5, counter to our expectations, beat that figure by 2% and reached a 27%. Optimization of random weights did not lead to any changes.

Although expert weights were closer to the found optimum (44 out of 576 weights were changed in the process of optimization and all of them were decreased) than constant weights (all weights were changed, 524 weights increased and 52 decreased), in both cases the optimization procedure arrived at similar results. In the process of optimization, 34 and 52 mapping links were removed in the expert and constant conditions respectively. Despite promising results, our solutions look like local optimums. Mean absolute difference of weights between expert-originated and constant-originated sets is 0.3581 and the number of removed links agreed upon is only 13.

5 Discussion

Our experiments have successfully shown the high potential of using numeric optimization to refine the expert's alignment of two user models in general. In our case the mapping of the source constraint-based mode of SQL-Tutor to the concept-based model of SQL KnoT was significantly improved. The original 45% error of the mapped model was reduced to 29%; only 4% shy of the source model's error. Taking into account principal differences between the two domain vocabularies mapped, the achieved result is impressive. The refinement of uniformly assigned constant weights, against our expectations, was able to achieve an even better 27% error. Although a 2% difference does not seem like a tangible one, what counts is that refining constant weights was comparable to the original constraint-based model. In the light of these results, in the future one could free experts from weight assignment entirely. The only thing left for experts to do is to set inter-model relations. The rest could be handled by the procedure we proposed. The optimization procedure also demonstrated the ability to not only change the weights, but also to drive them to zero as an indication of some of the model mapping links to be *redundant*. This could serve as an additional check of the mapping quality and as an aid to human experts in the by-hand iterative process of mapping.

Despite its merits, there are several limitations to this work. First, mapping links could only be removed, not added as in some automated mapping approaches. A possible remedy for this could be changing the link selection procedure when merging the experts' suggestions. Instead of using the relations agreed upon by the majority of experts, all suggested links could be considered and then the refinement procedure would do the necessary filtering. In general, additional expert verification of the refined mapping might be necessary. Computationally optimal mapping might not be pedagogically sound. Inappropriate mapping links might be emphasized by the use of maximal weights and important links could be removed. Second, we did not assume conditional independence of mapping weights and optimized all 576 weights at once. This posed a computation challenge for the optimization procedure. Also we used only a part of SQL-Tutor logs, one course worth of logs out of three available. A limited number of data points (3544) might have been a contributing factor for entrapment in the local minima of the objective function and possibly over-fitting. Mining large volumes of user data and enforcing conditional independence of the mapping weights could solve both of these issues. Third, the user model transformation formula inherently normalizes weights and scaling all mapping weights has no effect. For example, a set weights equal to $\{1, 2, 3\}$ would be equivalent to $\{.1, .2, .3\}$ or $\{.2, .4, .6\}$. However, optimization procedure treats scaled weights as completely different. Changing the transformation formula or the optimization procedure for the ones that don't suffer from this phenomenon could further improve the mapping. And finally, both the proposed procedure and the refined weights it produced could possibly be sensitive to a particular user modeling approach employed (Bayesian Knowledge Tracing in our case). At this point we cannot confirm whether the obtained weights would remain optimal, if modeling formalisms are replaced with alternative ones. We plan to continue this work and further investigate the issue.

References

- [1] Berkovsky, S., Kuflik, T., and Ricci, F. (2006). Cross-Technique Mediation of User Models. In V. P. Wade, H. Ashman, and B. Smyth (Eds.), 4th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2006), (pp. 21-30).
- [2] Brusilovsky, P., Mitrovic, A., Sosnovsky, S., Mathews, M., Yudelson, M., Lee, D., and Zadorozhny, V. (2009). Database exploratorium: a semantically integrated adaptive educational system. In S. Berkovsky, F. Carmagnola, D. Heckmann, and T. Kuflik (Eds.), 7th International Workshop on Ubiquitous User Modelling (UbiqUM 2009).
- [3] Brusilovsky, P., Sosnovsky, S., Yudelson, M., Kumar, A., and Hsiao, S. (2008). User model integration in a distributed adaptive E-Learning system. In S. Berkovsky, F. Carmagnola, D. Heckmann, A. Krueger, and T. Kuflik (Eds.), International Workshop on User Model Integration, (pp. 1-10).
- [4] Brusilovsky, P., Sosnovsky, S. A., and Shcherbinina, O. (2005). User Modeling in a Distributed E-Learning Architecture. In L. Ardissono, P. Brna, and A. Mitrovic (Eds.), 10th International Conference on User Modeling (UM 2005), (pp. 387-391).

- [5] 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.
- [6] Dawes, R. (1979). The robust beauty of improper linear models in decision making. *American Psychologist*, 34(7), 571-782.
- [7] Denaux, R., Dimitrova, V., and Aroyo, L. (2005). Integrating Open User Modeling and Learning Content Management for the Semantic Web. In L. Ardissono, P. Brna, and A. Mitrovic (Eds.), *10th International Conference on User Modeling (UM 2005)*, (pp. 9-18).
- [8] Giarratano, J. C. and Riley, G. (1989). *Expert Systems: Principles and Programming*. Boston, MA: PWS-Kent Publishing Co.
- [9] Kalfoglou, Y. and Schorelmmmer, M. (2003). Ontology Mapping: the State of the Art. *The Knowledge Engineering Review*, 18(1), 1-31.
- [10] Kay, J., Kummerfeld, B., and Lauder, P. (2002). Personis: A Server for User Models. In P. D. Bra, P. Brusilovsky, and R. Conejo (Eds.), *2nd International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, Second (AH 2002)*, (pp. 203-212).
- [11] Mitrovic, A. (2003). An Intelligent SQL Tutor on the Web. *International Journal of Artificial Intelligence in Education*, 13(2-4), 173-197.
- [12] Mitrovic, A. and Devedzic, V. (2004). A Model of Multitutor Ontology-based Learning Environments. *Continuing Engineering Education and Life-Long Learning*, 14(3), 229-245.
- [13] Sosnovsky, S., Mitrovic, A., Lee, D. H., Brusilovsky, P., Yudelson, M., Brusilovsky, V., and Sharma, D. (2007). Towards integration of adaptive educational systems: mapping domain models to ontologies. In N. Capuano, D. Dicheva, A. Harrer, and R. Mizoguchi (Eds.), *5th International Workshop on Ontologies and Semantic Web for E-Learning (SWEL'2007)*.
- [14] Sosnovsky, S. A., Brusilovsky, P., Lee, D. H., Zadorozhny, V., and Zhou, X. (2008). Re-assessing the Value of Adaptive Navigation Support in E-Learning Context. In W. Nejdl, J. Kay, P. Pu, and E. Herder (Eds.), *5th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH 2008)*, (pp. 193-203).
- [15] Sosnovsky, S. A., Dolog, P., Henze, N., Brusilovsky, P., and Nejdl, W. (2007). Translation of Overlay Models of Student Knowledge for Relative Domains Based on Domain Ontology Mapping. In R. Luckin, K. R. Koedinger, and J. E. Greer (Eds.), *13th International Conference on Artificial Intelligence in Education (AIED 2007)*, (pp. 289-296).

An Annotations Approach to Peer Tutoring

John Champaign and Robin Cohen

jchampai@cs.uwaterloo.ca, rcohen@ai.uwaterloo.ca

David R. Cheriton School of Computer Science, University of Waterloo

Abstract. In this paper we detail a preliminary model for reasoning about annotating learning objects and intelligently showing annotations to users who will benefit from them. Student interactions with these annotations are recorded and this data is used to reason about the best combination of annotations and learning objects to show to a specific student. Motivating examples and algorithms for reasoning about annotations are presented. The proposed approach leverages the votes for and against an annotation by previous students, considering whether those students are similar or dissimilar to the current student, in order to determine the value of showing this annotation to the student.

1 Introduction

McCalla's ecological approach to e-learning systems [8] is described as “attaching models of learners to the learning objects they interact with, and then mining these models for patterns that are useful for various purposes.” Using techniques inspired by collaborative filtering, the basis of this approach is to identify which users in a system are similar to each other, to then preferentially recommend learning objects that similar students have found useful. Learning objects are used in the ecological sense, in that instead of attaching static ontologies to objects as metadata, models of previous interactions (i.e. presenting learning objects to students) are used in their place. These models are then actively interpreted, with the meaning derived by real-time processes, as information about the object is needed.

Since McCalla's ecological approach is intended to primarily be a general philosophy for designing intelligent systems, individual researchers may then create their own algorithms and systems to embody this approach.

To honour the basic evolutionary approach, we are particularly interested in exploring the use of student annotations: allowing students to leave short comments on learning objects they are interacting with. More than simply tags, this could be a question or a commentary about what they're learning. Subsequent students would identify which annotations they found useful, which would then be intelligently shown to similar students. Asynchronous collaboration or, at least, to allowing the interactions of the student in the past to inform the interaction with the current student, honours the ecological approach [8].

In developing a model for reasoning about which learning objects and annotations to show to students, we are thus focused on mining the repository to improve the educational data sets of use in the tutoring process.

2 Background

In McCalla's ecological paradigm for intelligent tutoring system, the basis for tutoring is a repository of learning objects (which, for example, could be a chapter from a book, a video or an exercise which is completed and evaluated), which are provided to students to interact with. Information about these interactions is used to reason about the best learning objects to be shown in the future to other students, based on the utility of past interactions of similar students with the various objects in the system.

In previous research, we explored how best to select learning objects to present to students by reasoning about the benefits derived by previous students from these learning objects [4]. Beyond implicitly reasoning about interactions, in this work we consider extending this paradigm so as to allow students to explicitly leave information for future students. This takes the form of “annotations”, or short text messages, which are attached to the learning object and intelligently shown to (or hidden from) future students. As a motivating example, suppose a student Carol in a Computer Science 101 course was struggling with the concept of procedures. She suddenly realizes that procedure is another name for a function, which is the term her high school teacher had used. She leaves a short annotation (“OMG, I realized half way through the lesson that procedures are the same as functions!!! duh! :-)”) on the learning object she was studying with this insight and carries on with her lessons. Future students are shown this annotation and given the option of endorsing it (by clicking on a “thumbs up”) or denouncing it (with a “thumbs down”). Over time, the system learns that the annotation tends to be useful to students with a background that used the term “function” instead of “procedure” (and shows it to them), but not to others (and hides it from them).

3 Our Approach

For the actual mechanics of our approach, we use a set of algorithms for dealing with the annotations. Our approach is as follows. We model the overall reputation of a student, initially set as 0.5 and afterwards based on the extent to which the student's previous annotations have been found useful by other students. The authoring student's reputation is used as the initial reputation of an annotation, which is afterwards independently tracked and modeled with its own reputation. Students' ratings of the value of the annotation serve to adjust the overall reputation of the annotation (and indirectly, the annotation author's reputation). Once a student has provided a set of ratings on annotations, it is then possible to reason about their similarity to other students, based on mutually rated annotations. This allows the probability that an annotation be shown to be determined by both the overall quality of that annotation, as rated by all students, adjusted by the similarity between the current student and the students who have previously rated the annotation.

3.1 Annotations Algorithms

```
Function calculateStudentReputation
  Arguments: Student
  if (numberOfAnnotationsCreatedByStudent == 0)
    // If the student hasn't made any annotations,
    // assign their reputation to 0.5
```

```

        Reputation = 0.5
    else
        Reputation = 0
        // Calculate the average reputation of all annotations
        // made by the student
        for each previous annotation left by student
            reputation += annotation.reputation
        end for
        Reputation /= numberOfAnnotationCreatedByStudent
    Return: Reputation (0.0:1.0)
Endfunction

```

Example:

Bob has left his first annotation (a question asking for the definition of a term on the slide) on a CS 115 slide and no one has rated it yet. His reputation is 0.5.

Carol has left three annotations, two on slides and one on a video of the CS 115 instructor explaining Scheme structures (each one discussing how the concepts relate to more advanced concepts she has read independently). After receiving ratings from students presented with the learning object these annotations have reputations of 0.476, 0.551 and 0.704. Carol has a reputation of 0.577 (the average of these 3).

```

Function makeAnnotation
    Arguments: Student, Learning Object, Annotation
    student.reputation =calculateStudentReputation(student)
    annotation.reputation = student.reputation
    // The annotating student will be considered to implicitly give
    // their own annotations a positive recommendation
    evaluateLearningObjectAnnotation(student, annotation)
    learningObject.attach(annotation)
    Return: nothing
Endfunction

```

Example:

Allison and Carol each leave an annotation on a new set of example exercises that have been posted (Allison asks if the solutions have been posted, and Carol replies that they will be posted on the March 18th after the assignment due date). Allison's new annotation starts with a reputation of 0.5 (no one has rated his first annotation yet – Carol didn't bother) and Carol's has a reputation of 0.557 (there haven't been any additional ratings to her annotations since her reputation was calculated).

```

Function similarity1
    Arguments: Current Student, Annotation Student
    votedTogether = 0
    votedAgainst = 0
    for each annotation both students voted on
        if (currentStudent.vote == annotationStudent.vote)
            votedTogether += 1
        else
            votedAgainst += 1
    end for

```

¹ Students are considered “similar” to one another if they tend to benefit in the same way from the same learning objects. This straightforward comparison can be expanded as future work.

```

    end for
    similarity = (votedTogether - votedAgainst)
    similarity /= (votedTogether + votedAgainst)
    Return: Similarity [-1.0:1.0]
Endfunction

```

Example:

Bob has given a “Thumbs up” (indicating approval) to annotations A, B, F & G and a “Thumbs down” to annotations D, M & Y. Carol has given a “Thumbs up” to annotations F, M, N, and Z and a “Thumbs down” to C, G, and Y.

Bob and Carol's similarity is:

$$\frac{2(\text{they agreed on their assessment of } F, Y) - 1(\text{they disagreed on their assessment of } G)}{2 + 1(\text{the number of annotations both have rated})}$$

or 0.5

```

Function show annotation
Arguments: learning object, student
for each annotation attached to learning object
    // base chance of showing is based on author's
    // reputation AT THE TIME THEY MADE the annotation.
    votes_for = annotation.votes_for
    votes_against = annotation.votes_against
    for each vote on annotation
        sim = similarity(currentStudent,
                        annotationStudent)

        if vote.for
            votes_for += 1 * sim
        else
            votes_against += 1 * sim
    AT = arctan(votes_for - votes_against) / pi
    chanceToShow = 0.5 + AT
    randomNumber = generateRandomNumber[0.0 : 1.0]
    if (randomNumber < chanceToShow)
        show annotation to student
        // implicit in showing an annotation is that
        // any parent annotations will also be
        // shown, this is applied recursively up to
        // a root annotation
    Return: boolean
Endfunction

```

Example:

Bob is viewing the learning object annotation Z is attached to. It has a reputation of 0.670 and Carol is the only other student he has a similarity rating with who has previously rated it. She gave it a thumbs up, which leads to the annotation being given a reputation of $0.670 * (1 + (0.05 * 0.5)) = 0.687$ for Bob. This means there is a 68.7% chance that this annotation will be shown to him when he uses the learning object.

In the above example, Bob's annotation has a $(1 - (1 * 0.5 * 0.453))$ or 0.774 (since it may be shown on its own, or will be shown if Carol's reply is shown).

```
Function evaluate learning object annotation
Arguments: Learning object, student
if (student votes on some annotation)
    votedAnnotation.attach(vote, student)
    votes_for = tallyOfForVotes(annotation)
    votes_against = tallyOfAgainstVotes(annotation)
    AT = arctan(votes_for - votes_against) / pi
    annotation.current_reputation = 0.5 + AT
end if
Return: nothing
Endfunction
```

Example:

While Bob uses the learning object, he gives the annotation Z a “Thumbs Down” (he doesn't find it useful). This lowers the annotation's reputation from 0.670 to 0.638. His similarity to Carol is also lowered (now that they disagree on 2 annotations) and they will now have a similarity of 0 (their preferences will no longer influence one another). If either gives another rating that conflicts with the other, they will begin to negatively recommend annotations for one another (what one likes will be less likely to be shown to the other).

The formula which determines whether an annotation is shown or not:

$$Probability\ Annotation\ is\ Shown = \frac{1}{2} + \frac{\arctan(votes_{for} - votes_{against})}{\pi} \quad (1)$$

has a number of attractive properties.

3.2 Larger Number of Votes Given Greater Weight than Smaller Number

Consider 3 students as voting in favour of an annotation and 5 against it and contrast this with 30 students voting in favour and 50 voting against it. In each case, 37.5% of the student found the annotation useful. However, given that 10 times as many students have voted on the annotation in the second case, we want to take into account the greater certainty in the outcome given the larger number of voters.

Contrast m votes for and n votes against and $k \cdot m$ votes for and $k \cdot n$ votes against.

Formula (1) is a linear scaling of $\arctan(votes_{for} - votes_{against})$. Given that \arctan is a strictly increasing monotonic function, $\arctan(k(m - n)) > \arctan(m - n)$ and our approach will give a greater weight to larger groups of voters.

3.3 Probability Approaches But Doesn't Reach 0 and 1

Consider:

$$\lim_{x \rightarrow \infty} f(x) = \frac{1}{2} + \lim_{x \rightarrow \infty} \frac{\arctan(votes_{for} - votes_{against})}{\pi} = 1$$

$$\lim_{x \rightarrow -\infty} f(x) = \frac{1}{2} + \lim_{x \rightarrow -\infty} \frac{\arctan(\text{votes}_{for} - \text{votes}_{against})}{\pi} = 0$$

Given that arctan is a strictly increasing monotonic function, we know that the probability that an annotation will be shown or not approaches, but never reaches 0 and 1. This is worthwhile as the usefulness of an annotation may be discovered at a later date, and it is then given a chance to be promoted. Conversely, if a well-regarded annotation is shown to be vacuous, the community has a chance to immediately begin decreasing its prominence.

Consider the example of a clarifying annotation where a student made the connection, in a video that explains parameters, that procedure another name for a function. After this annotation was given high ratings, a new article was added which explained functions and clearly presented the various terms such as routine, function, procedure or method. After having read this article, students begin finding the previously useful annotation redundant and it begins to receive negative ratings from current students. The system is immediately responsive to this and each negative vote decreases the probability that this (once highly-regarded) annotation is shown to current students.

4 Validation of Approach

Our intention is to validate this work using simulated students.

Let knowledge be defined as the known concepts in the domain under consideration (the course the ITS endeavours to educate the student on).

$$k = \{ \text{set of known concepts} \}$$

After an interaction between student s and learning object l , there will be a set of relationship such that:

$$\text{if } k_s \in k_l \text{ then } k_s = k_s \cup k_{new} \text{ with probability } p$$

e.g. suppose learning object abc had the relationship:

$$\text{if } k_s \in \{B, J, U\} \text{ then } k_s = k_s \cup \{M\} \text{ with probability } 0.25$$

This would imply that if a student using this learning object had attained concepts B, J and U, then upon completion of using this object he would have a 25% chance of attaining concept M.

Let overall knowledge (K) be represented as a percentile, considered roughly analogous to the student's expected mark given their current understanding.

$$K = \frac{(\text{Known Concepts})}{(\text{All Concepts})}$$

e.g. suppose an ITS had 26 concepts, each represented by a letter the alphabet. Given a student who had obtained concepts B, J, M and U, their overall knowledge would be:

$$K = \frac{(\{B, J, M, U\})}{(\{A, B, C, \dots, Z\})} = \frac{4}{26} = 15\%$$

The goal of the system is to maximize the average K of student's using the system.

5 Annotation

An annotation will modify the relationships of a learning object in one of two ways.

1. Create a new relationship with a substitute, removed concept
2. Increase or decrease the probability of attaining the new concept for an existing relationship

5.1 Example

Student Amy annotates a chapter from a text book that was assigned to her by the system. Her annotation has the effect of creating a new relationship, based on the above example but instead of requiring an understanding of variables, constants, and functions it will now require an understanding of variables, constants and procedures (in order to understand the concept of recursion). Students who use this new learning object with the annotation attached, have two “paths” to obtaining the concept of recursion.

From a “real life” perspective, this could be viewed as Amy relating the learning object to an alternative background (in some way showing that functions are analogous to procedures and the annotation allows students with this alternative background to comprehend the chapter).

Student Bob annotates a video about data structures. His annotation (incorrectly claiming a B+ tree is a B tree written in C++) has the effect of adjusting the probability of an existing relationship. This annotation makes it 10% less likely that students seeing the learning object with Bob's annotation will attain the new concept compared to student who experience the learning object without Bob's annotation. Bob has confused students and prevented them from properly understanding what the video is trying to convey. The system should stigmatize this annotation and prevent it from being shown.

The system will run, and use the reputation and previous ratings to determine which annotations are shown to a student.

6 Related Work

Peer tutoring has been explored by a number of previous researchers. Some work, such as [10] and the COMTELLA project of [11], have investigated annotation techniques such as folksonomies and user tagging. While on the surface, this may seem similar to our work, there are important distinctions. With tagging, the purpose is to have users categorize items in ways that are meaningful for them, with the goal of sidestepping many of the problems inherent with ontologies (as articulated in [8]). In contrast, our approach endeavors to not just help students find an appropriate learning object, but to actually clarify that object and allow students to share insights with one another. Other works, such as [9][10], have been more explicit about arranging peer-tutoring. In their COPPER system, they arrange for students to practice conversations with one another, taking into account each student's level of proficiency, previous interactions and how they can best learn from one another. While our approach is a far less intense interaction than peer-tutoring that reasons about groups and gives them task in order to learn from

one another, our approach has the benefit of allowing asynchronous learning. Students may be able to benefit from annotations left by students who are no longer even in the course.

Other work [11][12] has been done considering text produced by learners, specifically the notes they take. They used these notes and text retrieval techniques to implicitly derive information about the student and to build a profile and social network about them. In contrast, we take an intensely pragmatic view of annotations and don't try to decipher the meaning. Instead, our approach reasons directly about which annotation will help a student learn, and ignores the actually underlying content of the annotations.

iHelp [2] is a project related to COMTELLA that involves reasoning about matching stakeholders (such as students, markers, tutorial assistants and instructors) in order to get the right information to the right person (both in public and private discussions). In [3] the authors extend iHelp to explore the value of tools such as chat rooms where learners are automatically drawn when using learning objects, shared workspaces where multiple learners can edit the same source code while discussing it and visualization tools for indicating a particular student's degree of interaction with her classmates. All of this done to encourage "learner collaboration in and around the artefacts of learning". In contrast, our work seeks to provide repositories of useful information from past students, rather than provide tools to assist in the interactions between current students. In many cases these "past students" may be a classmate who used the learning object the day before, while in others it might be a former student who has since graduated and left the school.

The work of John Lee et al. on the *Vicarious Learner* project [7], investigates how to automatically identify worthwhile dialogs to show to subsequent students, by determining the "critical thinking ratio" of a dialog, generated using a content analysis mark-up scheme. This ratio is determined from the positive and negative aspects within a discussion, with the assumption that discourse patterns provide signs of deeper levels of processing by learners and lead to a "community of enquiry" which benefits students. Dialogs with higher ratios could then be considered as valuable to show to new students. Our work differs from theirs in that we are interested in messages that have been explicitly left for future students and tied to a particular part of the course, rather than data-mining past interactions between students. Additionally, our approach is able to leverage similarities between students, in order to have a user-specific process for deciding which annotation should be shown. It may be interesting to integrate Lee et al.'s automated analysis of the critical thinking of text, as a component of deciding whether an annotation should be shown to a student.

6.1 Incentives

A criticism may be leveled that students won't be interested in helping their classmates (or admitting their ignorance) and therefore won't leave annotations. In the case that the intrinsic benefit aren't enticing to a group of students, various approaches, such as [5][6] could be used to encourage participation. The authors' feeling is that such explicit systems for coercing participation will not be needed in many learning contexts. Learning could be considered an inherently social process that naturally develops as sharing information with other student. Intrinsically this can be thought of as developing

social capital by sharing information, leading to greater trust and respect within the community.

7 Conclusion and Next Steps

We have demonstrated in this work the foundation for an approach for allowing students to explicitly share insights from their educational experiences to similar students going through the same process. Our next step will be to validate this approach, using simulated students following our approach, compared with both random and ideal provision of learning objects to students. Following this, we intend to corroborate our results with a study on real students.

There are various directions as well for extending our current model for reasoning about the annotations to be shown to students. Currently, which annotation is shown to a student is sensitive to the reputation of the annotation and the similarity of the current student to those who have rated this particular annotation. For future work, it is worthwhile to be modeling more extensively the student providing the annotation, assigning a greater weight to those annotations provided by students with greater learning proficiency. It may also be useful to be examining the level of achievement of the student who provided the annotation, considering as more valuable those students who are at a higher level of learning.

In our previous work [4], we presented an algorithm to determine which learning object to present to a student, based not only on their similarity to previous students but also on the extent to which those students benefited in their learning, when using that object. The methods proposed in that model for capturing the gains in learning of students could form a useful starting point for determining how to incorporate the learning proficiency of students into our algorithms for determining which annotations to show.

Another direction for future work is to examine alternative formulae for managing the votes for and against (beyond our current proposal to employ an arctan function). One possibility would be to examine alternative methods for converting the interval to a $[0,1]$ range. Another direction would be to examine in more detail the statistical confidence between the votes that are being registered, as a method of determining the importance of the votes towards the decision of showing a particular annotation. In addition, we are currently troubleshooting the proposed arctan function, as it is desirable to better incorporate the annotator's reputation and still maintain the desired $[0,1]$ range.

It is also worthwhile to be exploring more sophisticated metrics for determining the similarity between two students. Current research in collaborative filtering approaches for recommender systems has suggested more sophisticated techniques for making a recommendation, such as: statistical collaborative filtering, cluster models, and Bayesian networks[1]. We intend to examine whether these techniques may be applicable to our approach to annotations.

Finally, it would be useful to consider our proposal for presenting annotations to learning objects together with other algorithms for determining which learning objects should be presented, as part of the overall tutoring of that student. Our own investigations into curriculum sequencing [4] and peer-based development of the corpus of learning objects [5] would be particularly relevant, here.

References

- [1] J. S. Breese, D. Heckerman and C. Kadie. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, 1998, p. 43-52. San Francisco
- [2] H. Bretzke, and J. Vassileva, Motivating Cooperation in Peer to Peer Networks, *User Modeling*, 2003, pp. 218-227.
- [3] C. Brooks, R. Panesar, and J. Greer. Awareness and Collaboration in the iHelp Courses Content Management System. *1st European Conference on Technology Enhanced Learning*, 2006. Crete, Greece.
- [4] Champaign, J., and Cohen, R. A Model for Content Sequencing in Intelligent Tutoring Systems Based on the Ecological Approach and Its Validation Through Simulated Students. *Proceedings of the 23rd International FLAIRS Conference*, 2010. Daytona Beach, Florida (to appear).
- [5] Champaign, J., and Cohen, R. A Distillation Approach to Refining Learning Objects. *The Third International Conference on Educational Data Mining*, 2010. Pittsburgh, Pennsylvania (to appear).
- [6] Cheng, R., and Vassileva, J. Design and evaluation of an adaptive incentive mechanism for sustained educational online communities. *User Modeling and User-Adapted Interaction*, 2006, 16(3-4), p. 321-348.
- [7] Lee, J., Dineen, F., and McKendree, J. 1998. Supporting Student Discussions: it isn't Just Talk. *Education and Information Technologies*, 1998, 3, 3-4, p. 217-229.
- [8] McCalla, G. The Ecological Approach to the Design of E-Learning Environments: Purpose-based capture and use of information about learners. *Journal of Interactive Media in Education: Special Issue on the Educational Semantic Web*, 2004, 7, p. 1–23.
- [9] Rambo, N. and Beahler, C. Knowledge-Based Information and Systems. In *Public Health Informatics and Information Systems*, 2003, p. 352-375.
- [10] Read, T., Barros, B., Bárcena, E., and Pancorbo, J. Coalescing individual and collaborative learning to model user linguistic competences. *User Modeling and User-Adapted Interaction*, 2006, 16(3-4), p. 349-376.
- [11] Vassileva, J. Toward Social Learning Environments. *IEEE Transactions on Learning Technologies*, 2008, 1(4), p. 199–214.
- [12] Wan, X. Jamaliding, Q., and Okamoto, T. Discovering Social Network to Improve Recommender System for Group Learning Support. *Computational Intelligence and Software Engineering*, 2009, Wuhan, China.

Using Educational Data Mining Methods to Study the Impact of Virtual Classroom in E-Learning

Mohammad Hassan Falakmasir, Jafar Habibi
falakmasir@ce.sharif.edu, jhabibi@sharif.edu

Computer Engineering Department, Sharif University of Technology

Abstract. In the past few years, Iranian universities have embarked to use e-learning tools and technologies to extend and improve their educational services. After a few years of conducting e-learning programs a debate took place within the executives and managers of the e-learning institutes concerning which activities are of the most influence on the learning progress of online students. This research is aimed to investigate the impact of a number of e-learning activities on the students' learning development. The results show that participation in virtual classroom sessions has the most substantial impact on the students' final grades. This paper presents the process of applying data mining methods to the web usage records of students' activities in a virtual learning environment. The main idea is to rank the learning activities based on their importance in order to improve students' performance by focusing on the most important ones.

1 Introduction

During the past decades, because of the significant benefits it brings for all participants, the use of information and communication technologies in the educational domain has become widespread all around the world. Particularly in Iran, according to the effective role of education in the national development plan, these types of training drew more attention from the major and prestigious universities. Thus, they began to set up e-learning departments one after another. Due to the shortcomings and deficiencies of the e-learning platforms in the early years, there were lots of unresolved problems which affecting both students and teachers performance. First of all, lack of collaboration and communication facilities caused students to feel lonely and unsupported. In addition, their educational tendency to instructor-led learning caused them face new challenges understanding self-paced learning materials. These problems inspired e-learning departments to use web conferencing and virtual collaboration tools to satisfy the students' demands. This paper presents the major findings that resulted from studying the e-learning activities and their impacts on students' final grades. The structure of the paper is organized as follows: Section 2 outlines a literature review and the related works. Section 3 presents a background of the e-learning platform and the students' learning activities. Section 4 describes the methodology used to conduct the study and analyses the results using a decision tree. Finally, conclusions and future works are presented in Section 5.

2 Related Work

Sometimes the term *Virtual Classroom* is referred as the whole e-learning process or all the teacher-student interactions. However, in this paper the term is used for online synchronous virtual meetings which are conducted by the participation of the teacher and students using audio and video conferencing technologies. There is a limited amount of

research concentrating on the impact of these technologies upon the learning effectiveness. In contrast, there are a significant numbers of studies which have examined the role of other activities in the process of learning using data mining methods. In what follows a review on the most prominent studies is presented.

Bower and Richards [3] studied the impact of virtual classroom laboratories in computer science education. The main purpose of this research was to study the pedagogical aspects of these technologies; as a result they showed that such virtual laboratories are helpful particularly in the field of computer science. Also, another research by Redfern and Naughton [11] discusses and approves the positive role of video conferencing technologies in online education. This study particularly concentrates on the creation of brainstorming style-discussions and small group meeting which are fundamental to many of modern educational techniques.

From another perspective, some researchers have used web usage mining to look into students' activities in online learning environments. One of the leading studies conducted by Zaiiane and Lou [18], employed these methods to improve the features of an e-learning environment. In [16], web usage data of students is used to cluster them based on 'Expectation Maximization' (EM) algorithm. Each cluster represents a group of students with similar behavior. Moreover, the results are used to give the students suitable advices according to their group. The EM algorithm is also used in [15] to extract similar behavioral pattern of students in a collaborative unstructured e-learning environment.

Minaei-Bidgoli and Punch [8], used genetic algorithms and a combination of multiple classifiers to predict students' final grades. In [5] and [9], several machine learning and classification techniques were applied in order to predict the students' final score; the relevance of each feature is also assessed. This work was extended in [4] and Artificial Neural Networks were used to predict students' final grades. Beck and Mostow [2], had a different approach toward studying the students' performance data. They used a method called 'Learning Decomposition' to evaluate students' success ratio based on the amount of pedagogical support they received. The most important point in all the studies is that predicting students' final score based on their online activities is the leading approach to examine the effectiveness of e-learning.

3 The Platform

The E-Learning Department of Iran University of Science and Technology (IUST) started its services in the spring semester of 2004 with about 700 students and is currently serving about 1,800 students in two Bachelors' and three Masters' programs. The instructional plan in this department is designed in a way that the learning materials are mainly developed in the form of multimedia courseware which can be accessed by students in a weekly manner. In addition, the teacher can add supplementary resources to the learning content and evaluate the process of learning by providing the students with assignments and online quizzes. Having gained proper perception about the course concepts, the students participate in a virtual classroom session so as to discuss the lessons with the teacher and other students. The teacher can also present complementary information and gain feedback about the students' learning progress.

After the three years of using commercial products as virtual learning environment, the E-Learning Department of IUST started to build its own e-learning platform based on Moodle, a free open-source Learning Management System (LMS). Moodle is designed to support the learning style of *Social Constructivism*, in which the process of learning is performed by a set of interactions between students, teacher, and learning materials [12]. This style is not mandatory in Moodle but is what it supports best. There are several kinds of activities which students can perform in a course such as: viewing courseware, uploading assignments, posting messages in forums, writing messages to teachers and other students, etc. The system keeps detailed information about the way students interact with the system which have inspired lots of researchers to use these data to apply knowledge discovery and data mining methods to extract useful information about the students learning behavior [7,13,14].

Although there are several activities such as messaging, forums, and text chat to support collaboration of teachers and students, a sophisticated synchronous collaboration tools in Moodle is still missing. Consequently, the e-learning center of IUST added a new module to its Moodle to conduct virtual classrooms. The module was developed based on Adobe Flash Platform [1] considering its attractive interface and low bandwidth requirements, making it suitable for students connecting from small towns in different parts of the country. For each course there are 16 two-hour online sessions in a semester which are conducted on the specified time every week. During the session, different levels of interaction such as using video, audio, document sharing, whiteboard, and text chat can be used depending on the requirements of the lesson. For example, the teachers can share a power point slide or simply use a virtual whiteboard to present the content as well as broadcasting their own voice and video. Students primarily use text chat to interact with the teacher and ask questions. It is also possible for teachers to permit the students send their voices. To support the “any-time, any-where” promise of e-learning, all the sessions are recorded and archived for the students who cannot participate online sessions. The students who attend the class can also review the parts of lesson they didn’t follow or understand. In fact, these recorded sessions can be used as a permanent learning resource and the students can review them as many times they want.

4 Methodology Design

4.1 Main Idea

Although Moodle presents several reports on the students’ activities, they are not flexible enough to satisfy the instructors’ needs for observing their interactions with the system [6]. Additionally, there is no way for educational technologists and training managers to indicate the value of each activity in success of students. As it was mentioned before, this research aims to rank online learning activities based on their impact on the students’ final grades. For this reason, some variables have been defined as key performance indicators (KPIs) of students. Then the impact of each variable has been evaluated based on its influence on the score of students in the final exams. Particularly, data mining techniques have been employed to analyze the web usage logs of the virtual learning environment to infer some rules about the importance of each activity in the performance of students.

4.2 Participants

The current study have been conducted on the web usage logs of the system in Fall Semester 2008, when about 1,300 students were enrolled in almost 100 courses. However, the research is limited to 824 students in 11 courses; the instructors used most of the learning activities; and, the final grades of students were also available. The total population of students under investigation for this research is larger than similar studies. In addition, the students were completely remote from the university and had to learn most of the concepts and practices just by using the system via the Internet. In previous studies [8,13,14] the e-learning platform was used to facilitate teacher-student interactions and the online activities of students were not assumed as the most essential part of the learning process.

4.3 Procedure

The general process of educational data mining consists of four steps: *Collecting Data*, *Preprocessing*, *Applying data mining*, and *Interpreting the results* [13]. Here, a similar process has been used which follows slightly different methods in data collection and preprocessing steps. The two steps are integrated into a single extended stage of building a data warehouse from the activity logs of students. This approach makes it possible to monitor and study the learning behavior of the students and its relevant trends more in depth. The use of Data Warehouse and On-Line Analytical Processing (OLAP) tools in e-learning is gaining popularity among educational institutes and virtual universities [19]. In this section the whole process of applying data mining methods on the students' usage information is described.

4.3.1 Building the Data Warehouse

As it was mentioned before, Moodle keeps detailed records of students' activities. The teacher has access to summarized reports about students such as the date of their first and last logins, and the number of pages visited by them. The information about each learning activity is also available according to the categories specified by the system, not by the professor. Consequently, we designed a model and built a data warehouse to monitor the students' activities in precise detail. The activities are classified into nine categories: *resource view*, *virtual classroom participation*, *archive view*, *assignment view*, *assignment upload*, *forum read*, *forum post*, *discussion read*, and *discussion post*. In addition, according to our interviews with instructional technologists and training managers of IUST, a list of data elements and analytical dimensions along with the students' KIPs have been defined. Then, corresponding information was extracted from the Moodle database to answer their questions. Anyhow, the details of the dimensional modeling are beyond the scope of this paper.

For this study an information model is being used which gathers the information about the identified business requirements in the form of a summary table. Each column of the table represents a dimension important according to our objectives. Table 1 shows the design of the summarized table. A brief description of each dimension is also included. The structure is quite similar to the one which was used in [13] but it contains some other

attributes based on the KPIs extracted. To promote the level of interpretation and to facilitate the comprehensibility, the grades are stored in a discrete format. There are four categories of grades: A, if the value is equal or above 16.6; B, if the value is between 13.3 and 16.6; C, if the value is between 10 and 13.3; and F, if the value is less than 10.

Table 1: Summarization table of students activities

Name	Description
UserName	Name of User
CourseName	Name of the Course
ResourceView	Number of Courseware and Other Supporting Materials Views
VirtualClassroom	Number of Virtual Classroom Participations
ArchiveView	Number of Archive Views
ForumRead	Number of Forum Reads
ForumPost	Number of Forum Posts
DiscussionRead	Number of Discussion Reads
DiscussionPost	Number of Discussion Responses
AssignmentView	Number of Assignments Views
AssignmentUpload	Number of Assignment Answer Uploads
FinalGrade	Final Grade

4.3.2 Applying Data Mining Methods

The main group of data mining algorithms used in this study is ‘Feature Selection’. These methods, also known as ‘Attribute Evaluation’ algorithms, try to select the most relevant features according to a target concept. Several feature ranking and attribute selection methods have been proposed in the machine learning literature which use different metrics to discard irrelevant features and select the important ones including: information gain, gain ratio, symmetrical uncertainty, relief-F, one-R, and chi-squared. Each metric has its own bias. For example, the information gain measure is biased toward attributes with many values. Here, we use *gain ratio* [10] as the main evaluation metric since there are various number of records in the table regarding to each activity. The results of ranking based on the other methods are also presented and can be compared.

In this project, the data mining software package used to rank the attributes is Weka [17]. The reasons are that it is a free open-source application which implements several methods for attribute evaluation. Table 2 presents the results obtained from applying *gain ratio* attribute evaluation method on the summarized table of students’ activities. As the table shows the **virtual classroom participation** plays the most prominent role in this ranking while the second place belongs to the **archive views**.

Table 2: The results of ranking activities based on gain ratio metric

Attribute	Gain Ratio
Virtual Classroom	0.0839
Archive View	0.0694
Forum Read	0.052
Assignment View	0.0517
Assignment Upload	0.0497
Discussion Read	0.0364
Resource View	0.0324
Forum Post	0
Discussion Post	0

To confirm the results obtained from this evaluation some other methods are applied on the dataset and the attributes are ranked using other metrics. The results are outlined in Table 3. It can be perceived from the table that virtual classroom participation is the most influential feature affecting students' final grades among all other attribute evaluation methods.

Table 3: The The results of ranking activities based on other methods

Attribute	χ^2	Info-Gain	Symmetric Uncertainty	One-R	Relief-F	SVM
Virtual Classroom	1	1	1	2	2	2
Archive View	3	3	2	1	3	7
Forum Read	2	2	3	4	7	1
Assignment View	7	7	6	5	4	6
Assignment Upload	4	4	4	3	5	5
Discussion Read	5	5	5	7	6	4
Resource View	6	6	7	8	1	9
Forum Post	8	8	8	9	9	8
Discussion Post	9	9	9	6	8	3

4.3.3 Interpreting the Results

To illustrate and explain the results obtained from the research, a decision tree was created based on the C4.5 algorithm [10]. This algorithm uses the *gain ratio* metric to select the attributes and to build the tree. Figure 1 shows the first two levels of the tree. As depicted, the number of virtual classroom participation comes in the first level separating the students into two groups. Students who have participated fewer than 11 virtual classroom sessions, will probably (with the probability of about %55) fail in their exam. In contrast, Students with more than 11 participations might (with the probability of about %42) pass the exam with a C.

In addition, each node of the tree can be used to extract a rule to predict students' final grades based on their activities. For example, as highlighted in the figure, students with more than 11 virtual classroom participation and 17 archive views would get an A in the final exam. The coverage of this rule is about %25 and the accuracy is almost %41. These rules may help the teachers to identify the most important activities to focus on in order to improve their teaching style. The rules can also be employed by training managers and executives to provide with helpful information in resource planning and decision making.

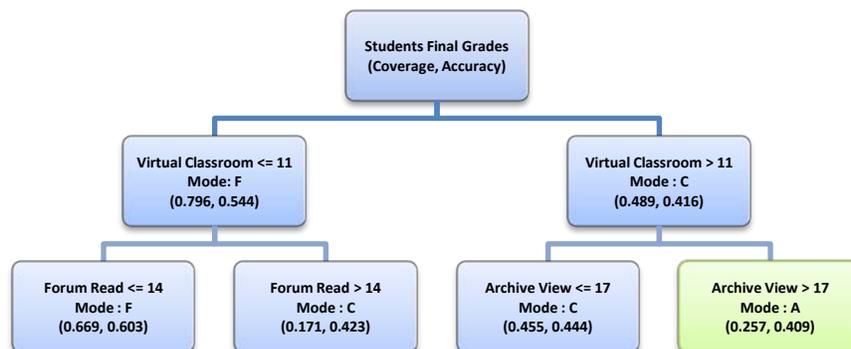


Figure 1: The first two levels of the decision tree model generated to predict students' final grades

5 Conclusions and Future Work

In this work we described the process of applying data mining methods in order to rank the students activities based on their impact on the performance of students in final exams. We used a number of 'Feature Selection' and 'Attribute Evaluation' methods together with real usage data of students picked up from the Moodle LMS in order to perform the case study. The results indicated that participation in virtual classroom sessions has the greatest impact on the effectiveness of learning in the particular settings of the IUST e-learning center. As a result, this fact motivated the managers and instructors to pay more attention to virtual classrooms and encourage the students to participate in these sessions. In the future, the effect of virtual classroom will be studied more profoundly considering some variables other than just the number of participation and archive views. It is also possible to analyze the students' behavior in virtual classrooms more deeply considering the activities performed by students. Finally, the teachers' instructional model in the virtual classroom will be studied in order to find the best methods that fulfill students' demands which might have a great impact on their learning performance.

Acknowledgement

The authors gratefully acknowledge the managers, executives, instructors, and officials of the E-Learning Department of Iran University of Science and Technology for their valuable suggestions and comments.

References

- [1] (2010) Adobe Flash Platform. [Online]. www.adobe.com/flashplatform/
- [2] J. E. Beck and J. Mostow, "How who should practice: Using learning decomposition to evaluate the efficacy of different types of practice for different types of student," in *the 9th International Conference on Intelligent Tutoring Systems*, 2008, pp. 353-362.
- [3] M. Bower and D. Richards, "The Impact of Virtual Classroom Laboratories in Computer Science Education," in *Thirty-Sixth SIGCSE Technical Symposium of Computer Science Education*, St. Louis, Missouri, USA, 2005, pp. 292-296.
- [4] A. T. Etchells, A. Nebot, A. Vellido, P. J. Lisboa, and F. Mugica, "Learning What is Important: Feature Selection and Rule Extraction in a Virtual Course," in *The 14th European Symposium on Artificial Neural Networks, ESANN*, Bruges, Belgium, 2006, pp. 401-406.
- [5] S. B. Kotsiantis, C. J. Pierrakeas, and P. E. Pintelas, "Predicting Students' Performance in Distance Learning Using Machine Learning Techniques," *Applied Artificial Intelligence*, vol. 18, no. 5, pp. 411-426, 2004.
- [6] R. Mazza and V. Dimitrova, "CourseVis: A graphical student monitoring tool for supporting instructors in web-based distance courses," *International Journal of Human-Computer Studies*, vol. 65, no. 2, pp. 125-139, 2007.
- [7] A. Merceron and Kalina Yacef, "Mining student data captured from a web-based

- tutoring tool: Initial exploration and results," *Journal of Interactive Learning Research*, vol. 15, no. 4, pp. 319–346, 2004.
- [8] B. Minaei-Bidgoli and B. Punch, "Using Genetic Algorithms for Data Mining Optimization in an Educational Web-based System," *Genetic and Evolutionary Computation*, vol. 2, pp. 2252–2263, 2003.
- [9] A. Nebot, F. Castro, A. Vellido, and F. Mugica, "Identification of Fuzzy Models to Predict Students Performance in an e-Learning Environment," in *The Fifth IASTED International Conference on Web-Based Education*, Puerto Vallarta, Mexico, 2006, pp. 74–79.
- [10] J. R. Quinlan, *C4.5: Programs for Machine Learning*.: Morgan Kaufmann, 1993.
- [11] Sam Redfern and Neil Naughton, "Collaborative Virtual Environments to Support Communication and Community in Internet-Based Distance Education," *Journal of Information Technology Education*, vol. 1, no. 3, pp. 201-211, 2002.
- [12] W. H. Rice, *Moodle e-learning course development. A complete guide to successful learning using Moodle*.: Packt Publishing, 2006.
- [13] C. Romero, S. Ventura, and E. Garcia, "Data mining in course management systems: Moodle case study and tutorial," *Computers & Education* , vol. 51, no. 1, pp. 368–384, 2008.
- [14] C. Romero, S. Ventura, P. G. Spejo, and C. Hervás, "Data Mining Algorithms to Classify Students," in *the 1st International Conference on Educational Data Mining*, Montral, Canada, 2008, pp. 8-17.
- [15] L. Talavera and E. Gaudioso, "Mining Student Data to Characterize Similar Behavior Groups in Unstructured Collaboration Spaces," in *Workshop in Artificial Intelligence in Computer Supported Collaborative Learning in conjunction with 16th European Conference on Artificial Intelligence, ECAI'2003.*, Valencia, Spain, 2004, pp. 17–22.
- [16] C. Teng, C. Lin, S. Cheng, and J. Heh, "Analyzing User Behavior Distribution on e-Learning Platform with Techniques of Clustering," in *Society for Information Technology and Teacher Education International Conference*, 2004, pp. 3052–3058.
- [17] (2010) Weka. [Online]. <http://www.cs.waikato.ac.nz/~ml/weka/>
- [18] O. Zaïane and J. Luo, "Web usage mining for a better web-based learning environment," in *the conference on advanced technology for education*, Banff, Alberta, 2001, pp. 60-64.
- [19] M. E. Zorilla, "Data Warehouse Technology for E-Learning," in *Methodologies and Supporting Technologies for Data Analysis*. Berlin, Heidelberg: Springer-Verlog, 2009, pp. 1-20.

Mining Students' Interaction Data from a System that Support Learning by Reflection

Rajibussalim^{1,2}

raji0050@uni.sydney.edu.au

¹Physics Department, Syiah Kuala University

²School of Information Technology, the University of Sydney

Abstract. In this paper we utilising some popular educational data mining (EDM) methods to explore and mine educational data resulted from a system that supports reflection for learning called Reflect. Our objective is to study whether using the Reflect helps students learn better in the subject. First, we explore data by means of statistical analyses. Then, we used clustering and classification techniques to identify learning behaviours associated with positive or negative outcomes. The results suggest that there is a positive correlation between reflection and student performance characterised by the level of activities carried out in the Reflect and the exam mark. We claim that our approach has resulted in the identification of some behaviour that leads to the increase in the students' performances.

1 Introduction

Literature suggested that reflection defined as “*individuals engage to explore their experiences in order to lead to a new understanding and appreciation*” [1] improves learning. Learning by reflection is considered important since one's learning effectiveness can increase when learners adapt to their learning experiences by reflecting on their learning process and the state of their knowledge [1, 2, 3]. The same researchers also suggested that self-assessment helps learners to develop the ability to identify their strengths and weaknesses and focus their study to the area that is need for improvement. However, most of the existing work that suggested this theory was not utilising education data mining (EDM) methods to draw their conclusion. In this paper, we explore the use of EDM methods on data gathered from an online learning system that support learning by reflection called Reflect (more discussion on Reflect presented in Section 2). In particular our goal was to evaluate the effectiveness of EDM methods for: (1) extracting knowledge about the impact of reflection on learning, (2) gaining knowledge about students' learning behaviour, and (3) identifying which behavioural patterns lead to positive or negative outcomes. This knowledge can be useful for teachers to better understand their students' learning behaviour and to inform students if their current behaviour was associated with negative outcomes in the past. The information can also be benefited if presented to students in which they can reflect on how effective their learning habits are against their current performances.

In section 2, we describe Reflect and how it supports student's self-assessment, followed by section 3 that outlines the subjects and data. Section 4 presents data pre-processing, the main data exploration, application of two educational data mining methods: clustering and classification techniques, we present the results and discussion. Section 5 summarises our conclusion.

2 Reflect

The Reflect system is a web-based educational learning system used for learning, practise and testing one's programming knowledge. This system is not intended as a sole mean for learning a topic but it serves as an additional learning medium to face to face teaching. The Reflect system was used by students as an alternative source of learning and practices of the topic Software Construction I (SOFT2130/2830) at School of IT, the University of Sydney. Reflect can offers a great flexibility in learning experience. It offers extra attributes that differ and not commonly found in other LMSs or ITSs such as when students assess an example solution, they can view the tutor's assessment that are comparable against the students' assessment. It will show a discrepancy table if the student's assessment differ from the tutor's ones. Students may also view their progress in a graph showing how close they are getting to the tutor's assessment.

The Reflect system is unique because it aims to promote learner reflection and student self-assessment through scrutable learner model that shows the learner a model of their progress [4]. Kay et al. [4] suggested that there are two key elements of self-assessment: (1) identification of criteria and standards to be applied to their own work and (2) evaluation of their work compared to these criteria and standards. In Reflect, these two key elements of self-assessment were carried out by students when they performed the following steps in order to solve the problems:

1. Students read a task and assess example solutions made available in the system.
2. Students provide their own solutions to solve the problem and
3. They self-assess those solutions compared to the criteria teachers defined for the task.

By performing these steps, the students had practised to self-assess themselves and their state of knowledge of a particular task. Thus, they carried out learning by reflecting to their learning experiences.

3 Subject and Data

3.1 Subjects

The subjects used in this research are the students who undertook the Software Construction I (SOFT2130/2830) course at School of Information Technology, the University of Sydney in the second semester of 2007. There were 175 students enrolled for the course, however only 156 (89.1%) students managed to completed it. Out of 156 students who completed the course, 109 (69.9%) students passed and 47 (30.1%) of them failed the course.

3.2 Data

The data used in this research are coming from two sources: (1) students' activities data gathered from Reflect system and (2) students assessment data gained from the lecture. In experiments both of data were used. Student activity data are gathered from Reflect database in the form of XML format. The data are organised in a hierarchical structure i.e. each task consists of a task name and a number of learning objectives. The numbers

of learning objectives for each task are varying from one to five learning objectives. Moreover, a learning objective may appear in one or more task. The student assessment data are spread over several worksheets of Microsoft Excel format. These include lecture quiz marks, homework and labs marks (including practical exam marks on week 4, 8 and 12), quizzes marks, and the final exam mark of students who enrolled for Software Construction I topic for the year of 2007. These data are obtained from the lecturer who taught and organised the course.

4 Process of EDM: Results and Discussion

4.1 Data pre-processing

Data pre-processing tasks consist of cleaning the data from incomplete or inconsistency data and errors and also integrating and transforming the data to an appropriate format for mining.

Cleaning the Data

After data extracted from the Reflect database in XML format, the next step is to clean the data from inconsistencies and errors before they are ready to be mined. We did the process of data cleansing manually by searching for incomplete data or errors and removed them from database. The errors may come from unintended users who did not enrol for the topic. These users include Reflect system administrator who used the system for testing and other students who did not enrol for the course but used the system for learning and practising their C programming skills. The error may also come from a user who had a double user logins. This could happen when a student changed their user login at some points during the semester.

Integrating and Transforming the Data

Data integration is a process of integrating or merging the data from multiple tables or databases into a coherent data store [5]. Educational data mining processes often involve retrieving and analysing multiple data attributes that spread across several tables or databases. The data, therefore, needs to be summarised into a new summarisation table consisted all necessary attributes for mining. As mentioned earlier in Section 3.2, our data sources are coming from: the students' activity data gathered from the Reflect database and students assessment spreadsheet obtained from the course coordinator. To perform the data mining out of these data, we are required to merge necessary data attributes into a summary table. Table 1 shows the data attributes of this summary table.

Data transformation may involve a number of techniques including smoothing i.e. to remove noise from the data; aggregation i.e. summary operation applied to the data; generalisation where the low level data are replaced by higher-level concepts and normalisation where the attributes data are scaled thus they fall within a small range i.e. 0.0 to 1.0. [5]. In regards to the Reflect data, we used a smoothing technique to remove a noise from a student with an excessive self assessment input. We also used aggregation and normalisation techniques to smooth our data. We used aggregation to summarise the

weekly Homework/Lab marks and quiz marks into Total Homework/Lab and Total quizzes marks, while we used normalisation to normalise the attribute values of Total Lecture quizzes and Lecture quiz attendances. Finally, we transformed our data into ARFF and csv for mining with Weka.

Table 1. List of data attributes in summary table

Attribute Name	Description
L2_mem_diagram	Week 2 lecture quiz mark (numeric)
L5_pointers	Week 5 lecture quiz mark (numeric)
L8_scope	Week 8 lecture quiz mark (numeric)
Tot_Lec_qz	Total lecture quiz marks (%)
Lec_qz_attd	Total lecture quiz attendances (%)
Tot_HWL	Total homework/lab marks (%)
HWL_attd	Total homework/lab attendances (%)
Tot_qz	Total quiz marks (%)
qz_attd	Total quiz attendances (%)
Exam	Final exam mark (numeric)
exam_cate	Categorical final exam mark (ordinal)
n_task	Number of tasks done in the Reflect (numeric)
n_lo	Number of learning objectives done in the Reflect (numeric)

Constructing Dataset for Experiments

From the list of data attributes showed in Table 1, we constructed two sets of data: (1) a data set that consist only numerical data and percentages, and (2) a data set that consist of numerical data, percentages and categorical final exam mark. The first data set is used for statistical and clustering analyses while the second data set is mainly used for classification analyses.

4.2 Data Exploration

The statistical analyses are often providing a starting point for data analyses. Therefore we carried out a number of correlation analyses to study the relationships and to measure if one data attribute is significantly correlated to another. In addition, a statistical analysis can be used to determine which variable is best explain the differences between two or more groups, that is a variable that can distinguish one group from another. Our objective is to utilise statistical analyses to find the relationships and correlation analyses between: (1) lecture quizzes score and final exam mark, and (2) lecture quizzes attendances and final exam mark.

Discussion

Statistical analyses suggested that although there were positive correlations for both attending (Lec_qz_attd) and performing quizzes (Tot_Lec_qz) set by the teachers in the lectures and the final exam mark, these correlation are not considered high enough. The

correlation score (r) between Lec_qz_attd and exam is only 0.462 ($p < 0.01$) and correlation (r) between Tot_Lec_qz and exam mark is 0.413 ($p < 0.01$).

Meanwhile, another analyses suggested that correlation score between Total Homework/Lab (Tot_HWL) and exam is reasonably high ($r = 0.618$; $p < 0.01$). However, the correlation between Homework/Lab attendances (HWL_attd) and exam is not high enough ($r = 0.354$; $p < 0.01$).

The results indicated that Homework/Lab performance is considered significantly important toward achieving a good mark in the final exam. However, this is not necessarily the case for lecture quiz and its attendances as their correlation with the exam mark are not significantly high. Nevertheless, the results showed positive correlations that indicated the importance of attending both and performing well in both lecture quizzes and Homework/Lab activities.

4.3 Clustering Students

Clustering is an unsupervised classification used for grouping objects into classes of similar objects [6]. A number of researchers have implemented clustering techniques for mining e-learning data with various purposes such as for finding groups of students who have similar learning characteristics, to encourage group-based collaborative learning and to provide incremental learner diagnosis [7].

Our work utilised a K-means clustering algorithms to cluster group of students based on their similar behaviour in using the Reflect system. We choose K-means clustering algorithm because it is considered as one of the most popular and mostly used clustering algorithm in broad data mining research community [8]. Another reason for choosing K-means is because it is available and ready to use in Weka system.

Lecture quizzes versus final exam score

We utilised both numerical data and a combination of numerical and nominal data set. Before running the K-means algorithm, the data set are transformed into comma separated value (csv) format. The reason is because, a csv file format is easy to use and it is one of file format acceptable in the Weka system. Here, our objectives are first, to distinguish stronger group of students from the weaker ones and then to study learning characteristics that differentiate each group. To achieve this objective, we utilised numerical data set with categorical exam mark.

Discussion

Clustering analyses are able to provide more detailed information about students beyond what simple statistical analyses may offer. Based on the clustering results, we are able to categorise students into several clusters based on their performances and attendances in the lecture quizzes. Each group is characterised by how often their attended the lecture quizzes and their performance on those quizzes.

Table 2. Student cluster using K-means (k=4)

Cluster size and (percentages)	Lecture quizzes Mark	Lecture quizzes attendance	Exam Mark*	Cluster label
52 students (33%)	Average High mark of all lecture quizzes	Very high % of lecture quizzes attendances (99%)	High Moderate (highest among other clusters) exam	Strong group
14 students (9%)	Average Low mark of the lecture quizzes	High % of lecture quizzes attendances	Moderate exam mark	Moderate group
47 students (30%)	Average Low mark of the lecture quizzes	Medium % of lecture quizzes attendances	Low exam mark	Lower Moderate group
43 students (28%)	Lecture quiz marks not available	Very low % of lecture quizzes attendances (15%).	Very Low (almost failed) exam mark	Weak group

*Exam mark: fail < 40, Low >=40 and <55; Moderate >=55 and <70; Good >=70 and < 80; Very Good >=80.

The results summarised in the Table 2, suggested that 33% of students who achieved higher exam mark had at least average high mark on lecture quizzes and its attendances. In the other hand, 28% students who achieved very unsatisfactory (very low) exam results had not attended enough lecture quizzes (very low attendances).

These results highlighted the important of every lecture quiz toward the increasing of students' understanding to the topic. In other words, attending and performing tasks and exercises in the lecture quizzes may have a direct impact on student's knowledge of the topic and may affect their performance in the exam. This is maybe because attending and completing tasks and exercises in the "exam-like" environment, such as in a lecture quiz, during the lectures may make the student familiar and well prepared with the type of question being asked in the final exam. Meanwhile, the students with poor class attendances would miss the opportunity to familiar themselves to the type of question that may appear in the exam. This result supports our hypothesis that "*students who attended and have good marks on all lecture quizzes (week 2, 5 and 8) performed well in the final exam*".

4.4 Classifications

Classification is a supervised classification in which the labels of pre-classified patterns are identified. This pre-classified pattern is known as training data set. Within the training data set there is a class attribute that will be used to label a newly encountered, still unlabelled pattern [HanK06_Data_mining]. Our approach is to use a well known C4.5 (J48) algorithm from Weka data mining tools. The J48 algorithms would generate decision trees that might be used to extract classification rules. In our case, the objective is to classify students into different groups or branches with almost equals final exam mark. The model resulted from the experiments can be used to predict the final exam

mark of new students. . The J48 decision tree algorithm required that the label in this case is exam mark, must be in the categorical or nominal form. We, therefore, grouped the exam mark into the following category: Fail if the exam mark is <40; Low if the exam mark ≥ 40 and <55; Moderate if the exam mark ≥ 55 and <70; Good if the exam mark ≥ 70 and <80; and Very Good if the exam mark ≥ 80 . This categorisation is different to the categorisation of the final grade students received at the end of semester. The final grade which is determined by the course coordinator at the end of the semester integrated not only the final exam mark but also other assessment components including quizzes marks, homework/lab marks and assignments scores. The categorisation for the final exam mark is fixed by the course coordinator and agreed by students at the beginning of the course. Some rules were generated from classification analyses. These rules are summarised as follow:

```

IF (Tot_qz <= 0.67) AND (n_task <= 6) THEN exam = Fail
IF (Tot_qz <= 0.67) AND (n_task > 6) AND (HWL_attd <= 0.8) AND
  (Lec_qz_attd <= 0.33 ) THEN exam = Low
IF (Tot_qz <= 0.67) AND (n_task > 6) AND (HWL_attd <= 0.8) AND
  (Lec_qz_attd > 0.33 ) THEN exam = Moderate
IF (Tot_qz between 0.67 and 0.96) AND (n_lo <= 12) THEN exam = Low
IF (Tot_qz between 0.67 and 0.96) AND (n_lo > 12) THEN exam =
  Moderate.
IF (Tot_qz >= 0.96) THEN exam = Good

```

Discussion

The results suggested that the total quizzes mark (Tot_qz) was the main discriminator of the final marks followed by the number of task students done in the Reflect system. The rules generated by J48 algorithm revealed the characteristics of each group of students. For example, a student should at least complete six tasks or more in the Reflect system and achieved more that 67% of total quizzes mark to reduce the risk of fail in the final exam. Meanwhile a student who can achieved 96% of the total quizzes marks are directly classified as Good, meaning he/she will be among students who are likely to achieve a good mark in their final exam. The other groups of students are classified based on some other activities including homework/lab (HWL_attd) and lecture quiz attendances (Lec_qz_attd).

As we can see from the results, there are set of critical separation point for each class, for example 67% or 96% of total quizzes, 6 or 12 tasks done in Reflect, 33% of lecture quiz attendances (Lec_qz_attd) and 80% of homework/lab attendances (HWL_attd). These separation points are set automatically by J48 classifier.

Using J48 algorithm, our first experiment did not produce a good result. The accuracy level recorded for the first experiment is only 51.28%. This means that out of 156 data, only 80 instances were correctly classified into their classes. Other 76 instances were incorrectly labelled into the other classes.

In the second experiments, the accuracy level increased although not much. The accuracy level for the classifier model became 56.41%. This means that out of 156 instances, 88 of

them are correctly classified into their classes while remaining of 68 instances were still not correctly classify or labelled into their classes.

This information would be very useful for a course coordinator for example the classifier can be used to predict the final exam marks of new students or to promote some types of activities to obtain higher marks.

5 Conclusions

We have demonstrated how EDM methods have been utilised to extract some valuable information from the Reflect data. We have discussed how the users of Reflect can perform self-assessment by following certain procedures as described in the section 2. These include to rate their understanding of each learning objectives related to the task, providing answers to the tasks, self assess their answers and sample solution against certain criteria defined by the teacher and comparing the discrepancies between their answers to the teacher's assessment.

A number of issues have emerged during the study either related to the data or the interpretation of the results. First, the present study only used one semester data, hence the analyses is limited. For the further work, it would be more interesting if data mining is conducted for data that have been collected for many years. Secondly, the current data used was not designed in the first place to be suitable for mining. Therefore, the data are complex, contain noise, and heterogeneous.

The results of EDM analyses suggested that the Reflect system helps users to self assess themselves and thus satisfied the aims indentified in the introduction of section that is students learnt more by using system that support learning by reflection such as the Reflect system.

References

- [1] Boud, D., and Keogh, R. Promoting reflection in learning: A model, Reflection: turning experience into learning, 1985, pp. 18-40.
- [2] Schon, D. A. The reflective practitioner, New York, 1083.
- [3] Schon, D.A. Educating the reflective practitioner, San Francisco, 1987.
- [4] Kay, J., Li, L., and Fekete, A. Learner Reflection in Student Self-assessment, *Proceedings of ACE 2007, 9th Australasian Computing Education Conference*, Australian Computer Society, Ballarat, Victoria, 2007, pp. 89-95.
- [5] Han, J., and Kamber, M. Data Mining Concept and Techniques, Morgan Kaufman, San Francisco, 2006.
- [6] Witten, I.H., and Frank, E. Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, Morgan Kaufmann, 1999.
- [7] Tang, T., and McCalla, G. Smart Recommendation for an Evolving e-Learning System: Architecture and Experiment. *International Journal on e-Learning* 4, 2005. pp. 105-129.
- [8] Romero, C., Ventura, S., and Garcia, E., Data mining in course management systems: Moodle case study and tutorial. *Computers & Education in Press*, Corrected Proof 2007.

Process Mining to Support Students' Collaborative Writing

Vilaythong Southavilay¹, Kalina Yacef¹ and Rafael A. Calvo²

¹{vstoto, kalina}@it.usyd.edu.au

²rafa@ee.usyd.edu.au

¹School of Information Technologies, University of Sydney

²School of Electrical and Information Engineering, University of Sydney

Abstract. Writing, particularly collaborative writing is a commonly needed skill. Investigating how ideas and concepts are developed during the process of writing can be used to improve not only the quality of the written documents but more importantly the writing skills of those involved. In this paper, process mining is used to analyze the process that groups of writers follow, and how the process correlates to the quality and semantic features of the final product. Particularly, we developed heuristics to extract the semantic nature of text changes during writing. These semantic changes were then used to identify writing activities in writing processes. We conducted a pilot study using documents collected from groups of undergraduate students writing collaboratively in order to evaluate the proposed heuristics and illustrate the applicability of process mining techniques in analyzing the process of writing.

1. Introduction

Nowadays, Computer-Supported Collaborative Learning, particularly Collaborative Writing (CW) is widely used in education. Students often use computers to take notes during lectures and write essays for their assignments. Thanks to the availability of the Internet, students increasingly write collaboratively by sharing their documents in a number of ways. This has led to increased research on how to support students' writing on computers.

Our research motivation is to investigate ways to support these collaborative writing activities by providing feedback to students during the collaborative writing process. We have built a prototype which, based on the text written by students, can automatically perform analysis on text changes in order to discover types of semantic changes and identify writing stages.

In order to develop a tool supporting CW, understanding how ideas and concepts are developed during writing process is essential. The process of writing consists of steps of writing activities. These steps of writing activities can be considered as the sequence patterns comprising of both time events and the semantics of changes made during those steps. Therefore, process mining, which focuses on extracting process-related knowledge from event logs recorded by an information system, can be used to extract sequence patterns of writing activities that lead to quality outcomes.

In this paper, a process mining technique is used to detect semantic changes in writing activities in order to gain insight on how student write their documents. Our work used a taxonomy of writing activities, proposed by Lowry et al. [4]. In addition, we used a model developed by Boiarsky [2] for analyzing semantic changes in the writing process.

Although process mining techniques have been successfully applied to extract process-related knowledge from event logs recorded by information systems [3] for two decades, the techniques have only recently applied to educational data. For example, in Pecheniskiy et al [8], the authors utilized process mining tools to analyze data from online multiple choice examinations and demonstrated the use of process discovery and analysis techniques. They were interested in individual students' activities of answering online multiple-choice questions during assessment, not in activities where students write and edit texts collaboratively. In addition, there has been abundant research for improving the support of quality writing in education such as automatic scoring of essays [10], visualization [6], and document clustering [1]. However, these approaches, unlike ours, focus on the final product, not on the writing process itself.

The remainder of the paper is organized as follows. In Section 2, the framework for supporting CW is presented. The heuristics for extracting semantic changes and writing activities during writing process are proposed in Section 3. A pilot study mining student writing processes is discussed in Section 4. Finally, Section 5 concludes and outlines and our future work planned in this area.

2. WriteProc

The framework, WriteProc for exploring collaborative writing processes was introduced in [12]. It integrates a front-end writing tool, Google Docs, which not only supports collaborative writing activities, but also stores all revisions of documents created, shared, and written by groups of authors. WriteProc uses two process and text mining tools, ProM [9] and TML [13]. ProM provides a way to extract knowledge about writers' activities and collaboration. In [12], process mining was used to extract only patterns of students' interaction and collaboration for peer review. In this paper, the process analysis is used for identifying sequence patterns of writing activities that lead to a positive outcome and indicate patterns that may lead to problems. TML analyzes text changes between individual revisions of documents. This analysis can provide semantic meaning of changes in order to gain insight into how writers develop ideas and concepts during their writing process. Full details about WriteProc can be found in [12].

In our pilot study, WriteProc retrieves some revisions from Google Docs of all the documents written by the students, using Google Document Data API. Google Docs automatically saves documents every few seconds. Authors can also deliberately save their documents. In Google Docs, every saving command (either auto or committed) produces one revision for each document. WriteProc does not retrieve all revisions of the documents, as many of them do not contain any changes. For a particular document, it first obtains the revision history (log) containing information, such as revision number, timestamp of auto and committed saving, and author id. The revision history is then analyzed to identify writing sessions for individual writers. From this analysis, for each session WriteProc downloads a revision after each minute of writing. These downloaded revisions are then pre-processed to extract and index paragraphs and sentences for further analysis. Using predefined heuristics (defined in the next section), WriteProc performs a text-based analysis on the indexed revisions of each document which extracts semantic meaning of text changes and identifies writing activities. Sequences of writing activities

are then analyzed using process mining. In the next section, we propose the heuristics and explain how they will be used to detect writing activities.

Table 1. Heuristics for determining collaborative writing activities: Brainstorming (B), Outlining (O), Drafting (D), Revising (R), and Editing (E) based on text change operation (C1 – C8), text structure (S1 – S2), and functions (F1 – F3).

Abbreviation: An operation is allowed (Y) or not allowed (N), List (L), Structured List (SL), Sections and Paragraphs (Sec. & Par.), Number of Sentences (S), Number of Paragraphs (P), Changing/Fluctuating (F), Constant(C), Improving (I), and Not applicable (N/A).

Writing activities	Surface change	Reorganization of information	Consolidation of information	Distribution of information	Addition of information	Deletion of information	Alteration of form (Macro-structure change)	Micro-structure change	Structure	Number of Sentence vs Number of Paragraphs	Ratio of Number of Words	Topic Overlap	Cohesion Measure
	C1	C2	C3	C4	C5	C6	C7	C8	S1	S2	F1	F2	F3
B	Y	Y	Y	Y	Y	Y	Y	Y	L	$S \approx P$	F	N/A	N/A
O	Y	Y	Y	Y	Y	Y	Y	Y	SL	$S \approx P$	F	N/A	N/A
D	N	N	N	N	Y	Y	N**	Y	Sec. & Par.	$S > P$	F	F	F
R	N	Y	Y	Y	N*	N*	Y	N*	Sec. & Par.	$S > P$	F	F	I/C
E	Y	N	N	N	N	N	N	N	Sec. & Par.	$S > P$	C	C	C

3. Heuristics for determining collaborative writing activities

Writing activities in collaborative writing can be categorized into 6 common activities: brainstorming, outlining, drafting, reviewing, revising, and editing. The definition of these activities is described in [4]. It is important to note that in general these six activities do not occur in a linear sequence. In a document writing process, we consider reviewing activities made not only by the writers (owners) of the document, but also by instructors or editors or peers who read and annotate the document for content, grammar, and style improvements. In this work, we concentrate on automatically identifying the five collaborative writing activities: brainstorming (B), outlining (O), drafting (D), revising (R), and editing (E).

In order to identify the five collaborative writing activities in the writing process of a particular document, basic heuristics are proposed. Particularly, our heuristics are based on text changes, text structures, topic changes and cohesion improvement in the document from one revision to another. The heuristics utilized in our analysis are presented in Table 1. Each writing activity can be identified using text change operations (C1 to C8), text structures (S1 and S2), and functions (F1 to F3), which are explained below.

3.1 Text structures

The writing activities can be determined by the structure of the written texts (S1) and the number of sentences and paragraphs (S2). During brainstorming, authors normally write in bullet-point lists consisting of single words or phrasal words (compound nouns). Consequently, the number of paragraphs (the number of lines) is approximately equal to the number of sentences (the number of words or items). Although during an outlining phase the number of paragraphs and sentences are still the same, the text structure is more organized into sections and subsections. When writers start drafting their documents, number of sentences and paragraphs change dramatically. During this phase, the number of sentences is expected to be higher than the number of paragraphs. This is also truth for revising and editing phases.

3.2 Text change operations

Eight types of text change operations (C1 – C8) were used in our heuristics. These text change operations were based on the revision change functions proposed by Boiarsky [2]. Writers use the text change operations in their writing activities for different purposes in order to produce final pieces of writing. We developed basic assumptions for our pilot study as following:

- During brainstorming, writers can reorder, adding, or deleting items of lists of brainstorming ideas. They can also format the lists, alter the whole items of the lists, or change some items. Similarly, during outlining, writers can add, delete, reorder, format, and change some or the whole sections of their organized list.
- During drafting, revising and editing, text change operations become more complicated. Drafting activities start when the structure of the written text changes from bullet-point or structured lists to paragraphs. In other words, alteration of form (C7) usually indicates the start of drafting activities (as depicted by N** in the table). During drafting, information is added and removed all the time. Therefore, expansion of information (C5), deletion of information (C6), and micro-structure change (C8) imply drafting activities.
- However, if C5, C6 and C8 happen after reviewing activities, we consider them as revising activities (as noted by N* in the table). Particularly, peer review was incorporated in our pilot study. We assumed that after getting feedback from their peers, students may add, delete, and alter texts in their documents. Also students may completely erase the whole written text and rewrite the text from scratch after getting feedback from their peer review. This operation is C7.
- In addition, common revising activities are reordering (C2), consolidating (C3), and distributing texts (C4). These changes occur after writers start drafting and reoccur many times in the writing process. Our assumption is that during drafting writers may stop writing and revise their own written texts in order to improve the cohesion and effectively convey information and ideas to readers.
- For simplicity, all surface change operations (C1) including formatting, spelling and punctuation corrections are consider to be editing activities. Similar to C2 -

C4, editing activities are considered to be common and reoccur many times in the writing process.

3.3 Number of words

The ratios between the number of words of two consecutive revisions are computed (F1). The ratio was used in conjunction with topic overlap and cohesion measurement discussed below to determined writing activities.

3.4 Topic overlap

Our heuristics also used a topic overlap measurement (F2). We analyze the change in topics (concepts) for two consecutive revisions of documents. Our intuition is that when people write about something, they usually repeat the subject (topics) to keep readers' attention. By identifying concepts and comparing them between two consecutive revisions of pieces of writing, we gain information on how writers develop their idea and concept during writing process. Intuitively during drafting and revising, topics overlap (F2) changes dramatically. However, during editing F2 should be constant. The method for computing the topic overlap (F2) is described in Section 4.1.

3.5 Cohesion Changes

Another measurement used in our heuristics to detect writing activities is the cohesion of the text. We measure cohesion of each individual revision of documents. Particularly, we calculate the distance between consecutive sentences and paragraphs in the written texts in order to gain an insight on how paragraphs and the whole texts have been developed. Our assumption is that during a drafting phase, the cohesion of the written text fluctuates a lot. After authors revise the text, the cohesion of the text is usually improved. There should be no change in the cohesion of the written text during editing phase.

We use the Latent Semantic Analysis (LSA) technique to measure the cohesion of the text. In particular, for each revision of documents we compute average sentence and paragraph similarities using LSA for single documents as described in [14] and compare the result with the former revision of the same documents in order to determine if there is an improvement in cohesion for these two revisions.

4. Pilot study

As a way of evaluating the proposed heuristics and illustrating how process mining can be used to analyze writing activities, we conducted a pilot study to investigate writing processes of students in the course of E-business Analysis and Design, conducted during the first semester of 2009 at the University of Sydney. In this course, students were organized in groups of two and asked to write Project Specification Documents (PSD) of between 1,500 and 2,000 words (equivalent to 4-5 pages) for their e-business projects. They were required to write their PSD on Google Docs and share them with the course instructor. The course also used peer review in which each PSD was reviewed by other two students who were members of different groups. After getting feedback from their

peers, students could revise and improve their documents if necessary before submitting the final version one week later.

In addition, the marks of the final submissions of the PSD (as presented in [12]) together with a very good understanding of the quality of each group through the semester was used to correlate behaviour patterns to quality outcomes. In particular, to be able to give insight into how students wrote their own documents, we performed a process diagnostic to give a broad overview of students' writing activities.

4.1 Pre-processing

In this section, we describe the document pre-processing method used in our study. We analyzed 21 documents in this study. As discussed in Subsection 3.4, LSA was used for measuring the changes in cohesion in the written text. The pre-processing step for LSA involved the extraction of the terms from all concerned revisions of the documents. First, each revision of documents was split into paragraphs using a simple matching to the newline character. Then, each paragraph was divided into sentences using Sun's standard Java text library. After that, each sentence, paragraph and the whole text were indexed using Apache's Lucene, which performed the tokenization, stemming, and stop word removal. We used Porter's stemmer algorithm (Snowball analyzer integrated in Lucene) for stemming words. Next for each revision, a term-document matrix was created. Term frequency (TF) and document frequency (DF) were selected as weight terms. We discarded terms that only appear once in each revision of documents. The space of term-document matrix was reduced using Singular Value Decomposition (SVD). We adopted the method of Villalon et al. [14] to reduce the dimension of the LSA space to 75% of the total number of sentences.

In order to compute the topic overlap discussed in Subsection 3.3, we first extracted topics from each revision of documents. Our approach in extracting topics from each revision of documents was based on Lingo clustering algorithm developed by Osinski et al. [7]. Especially, we extracted frequent phrases from each revision (we use the assumption and definition of frequent phrase discussed in detail in [7]). Next, by using the reduced term-document matrix calculated for LSA above, for each revision we discovered any existing latent structure of diverse topics in a particular revision. After discovering topics of each revision of documents, we compared topics of two consecutive revisions to calculate the topic overlap between the two revisions. As a baseline measure, we selected a simplistic word overlap measures. This measure was used for sentence-level similarity such as in the work of [5].

The final step in our data preparation was to create a writing activity log of all documents in the format used by a process mining tool like ProM. First, for each revision (except the first revision) we compare it to the former revision and obtain the types of text change operations between the two using a file comparison utility like *diff* tool. Then, we applied the proposed heuristics using the obtained types of text changes, the results of LSA cohesion and topic overlap calculated above. In conjunction with timestamp and user identification obtained from the revision history discussed in Section 2, we can create an event log of writing activities, in which process analysis can be performed.

4.2 Writing process analysis

After preprocessing, the resulting event log consisted of 8,233 events in total. Each process case represented one document. The mean number of events per document was 392, with a minimum of 77 events per document and a maximum of 705 events per document. There were 6 different types of events corresponding to 6 types of writing activities. We performed process analysis in order to gain insight into how students wrote their documents.

The Dotted Chart Analysis utility of ProM [9] was used to analyze students' writing activities. The dotted chart was similar to a Gantt chart [11], showing the spread of events over time by plotting a dot for each event in the log. Figure 1 illustrated the output of the dotted chart analysis of students writing their PSD documents. All instances (one per document) were sorted by start time. In the figure, points represented writing activities occurring at certain time. There were six writing activities as mentioned above. They were represented with different colors and shapes: white circles denoted brainstorming, brown circles were outlining, black triangles represented drafting, black squares depicted reviewing, brown squares were revising, and white diamonds denoted editing activities.

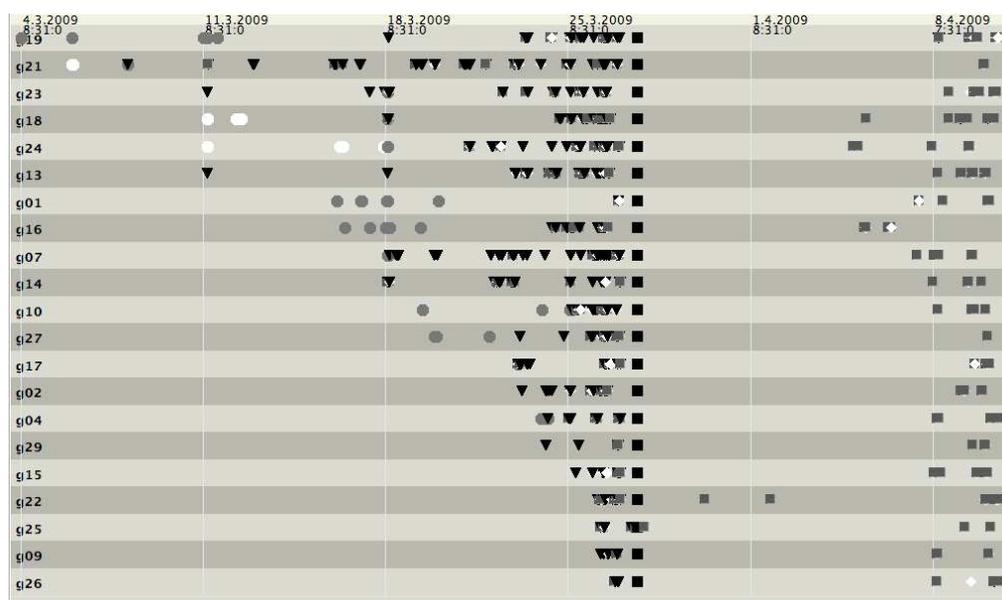


Figure 1. Dotted chart of 21 groups of students writing collaboratively (from ProM tool). White circles denoted brainstorming, brown circles were outlining, black triangles were drafting, black squares were reviewing, brown squares were revising, and white diamonds were editing activities.

From the figure, we observed that most students started their writing approximately two weeks before the due date for the submission of peer review (27th March 2009). Exceptionally, there existed 6 groups starting their writing activities quite early. Group 19 (received the final mark of 9/10) and 21(10)¹ started their outlining and brainstorming activities very early in the first week of the semester. Group 13(9), 18(9), 23(8), and 24(8) also commenced their writing tasks early in the second week of the semester.

¹ Group X(Y) denotes the group number X receives the final mark of Y out of 10.

Therefore, we noticed that students who performed brainstorming and outlining started their writing quite early and received quite good marks. In addition, as we expected, there were many writing activities during a week before the due date for peer review submission. Interestingly, there were 5 groups commencing their writing tasks few days before the due date. They were Group 9 (10), 15 (9), 22 (9), 25 (9), and 26 (9). These groups also received quite high marks for their writing, although they started writing quite late. Actually students of these groups did not use Google docs to perform outlining and brainstorming. They also started their writing using other word processing applications such as MS Word because all of them commenced their writing on Google Docs with substantial amount of texts (containing sections and paragraphs). During the one-week of peer review, we expected to have no writing activities since students were waiting for the feedback from peer review. However, Group 22, who just started their writing on Google Docs, performed some activities during this time. We checked the revisions of the document of Group 22 and found that there were substantial text changes performed by these activities. At the end they received a good mark of 9/10. Furthermore, after getting feedback from their peer review (3rd April 2009), students started revising and editing their documents before the final submission (10th April 2009). We observed that Group 16(9), 18(9), and 24(7) started working on their documents soon after getting feedback. They were among top groups in the class.

In addition, we were naturally interested in finding out more about writing activities of each group and the path each group was following in the process of writing. ProM provides a Performance Sequence Analysis (PSA) plug-in to find the most frequent paths in the event log [3]. Figure 2 illustrates a sequential diagram of students' writing activities in our pilot study. The patterns were ordered by the number of groups generating them. From the analysis above, we learned that not all groups of students performed their brainstorming and outlining before actually drafting their documents. The PSA also confirmed this. In addition, from the PSA we checked each individual group's activities in order to determine which groups did not conduct brainstorming and/or outlining for their writing tasks. From Figure 2, there were seven distinct patterns of activities. Pattern 0 and 5 indicated groups that started drafting without brainstorming and outlining. Pattern 0 was originated from 8 groups: 2, 9, 13, 14, 15, 22, 25, and 28. For Pattern 5, there was only one group: 26. Pattern 1, 3 and 4 involved all activities except brainstorming. There were 7 groups belonged to Pattern 1: 1, 4, 7, 10, 16, 17, and 27. For Pattern 3 and 4, each of them was generated by only one group namely 19 and 23, respectively. Clearly, more than half of the class did not conduct outlining and one third of the class did not bother performing brainstorming (at least on Google Docs) before drafting. We discussed this matter with the course instructor who was aware that most students performed their writing plans offline. Finally, there were 3 groups whose their writing activities generating Pattern 2 and 6: 18, 21 (for Pattern 2), and 27 (for Pattern 6). These groups planed their writing tasks with brainstorming and outlining. Consequently, all of them received high marks.

The process model of all 21 documents was discovered by using the Heuristic miner algorithm [15] with default threshold parameters (implemented in ProM). Figure 3 depicts a transition diagram of the model. The numbers in the boxes indicate the frequencies of the writing activities. The decimal numbers along the arcs show the probabilities of transitions between two activities and the natural numbers present the

number of times this order of activities occur. In addition, we also extracted the process model of each individual group in order to gain an insight on how the group conducting its writing.

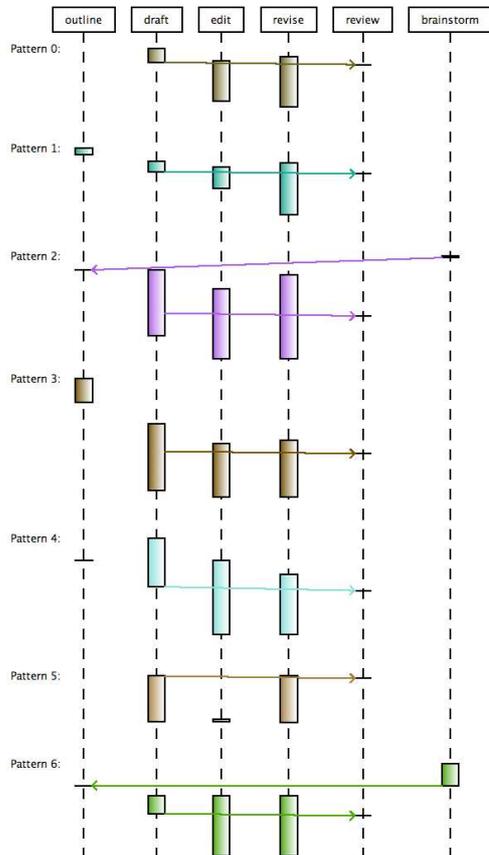


Figure 2. Sequence patterns of 21 groups of students writing collaboratively. The patterns are ordered by the number of groups generating them (from ProM tool).

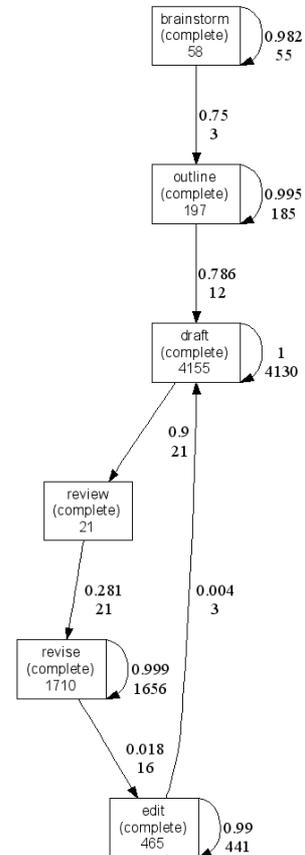


Figure 3. A transition diagram of the process model for 21 documents. The natural numbers refer to frequencies (of events and transitions), and the decimal numbers represent probability of the transitions.

5. Conclusion and future work

The work presented in this paper is a work in progress. The pilot study described in the previous section provides fundamental work for us to develop basic heuristics to extract semantic meaning on text changes and determine writing activities. Based on the heuristics, we were able to analyze student's writing activities using a process mining and discover 7 patterns on writing activities of 21 groups of students. However, correlated with final assessment, we could not distinguish clearly the better from the weaker groups.

This preliminary work gives us direction for the next step of our work. In the future, the discovered patterns, process snapshots provided by performance sequence and dotted chart analysis can be used for providing feedback to students during their writing so that they are aware of their writing activities and can coordinate effectively.

One way to improve our understanding of what writing processes lead to better outcome is to improve our heuristics. In our current work, the surface change operation indicates only changes in spelling, number, punctuation, and format. We did not include grammatical correction in the current work yet. In addition, one of text change operations proposed by Boiarsky is the improvement in vocabulary [2]. We did not detect the improvement in vocabulary in our current analysis. These two text change operations will be cooperated in the heuristic in the future. In addition, we already measure the change in topics (concepts) which represent word repetition. Although word repetition is common in writing, good writers usually utilize synonymy and pronouns to avoid annoying repetition. This issue was not considered in this paper and will be cooperated in the future work.

Acknowledgements

This work has been funded by Australian Research Council DP0986873. We would like to thank the many people involving in the development of ProM and TML.

References

- [1] Andrews, N. O. and Fox, E. A. *Recent Developments in Document Clustering*. Technical Report TR-07-35, Computer Science, Virginia Tech, 2007.
- [2] Boiarsky, C. Model for Analyzing Revision. *Journal of Advanced Composition*, 1984, 5, p. 65-78.
- [3] Bozkaya, M., Gabriel, J. and Werf, J. M. E. M. v. d. Process Diagnostics: A Method based on Process Mining. *International Conference on Information, Process, and Knowledge Management*, 2009.
- [4] Lowry, P. B., Curtis, A. and Lowry, M. R. Building a Taxonomy and Nomenclature of Collaborative Writing to Improve Interdisciplinary Research and Practice. *Journal of Business Communication*, 2003, 41, p. 66-99.
- [5] Metzler, D., Bernstein, Y., Croft, W. B., Moffat, A. and Zobel, J. Similarity Measure for Tracking Information Flow. *Proceeding of the 14th ACM International Conference on Information and Knowledge Management*, Bremen, Germany, 2005, p. 517 - 524.
- [6] O'Rourke, S. and Calvo, R. A. Semantic Visualisations for Academic Writing Support. *14th Conference on Artificial Intelligence in Education*, Brighton, UK, 2009, p. 173-180.
- [7] Osinski, S., Stefanowski, J. and Weiss, D. Lingo: Search Results Clustering Algorithm Based on Singular Value Decomposition. *Proceedings of the International IIS: Intelligent Information Processing and Web Mining Conference*, Zakopane, Poland 2004, p. 359--368.
- [8] Pechenizkiy, M., Trcka, N., Vasilyeva, E., Aalst, W. M. P. v. d. and Bra, P. D. Process Mining Online Assessment Data. *Educational Data Mining*, Cordoba, Spain, 2009.
- [9] ProM. <http://prom.win.tue.nl/tools/prom>, 2010.
- [10] Shermis, M. D. and Burstein, J. *Automated Essay Scoring: A Cross-disciplinary Perspective*. Volume 16, MIT Press 2003.
- [11] Song, M. and Aalst, W. M. P. v. d. Supporting Process Mining by Showing Events at a Glance. *7th Annual Workshop on Information Technologies and Systems*, 2007, p. 139-145.
- [12] Southavilay, V., Yacef, K. and Calvo, R. A. WriteProc: A Framework for Exploring Collaborative Writing Processes. *Australasian Document Computing Symposium*, Sydney, 2009.
- [13] TML. <http://tml-java.sourceforge.net>, 2010.
- [14] Villalon, J. and Calvo, R. A. Single Document Semantic Spaces. *The Australasian Data Mining conference*, Melbourne, 2009.
- [15] Weijters, A. J. M. M., Aalst, W. M. P. v. d. and Medeiros, A. K. A. d. Process mining with the heuristics miner-algorithm. *BETA Working Paper Series WP 166*, Eindhoven University of Technology, NL, 2006.

Automatic Rating of User-Generated Math Solutions

Turadg Aleahmad, Vincent Aleven, Robert Kraut

{[aleahmad](mailto:aleahmad@cs.cmu.edu), [aleven](mailto:aleven@cs.cmu.edu), [kraut](mailto:kraut@cs.cmu.edu)}@cs.cmu.edu

Human Computer Interaction Institute, Carnegie Mellon University

Abstract. Intelligent tutoring systems adapt to users' cognitive factors, but typically not to affective or conative factors. Crowd-sourcing may be a way to create materials that engage a wide range of users along these differences. We build on our earlier work in crowd-sourcing worked example solutions and offer a data mining method for automatically rating the crowd-sourced examples to determine which are worthy of presenting to students. We find that with 64 examples available, the trained model on average exceeded the agreement of human experts. This suggests the possibility for unvetted worked solutions to be automatically rated and classified for use in a learning context.

1 Introduction

Intelligent tutoring systems have made great progress on adapting instruction to match students' cognitive needs. They match instructional content to the learner's changing state of knowledge, the cognitive variables. There is little work, however, on matching instruction to the learner's interests, motivations and identity, the affective and conative variables [5], which can improve student engagement and test scores [4]. Personalizing to these other factors may require a large set of socially and topically varied problems and worked example solutions. Our previous work demonstrated that crowd-sourcing is a feasible approach to covering the gamut of learners with many worked examples [1]. However, it is important to determine which of the contributions are worthy of presenting to learners, which need more work, and which should be discarded. In the current paper we assess the feasibility of using data mining to automatically classify crowd-sourced worked example solutions by their readiness to present to learners.

Machine rating of quality has been studied in many domains (e.g. Wikipedia articles [2], student essays [3]). The strongest predictors are often simple, and sophisticated features add little. The 1960s Project Essay Grader using just word count, average word length, counts of punctuations and prepositions, etc. achieved 0.78 correlation with teachers' scores, almost as strong as the 0.85 correlation between two or more trained teachers [3].

Our data set comes from a corpus of worked examples to practice the Pythagorean Theorem. Each example consists of problem statement and a series of steps to solve it. The 278 examples were contributed by 10 math teachers, 20 non-math teachers and 110 amateurs. Each solution was coded manually by three experts on a 4 point scale: Useless, Fixable, Worthy, or Excellent. The ratings of the three coders (Cronbach's $\alpha=.71$) were averaged to create a single *solution quality* measure, for which the models were trained.

2 Results

Model accuracy is evaluated as the correlation of the machine prediction with the *solution quality* measure. Because a subsequent analysis on human correlations requires unseen instances, a test set was made holding out a stratified random sample of 28

instances. The remaining 250 worked examples were used to train a model evaluated by 10-fold cross-validation. The attribute space was restricted to features of the example (e.g. count of the word “solve”) and whether the contributor agreed to be further contacted. The learning algorithm selected was REPTree and was enhanced through a Bagging meta-classifier using Weka.

To see how the performance is affected by the number of instances, we trained on random subsets each of size 8, 16, 32, 64, 128 and 250. Twenty subsets were created with random sampling with replacement. Training on all 250 instances the mean accuracy (correlation with *solution quality* value) over the 20 runs was $r=.67$.

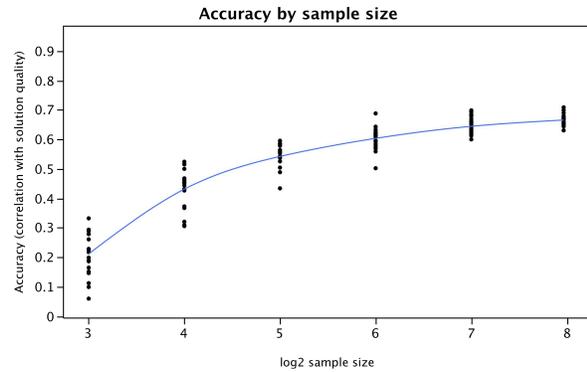


Figure 1 Accuracy by log2 sample size

For the examples in this data set, the machine-based rater correlated better with the human experts than they did with each other (*humans pair-wise average* $r=.53$ and *model average* $r=.67$). We found that 64 training examples were enough to beat the human raters on average and that with 128 examples the machine outperformed consistently. For domains in which these results hold therefore just 64 rated examples are needed to create a model that can automatically rate future contributions. Because it can do this instantly and on-demand, such models could be used to facilitate a peer-produced worked example system. Anyone could contribute a solution, and learners would only be shown those of the highest quality. This work supports the scalability and sustainability of such a system.

References

1. Aleahmad, T., Aleven, V., and Kraut, R. Creating a Corpus of Targeted Learning Resources with a Web-Based Open Authoring Tool. *IEEE Transactions on Learning Technologies* 2, 1 (2009), 3-9.
2. Dalip, D.H., Gonçalves, M.A., Cristo, M., and Calado, P. Automatic Quality Assessment of Content Created Collaboratively by Web Communities: A Case Study of Wikipedia. *Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries*, ACM (2009), 295-304.
3. Hearst, M.A. The debate on automated essay grading. *IEEE Intelligent Systems and Their Applications* 15, 5 (2000), 22–37.
4. Ku, H. and Sullivan, H. Student performance and attitudes using personalized mathematics instruction. *Educational Technology Research and Development* 50, 1 (2002), 21-34.
5. Martinez, M. and Bunderson, C.V. Building interactive World Wide Web (Web) learning environments to match and support individual learning differences. *J. Interact. Learn. Res.* 11, 2 (2000), 163-195.

Tracking Students' Inquiry Paths through Student Transition Analysis

Matt Bachmann¹, Janice Gobert^{1,2} and, Joseph Beck^{1,2}
{matt.bachmann, jgobert, joseph.beck}@wpi.edu

¹Computer Science Department,

²Social Sciences & Policy Studies Dept.
Worcester Polytechnic Institute

In the Science Assistentment project (http://users.wpi.edu/~sci_assistments/) we provide students with rich and dynamic opportunities to engage in inquiry while researchers and teachers use performance data as assessment indices of scientific inquiry skills and domain knowledge. Activities in the Science Assistentment Project follow a pedagogical model known as the inquiry cycle, namely, explore, hypothesize, experiment, and analyze. Figure 1 shows this cycle as a graph and the transitions that can be made between states. State “Comp” represents completion of the activity.

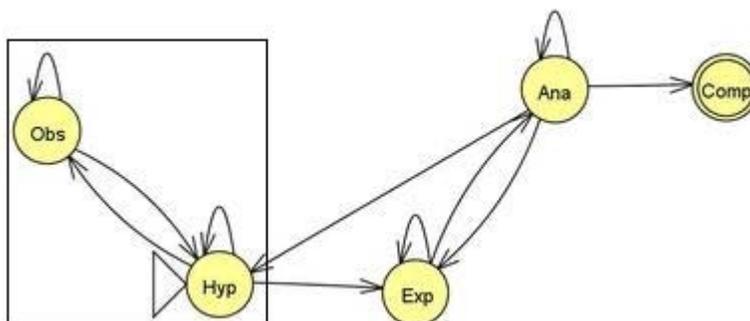


Figure 1. The Inquiry process as modeled by our activity.

Important to researchers and developers are ways to assess students' inquiry process within microworlds, as well as a rigorous method for providing scaffolding to students who struggle with inquiry during the activity. The method presented here utilizes a Markov chain to track a student's path through the various inquiry activities.

The data were collected from 148 eighth grade students, ranging in age from 12-14 years, from a public middle school in Central Massachusetts. Students engaged in inquiry using our learning environment including a microworld simulating the phase changes of water and a series of inquiry support tools for making a hypothesis, collecting data, and analyzing data.

Based on the logging features of these microworlds, we track the path students take through the activity by counting the transitions students make and create a Markov model of their inquiry process (Figure 1). This potentially affords researchers a way of grouping students based on their path through the activity. The rationale for this is that some paths are more effective representing systematic inquiry [1] and, in turn, lead to positive learning gains while other inquiry paths do not. Using this Markov model, researchers hope to eventually be able to develop scaffolds for students whose path corresponds to less effective inquiry. However, in order to do this we need have models that can diagnose a students' inquiry path. The current work seeks to develop such a student model.

The first attempt at breaking up different groups of students used the scores students obtained on the multiple choice pretest: those in the top third on the pre-test, and those in the bottom third. The middle third was not used for this analysis. Based on these two groups of students, we generated two transition models. To validate these models we computed the log likelihood of each model given each student. A paired samples t-test on the log likelihoods of the intended models (the model generated from the group which the student belonged) versus the unintended models (a model generated from students from a different group) found the models to be distinct ($t(98) = 3.385, p < .001$). Using a sign test (log likelihood of intended model - log likelihood of unintended model) for each student, we found that the intended model had a higher log likelihood 68% of the time. Although this disaggregation was somewhat successful, 68% accuracy in terms of classification is not a great improvement over a baseline of 50% generated by random guessing.

For our next attempt, rather than picking some measure and using that to disaggregate the models, we looked at the models themselves and attempted to disaggregate them by applying a K-means cluster analysis using Weka (version 3.6.2), which generated two models each of which was based on a cluster. The major difference between the two models is that students in model 1 tended to return to the observe state of the inquiry cycle (see highlighted portion of Figure 1).

The next steps going forward are to try to use our models along with other learner characteristics we have about our participants to predict what cluster the students fall into. We also need to find if the clusters fit into categories of high and low performance in the activity.

References

- [1] Buckley, B.C, Gobert, J.D., Horwitz, P. (2006) Using Log Files To Track Students' Model-based Inquiry. *Proceedings of the Seventh International Conference of the Learning Sciences (ICLS)*, Mahwah, NJ: Erlbaum, p.57-63.

DISCUSS: Enabling Detailed Characterization of Tutorial Interactions Through Dialogue Annotation

Lee Becker¹, Wayne H. Ward¹² and Sarel vanVuuren¹
{lee.becker, sarel.vanvuuren}@colorado.edu
{ward}@bltek.com

¹Computational Language and Educational Research Center, University of Colorado at Boulder
²Boulder Language Technologies

1 Introduction

We describe new work in using dialogue annotation for tutorial tactic discovery. When analyzing a tutorial dialogue session, surface features such as ratio of student to tutor speech, average delay between turns, and speech disfluency rates can serve as approximate measures for student participation, certainty, and affective state – factors critical to the success of an ITS. While these data are relatively easy to extract directly from a transcript and do provide insight into the nature of instruction, they are too coarse to give directed feedback to tutors and tutorial dialogue authors. Instead, an intermediate representation that summarizes the exchanges within a dialogue may be useful such as provided by a new annotation scheme for tutorial tactic representation called the Dialogue Schema for Unifying Speech and Semantics (DISCUSS), which we briefly describe here and detail in [1].

2 Background and Related Work

A variety of previous research has employed tutorial dialogue annotation for both general-purpose educational analyses as well as for more directed goals of optimizing ITS behavior. Graesser and Person [4] used tutorial dialogue annotation to determine the aspects of a dialogue that correlate statistically with learning gains. Boyer et al. [2] applied Hidden Markov Models to a corpus tagged with dialogue acts to discover patterns in tutorial dialogue strategies. Chi et al. [3] utilized smaller, more specific annotation to learn when a tutor should *elicit* rather than *tell* an answer to students. Collectively, these cases illustrate how dialogue level analysis requires richer representations than can be derived from transcripts alone. We are currently collecting a corpus of one-on-one ITS/WOZ led tutorial dialogues for inquiry-based instruction of FOSS science materials [5]. When complete, we will have nearly 1000 transcripts spanning 4 domains and 16 sub-domains of elementary school science. Preliminary observations from the WOZ transcripts suggest that the semantic entailments between student utterances and the goals for a lesson are the driving force behind tutor behavior.

3 The DISCUSS Annotation Scheme

Based on our analysis, we developed DISCUSS to reflect the relationships between learning goals and the dialogue. It does this in multiple layers, which capture the pragmatics, semantics, and structure of a conversation. Together, these layers yield a

comprehensive, yet domain-independent, description of the tutorial dialogue. Figure 1 illustrates how DISCUSS can be used to represent these relations.

Goal1: Batteries provide electricity Semantic Parse: [Agent].batteries, [Predicate].provide, [Theme].electricity.
S1: <i>It looks like the battery is giving electricity to the light bulb</i> Semantic Parse: [Agent].battery [Predicate].giving [Theme].electricity [Beneficiary].light bulb Answer/Describe/Process [Goal1]
T2: <i>Giving electricity. Interesting. Tell me more about that.</i> Mark/Null/Null [S1.Predicate, S1.Theme], Feedback/Positive/Null, Ask/Elaborate/Process [S1]

Figure 1: Sample snippet of DISCUSS annotation

4 Discussion

This annotation will allow extraction of more detailed features that can give better indication of conversation cohesiveness, student understanding, and tutor pedagogical style. The shallow semantics that DISCUSS captures will allow us to investigate how tutorial tactics vary and agree across domains and concept types as well as between tutors, and it will allow us to identify relations between student populations and instructional styles. Most importantly, this more detailed representation will allow us to map specific aspects of the dialogue onto the general patterns of tutorial dialogue strategy.

5 Acknowledgements

This work was supported by grants from the NSF (DRL-0733322, DRL-0733323) and the IES (R3053070434). Any findings, recommendations, or conclusions are those of the author and do not necessarily represent the views of NSF or IES.

6 References

- [1] L. Becker and W. H. Ward. Discuss - the dialogue schema for unifying speech and semantics. Technical report, University of Colorado at Boulder, 2010 - Forthcoming.
- [2] K. E. Boyer, E. Y. Ha, et al, Discovering tutorial dialogue strategies with hidden markov models. In *Proc. international conference on artificial intelligence in education (aied '09)*, 141-148, , 2009.
- [3] M. Chi, P. Jordan, K. Vanlehn, and D. Litman. To elicit or to tell: does it matter?. In *Proceeding of the 2009 conference on artificial intelligence in education*, 197--204, Amsterdam, The Netherlands, The Netherlands, 2009. , IOS Press.
- [4] A. C. Graesser and N. K. Person. Question asking during tutoring. *American Educational Research Journal*, 31(1):104-137, 1994.
- [5] W. H. Ward and S. Van Vuuren. Improving science learning through dialogs with virtual tutors . NSF Grant Proposal.

Data Mining of both Right and Wrong Answers from a Mathematics and a Science M/C Test given Collectively to 11,228 Students from India [1] in years 4, 6 and 8

James A. Bernauer, Ed.D. and Jay C. Powell, Ph.D.
bernauer@rmu.edu; jpowell@better-schooling-systems.org ;
SESS Robert Morris University, Better Schooling Systems

Abstract: Our poster presentation illustrates how to include wrong answers in test analyses using Response Spectrum Evaluation (RSE) procedures to track answer patterns on an answer-by-answer basis.

RSE is a statistical procedure adapted from the multinomial [3] that bypasses the linear dependency problem so that alternative (wrong) answers can be included in data-mining analyses. Thus, the study of the dynamics of learning events can be conducted on an answer-by-answer basis.

Previous investigations [3] using this procedure have revealed:

1. The selection of answers is the result of the way students interpret the test questions.
2. These interpretations are directly inferable from the answers selected (or presented).
3. Selection procedures involve a number of strategies that are characteristic of each student, providing diagnostic information that can inform teaching.
4. This information is of more value to teachers who focus upon teaching how to think and how to learn instead of reproducing course content.
5. Some students show systematic development similar to the sequence described by the clinical observations of Piaget [1], while others show deterioration in the reverse direction.
6. Some students systematically shift from the right answer on the easy questions to particular types of “wrong” answer when their ability breaks out of “all” or “nothing” thinking (without considering other options) into more intellectually flexible mind-sets. RSE is the only procedure with this detection capability.
7. The focus upon the right answers in the psychology of test-taking reinforces closed-minded thinking on the part of students taking the test, meaning that if the objective of teaching is profound understanding, the focus upon “right” answers is psychologically invalid.
8. The dynamics of learning revealed by the RSE procedures are non-linear and multichotomous, meaning that the use of total-correct scores to assess student

performance is mathematically invalid because the normal distribution requires dichotomous data to be utilized.

Our study considers two tests, mathematics and science, given to the same students at three school levels, years 4, 6 and 8. It presents three examples from each test, showing the patterns of answer selection transitions on these items and the interactions among them.

In our study, we draw implications for using RSE to add important diagnostic and interpretive information for teachers. This information can often be derived by direct analysis of each answer to each item. The ways in which these behaviors aggregate, however, requires determining the associations among answers with all answers in the test (both right and wrong), requiring that the analysis bypasses linear dependency.

We are grateful to Educational Initiatives Pvt. Ltd. (India) for sharing raw response data and questions from the ASSET test used in their study “Student Learning in Metros 2006.”

References

- [1] FLAVELL, J. H. (1963). *The Developmental Psychology of Jean Piaget*. New York, NY: Van Nostrand.
- [2] POWELL, J. C. (2010). Testing as feedback to inform teaching. Chapter 3 in *Learning and Instruction in the Digital Age: Making a Difference through Cognitive Approaches*. New York: Springer. (In press).
- [3] POWELL, J. C. and SHKLOV, N. (1992). Obtaining information about learners’ thinking strategies from wrong answers on multiple-choice tests. *Educational and Psychological Measurement*, 52, 847-865.

Mining information from tutor data to improve pedagogical content knowledge

Suchismita Srinivas¹, Muntaquim Bagadia¹, Anupriya Gupta¹
Educational Initiatives Pvt. Ltd., Ahmedabad, India.
(suchi@ei-india.com)

School-based intelligent tutoring systems present a unique, largely untapped, teacher-development opportunity. Research on teacher content knowledge has shown that effective teachers develop a specialised 'pedagogical content knowledge' for teaching mathematics. School-based tutors provide a rich record of the student learning process. This record can be mined for insights that contribute to teacher's 'pedagogical content knowledge'. By situating this knowledge in the context of an active teaching-learning process occurring in the school, these insights contribute to a continuous data-driven teacher development exercise.

1. Introduction

School-based intelligent tutoring systems are designed to assist teachers in meeting learning objectives by providing personalised learning opportunities to students. Such systems embody expert pedagogical knowledge as well as student models which are used to guide students through content so as to maximise learning. Mindspark[®], developed by Educational Initiatives, is a mathematics tutoring system being used as a part of the school curriculum by nearly 7000 students in more than 13 schools across India.

While most tutoring systems aim to provide comprehensive learning opportunities within the tutoring environment, a school-based tutoring system presents a unique, largely untapped teacher-development opportunity. Research on teacher content knowledge in the past 3 decades has brought to the table and refined the notion of 'pedagogical content knowledge' [1] and provided empirical measures of the subject knowledge that is required for teaching[2]. Insights mined from tutoring systems can be used to inform teachers about the nature of the learning process students go through in specific content domains and thus provide a source of continuous teacher development.

2. Data

The Mindspark[®] system consists of a sequence of specially designed learning units (clusters), which contain finely graded questions on concepts that make up the topic. Students learn through feedback and by going through specific remedial clusters, which help address specific problems. The questions are MCQs, 'fill-in-the-blank' type, multiple-select or interactive. Student response, the time taken in making the response, the time spent reading the feedback, the no. of repetitions of each learning unit and other data is recorded in a database.

3.1 Detecting misconceptions and common errors

The Mindspark system helps teachers identify specific misconceptions or learning gaps prevailing at the grade or school level. Mindspark consists of distractor-driven MCQs that are capable of trapping common errors.

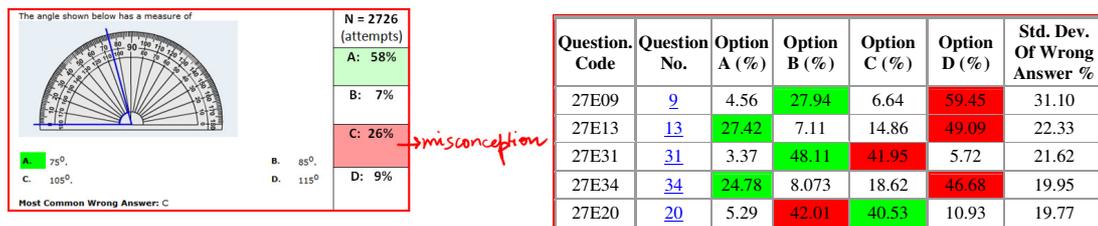


Figure 1 – Example of a common error/misconception highlighted and a table showing the use of standard deviation of wrong answer % as a sorting criterion. The correct answer is marked in green, common wrong answers in red/pink.

A measure that directly picks up questions where a large proportion of students have been drawn to a particular wrong answer (thus indicating a common error pattern) is the standard deviation of the % of students choosing the various wrong answers. A higher SD identifies clustering of response in particular options. Tutor data from across grades can then be used to inform teachers about how misconceptions evolve with age and ability.

3.2 Identifying ‘speed-breakers’ in learning

The Mindspark learning units contain very finely graded learning units based on conceptual progression as determined by curriculum and content experts.. However, students sometimes treat closely related question-types very differently and such a situation often represents a ‘kink’ in the curriculum and a mismatch between student understanding and teacher’s perception of how difficult different related tasks are. Such potential kinks in the curriculum represent specific points where student learning may be derailed, both in the tutor environment as well as in the classroom.

3.3 Identifying and gaining from difficult learning situations

Students using the Mindspark system occasionally get trapped in difficult learning situations. They make no progress and repeatedly fail to clear particular learning units despite feedback. By using correlation-based measures to classify student trajectories, we provide useful information to teachers on the student’s experience and make it possible for these situations to be fruitfully utilized to systematically improve teacher’s pedagogical content knowledge.

4. Conclusion

School-based intelligent tutoring systems provide a rich record of the student learning process. This record can be mined for insights that contribute to teacher’s ‘pedagogical content knowledge’. By situating this knowledge in the context of an active teaching-learning process occurring in the school, these insights contribute to a continuous data-driven teacher development exercise.

4. References

1. Shulman, L. S. Those who understand: Knowledge growth in teaching. *Educational Researcher*, 1986, 15(2), p. 4-14.
2. Hill, H., Rowan, B., Ball, D. Effects of Teachers’ Mathematical Knowledge for Teaching on Student Achievement. *American Educational Research Journal*, 2005, 42, p.371-406.

Clustering Student Learning Activity Data

Haiyun Bian

hbian@mscd.edu

Department of Mathematical & Computer Sciences

Metropolitan State College of Denver

Abstract. We show a variety of ways to cluster student activity datasets using different clustering and subspace clustering algorithms. Our results suggest that each algorithm has its own strength and weakness, and can be used to find clusters of different properties.

1 Background Introduction

Many education datasets are by nature high dimensional. Finding coherent and compact clusters becomes difficult for this type of high dimensional data. Subspace clustering was proposed as a solution to this problem [1]. Subspace clustering searches for compact clusters embedded within subsets of features, and it has proven its effectiveness in domains that have high dimensional datasets similar to educational data. In this paper, we will show that different clustering and subspace clustering algorithms produce clusters of different properties, and all these clusters help the instructor assess their course outcomes from various perspectives.

2 Clustering Student Activity Data

We assume that datasets are in the following format: each row represents one student record, and each column measures one activity that students participate in. Our test data contains 30 students with 16 activities, and 7 students failed this class. The final grade is the weighted average from the scores in all 16 activities.

2.1 Student clusters

Student clusters consist of groups of students who demonstrate similar learning curves throughout the whole course. These clusters are helpful to identify key activities that differentiate successful students from those who fail the course. We applied the SimpleKMeans from Weka [2] to the test dataset with k being set to 2. The results show that cluster1 contains 6 out of 7 students who failed the course, and cluster2 contains 24 students among whom 23 passed the course. One student who failed was clustered into cluster2, and we found out that this student's composite final score is 58%, which lies right on the boundary of passing/failing threshold.

2.2 Activity clusters

Here we focus on finding groups of activities in which all students demonstrate similar performance. For example, we may find a group of activities where all students show

worse than average performance. This suggests that the instructor may want to spend more time on these activities to cope with the difficulty. To find this type of clusters, we need to transpose the original data matrix. We applied SimpleKMeans to the transposed test dataset. We tried different k values and found out the best clustering results was obtained at k =4 by looking at the curve of within group variance as a function of k. Among these four clusters of activities, cluster4 is the most challenging group of activities because its cluster centroid is consistently lower than the other three clusters.

2.3 Subspace clusters

We only report the results from PROCLUS [3] algorithm due to limited space. PROCLUS needs to set the number of clusters (k) and the average subspace dimensionality (l).

SC_0: [0 0 0 0 0 0 1 0 0 0 1 0 0 0 0] #13 {2 5 7 8 10 13 14 15 17 21 23 27 29 }

SC_1: [0 0 1 0 0 0 0 1 1 1 0 0 0 0 0 0] #11 {0 1 4 12 16 19 20 24 25 26 28 }

SC_2: [0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0] #4 {3 6 9 22 }

SC_3: [0 1 0 0 0 0 0 1 1 1 0 0 0 0 0 0] #2 {11 18 }

For k=4 and l=3, the four clusters identified are shown as above. The first subspace cluster (SC_0) lies in a subspace that contains two features: activity 8 and activity 12. SC_0 contains 13 students, and they are: stu2, stu5, stu7, and etc. A simple investigation shows that SC_2 and SC_3 contain all students who failed the class. SC_2 suggest that 4 out of 6 students who failed this class had difficulty with activities 6 and 7, and SC_3 shows that the other two students who failed had difficulty with activities 2, 8, 9 and 10. We can also see that SC_1 and SC_3 are two clusters that are best contrasted by activities 6, 7 and 8. Since all students in SC_1 passed the course while SC_3 students failed the course, these three activities may be crucial for students to pass the course.

3 Conclusions

All three types of clusters provide us with different perspectives. Since not all students experience the same difficulty in all activities, subspace clustering seems to be well suited for this purpose.

References

- [1] Aggarwal C. C., Wolf J. L., Yu P. S., Procopiuc C., and Park J. S. Fast Algorithms for Projected Clustering, *Proc. of the 1999 ACM SIGMOD international conference on Management of data*, 1999, p. 61-72
- [2] Hall M., Frank E., Holmes G., Pfahringer G., Reutemann P., and Witten I. H. The WEKA Data Mining Software: An Update, *SIGKDD Explorations*, Volume 11, Issue 1, 2009
- [3] Müller E., Günnemann S., Assent I., Seidl T. Evaluating Clustering in Subspace Projections of High Dimensional Data, *Proc. 35th International Conference on Very Large Data Base*, 2009

Analyzing Learning Styles using Behavioral Indicators in Web based Learning Environments

Nabila Bousbia^{1,2} Jean-Marc Labat² Amar Balla¹ Issam Rebai³
{nabila.bousbia, jean-marc.labat}@lip6.fr, issam.rebai@limsi.fr
{n_bousbia, a_balla}@esi.dz

¹LMCS, Ecole nationale Supérieure d'Informatique, ESI ex INI, Algiers, Algeria

²LIP6, Laboratoire de Paris 6, Paris, France

³LIMSI-CNRS, Université de Paris-Sud, Orsay, France

Abstract. It is argued that the analysis of the learner's generated log files during interactions with a learning environment is necessary to produce interpretative views of their activities. The analysis of these log files, or traces, provides "knowledge" about the activity we call indicators. Our work is related to this research field. We are particularly interested in automatically identifying learners' learning styles from learning indicators. This concept, used in several Educational Hypermedia Systems (EHS) as a criterion for adaptation and tracking, belongs to a set of behaviors and strategies in how to manage and organize information. In this paper, we validate our approach of auto-detection of student's learning styles based on their navigation behavior using machine-learning classifiers.

1 Motivation

Several studies are currently being done on measuring Learning Styles (LS) by the analysis of learners' interaction traces (*eg.* DeLeS [6], Welsa [7], and Chang et al. [5]). Their general criticism is related to the use of a specific environment, and therefore specific traces and indicators. Our ambition is to develop an approach and interpretable indicators as independently as possible from the learning environment. What leads us to deal with Web-based learning environments widely used by EHS. The problem is to infer automatically high-level information about the learner preferences (behaviors and LS) from low-level ones: the navigation traces (visited URLs, clicks, etc.).

2 Approach

To validate our assumption that it is possible to deduce LS from navigational behavior, we made an experiment with 45 graduate students at the Higher National School of Computer Science (ESI-Algiers). They worked on machines equipped with a trace collection tool, with a web-based learning course. Based on their navigation traces, we calculate the five indicators we propose [2] to describe the learner's browsing behavior, to identify two attributes of the learning process layer of our LS model [1]: information processing and understanding. Their values correspond to two dimensions of the FSLSM [3]: active/reflective, and sequential/global. We used supervised classification methods to compare the psychological questionnaire ILS [4] results to those of four classifiers (K-Nearest Neighbor, decision trees, Bayesian Networks, and neural networks). We used the Weka tool and the cross validation method using 10 partitions, to address the sample size problem. Table 1 summarizes the obtained results, using the recall metric (number of

correctly classified participants by the classifier over the number of participants that it should find according to ILS).

Table 1. LS Classification results

Classification Method \ LS Attribute	Information Processing			Understanding		
	Active	Reflective	ACT/REF	Sequential	Global	SEQ/GLO
K-NN (K=3)	78.6%	41.2%	64.4%	47.4%	84.6%	68.9 %
Decision Trees C4.5	92.9%	0%	57.8 %	63.2%	73.1%	68.9 %
Bayesian Networks	82.1%	11.8%	55.6 %	42.1%	65.4%	55.6 %
Neural Networks	60.7%	64.7%	62.2%	63.2%	80.8%	73.3 %

Through Table 1, we notice that for the information processing LS' attribute, all the classifiers learn the active style better than the reflective one, except for Neural Networks. This is due to the stronger presence of active learners than reflective ones. Concerning the understanding LS' attribute, the global style was better learned by all classifiers than the sequential one for the same reason as the first attribute, where neural networks give the best total results. We observe that the total results are all over 50%. Thus, we can strengthen the hypothesis of the possibility to deduce information about learner preferences using simple navigational information that we can apply on any learning environment on the Web, without having to consider evaluation scores or the communication tool traces that allow us to give more details. We plan to continue the development of other indicators to improve the LS' identification results.

References

- [1] Bousbia, N., Balla, A., and Rebaï, I. Measuring the Learners' Learning Style based on Tracks Analysis in Web based Learning. *In 14th IEEE Symposium on Computers and Communications, ISCC, 2009*, p. 98-103.
- [2] N. Bousbia, J.-M. Labat, I. Rebaï et A. Balla. Indicators for Deducing the Learners' Learning Styles: Case of the Navigation Typology Indicator. *In the 9th IEEE International Conference on Advanced Learning Technologies, ICALT, 2009*, p.385-389.
- [3] Felder, R. M., Silverman, L. K. Learning and teaching styles in engineering education. *Engineering Education*, 1988, 78(7).
- [4] Felder, R., Soloman, B. A. ILS: Index of Learning Styles. 1996. <http://www.ncsu.edu/felder-public/ILSpa.html>, visited the 19/02/2010.
- [5] Chang, Y.-C., Kao, W.-Y. Chu, C.-P., and Chiu, C.-H. A learning style classification mechanism for e-learning. *Computers & Education*, 2009, 53(2), p. 273-285.
- [6] Graf, S. (2007). Adaptativity in Learning Management Systems Focussing on Learning Styles. Austria, Vienna University of Technology.
- [7] Popescu, E. Dynamic adaptative hypermedia systems for e-learning. *PhD. thesis*, 2008. University of Criova, Romania.

Using Topic Models to Bridge Coding Schemes of Differing Granularity

Whitney L. Cade and Andrew Olney
{wlcade, aolney}@memphis.edu
Institute for Intelligent Systems, University of Memphis

Abstract. While Intelligent Tutoring Systems (ITSs) are often informed by the data extracted from tutoring corpora, coding schemes can be time consuming to implement. Therefore, an automatic classifier may make for quicker classifications. Dialogue from expert tutoring sessions were analyzed using a topic model to investigate how topics mapped on to pre-existing coding schemes of different granularities. These topics were then used to predict the classification of words into moves and modes. Ultimately, it was found that a decision tree algorithm outperformed several other algorithms in this classification task. Improvements to the classifier are discussed.

1 Introduction

While expert human-to-human tutoring is considered to be the most effective form of tutoring [2], human tutors are costly and in short supply. Therefore, researchers strive to understand their pedagogical techniques and implement them in an Intelligent Tutoring System (ITS). To understand what these techniques are and how they are implemented, corpus analysis is often used to study tutors. These data are noisy and complex by nature, but coding schemes are one method of understanding the data in a corpus. However, coding schemes take time and manpower to implement, and are not always cost-effective. Automatic tools are quicker and can also provide some information about a corpus. One automatic tool is the Latent Dirichlet allocation (LDA) model [1], or a topic model. This method is unsupervised and easily interpretable, and the output can be used to “tag” words as belonging to a certain semantic category. This makes the topic a possible feature that could be used in either a larger classifier or a manual coding scheme. In this study, we examine the possibility of using a word’s topic as derived from a topic model to predict its label in two coding schemes of differing grain sizes.

2 Methods

We used a previously collected corpus of tutoring sessions conducted by expert tutors (see [3]). 40 tutoring sessions were recorded and transcribed, then coded according to two coding schemes. The move coding scheme (with 43 components) is a fine grained coding scheme, usually taking less than 1 conversational turn. It tends to capture small pedagogical and motivational phrases. The mode coding scheme (with 8 components) is a coarse-grained coding scheme, usually taking 10 or more turns, and it captures the overall structure of the tutoring session.

The transcripts were cleaned of common high-frequency words, i.e. “stop words”, so that only content words remained. Each conversational turn was made into a single document. These transcripts were input to topic modeling software of our own design, with the

number of topics set to 100, the prior for topics appearing in a document (α) set to 1, and the prior for words appearing in a topic (β) set to 0.01. The topic model assigned each word in the corpus to a topic which was then paired with the move and mode category associated with each word. Preliminary explorations of the data revealed that three dialogue moves were highly gregarious; therefore, words from these categories were not used in either the move or the mode analyses. After every word's topic was paired with its corresponding move or mode, it was then formatted for the Weka machine learning toolkit. Weka allows comparison of several machine learning algorithms on a variety of dimensions such as percent classified correctly. We used Weka to answer the following question: given the topic, can the mode or move be predicted?

3 Results & Discussion

Five machine learning algorithms were chosen to classify the data, and one algorithm was selected to serve as the baseline for comparison (for further information on each algorithm, see [4]). They are: ZeroR (baseline algorithm, chooses the majority class), J48 (decision tree algorithm), IBk (k=10; nearest-neighbor learner), LogitBoost (boosting algorithm) and SMO (support vector machine algorithm). Ultimately, all algorithms performed better than the *ZeroR* baseline algorithm for both the move (baseline: 10.05% correct) and mode (baseline 50.15% accurate) coding scheme ($p < .05$). Although many algorithms post similar results, J48 has the advantage that its associated decision tree is easily interpretable. Its accuracy is 19.19% for the move coding scheme, and 52.30% for the mode coding scheme.

In conclusion, it seems that topics alone are a viable predictor for modes and moves, but they do not provide a full picture by themselves. A confusion matrix of each coding scheme's results reveals that moves and modes with high entropy (where the content relies heavily on the domain rather than a formulaic saying) are harder to predict than low entropy categories. Additionally, it seems apparent that modes are harder to predict than moves, which may be due to their context-dependent nature. These results may be useful first steps in building an online classifier for an ITS.

References

- [1] Blei. D., Ng, A., Jordan, M. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 2003, 3, p. 993–1022.
- [2] Bloom, B. S. The 2 sigma problem: The Search for Methods of Group Instruction as Effective as One-to-One Tutoring. *Educational Researcher*, 1984, 13, p. 4 - 16.
- [3] Person, N. K., Lehman, B., Ozburn, R. Pedagogical and Motivational Dialogue Moves Used by Expert Tutors. Presented at the *17th Annual Meeting of the Society for Text and Discourse*, 2007.
- [4] Witten, I.H., Frank, E., *Data Mining: Practical Machine Learning Tools and Techniques with Java implementations*, 1998. San Francisco, CA: Morgan Kaufmann.

A Distillation Approach to Refining Learning Objects

John Champaign and Robin Cohen
jchamapi@cs.uwaterloo.ca, rohen@ai.uwaterloo.ca
David R. Cheriton School of Computer Science, University of Waterloo

1 Introduction

McCalla's ecological approach to e-learning systems [2] is described as “attaching models of learners to the learning objects they interact with, and then mining these models for patterns that are useful for various purposes.” The starting point of our research is to honour McCalla's ecological approach by identifying which users in a system are similar to each other, to then preferentially recommend learning objects¹ that similar students have found useful.

We classify learning objects as atomic or non-atomic. An atomic learning object would be something that is indivisible, such as a simulation or a flash quiz that would need to be reprogrammed to break it into smaller parts. A non-atomic learning object would be something like a book, which could be broken into chapters, which in turn could be broken into sections, paragraphs, sentences or phrases.

Our approach in this work is to provide students with tools to (optionally) divide non-atomic lessons, turning a single learning object into multiple learning objects (at least one of which they feel is more valuable than the previous whole).

2 Corpus Algorithm

A function is needed that will divide the learning object, based on the student's suggestion, into two or more learning objects. Of these objects, the student will specify which are worthwhile (and implicitly, the remainder will be determined to be less worthwhile). The worthwhile objects are considered to have a good interaction with the dividing student, while the others are considered to have had a bad interaction with her. The newly created learning objects are then available to be assigned to students using the ITS.

For example, Carol has been watching a supplemental video about Scheme for her CS 101 class, and found it to be not very useful except for one part that gave a very clear analogy for recursion which she found useful. Within the ITS (at the completion of the lesson), she uses the clipping functionality to designate the beginning of this useful section and the end. Three new learning objects are added to the system, the beginning of the lecture, the section she found useful, and the end of the lecture.

In order to determine which learning object is shown to a student, the initial assignment can be done in a number of ways. In previous work [1] this was done by assigning learning objects which were beneficial to similar students. If a learning object that has been divided is assigned, the system will consider the student's history and whether or not

¹ Learning objects can be considered anything that teaches a student something and can include, for example, chapters from a book, video, podcast, set of practice questions or training simulator.

he has already experienced one of the parts of the object already. If he has, then the learning object as a whole will be rejected (and another object assigned to the student). Conversely, if he has experienced one of the pieces, the learning object as a whole will no longer be a candidate.

As an example, Bob has watched the section of the lecture that Carol highlighted above and found it useful. Because it is similar to the original learning object, the system next recommends the complete lecture to him. Since he has already watched part of this, it isn't worthwhile to show him the entire lecture again and the system silently replaces this selection. It shows him the final part of the lecture instead. If no students found a certain object worthwhile, eventually that object would be ejected from the system.

This algorithm will be used to consider the set of all learning objects in the ITS and reason about which learning objects should be retained and which add little instructional value to any of the students who use the system. In its simplest form this can be considered a threshold of performance, below which a learning object is no longer shown to students.

At the same time, the system would track interactions so that the highest recommended objects may then be shown to other students. For example, if three "worthwhile" chapters are highlighted as recommended by one student, these pieces can be more useful than the whole. As a result of this positive interaction, these learning objects can be provided to similar students.

3 Validation

We are interested in validating this work first with simulated students and ultimately with real students. Previous work [1] has used simulations of students to demonstrate the effectiveness of techniques on groups of students that would be unfeasibly expensive to arrange for human trials.

In such an experiment, the effectiveness of this approach will be contrasted with a simulation where learning objects are randomly divided and to a simulation where the effectiveness of divisions is pre-calculated and only worthwhile refinements are made. This will allow us to position our approach between the base and optimal cases.

References

[1] Champaign, J., Cohen, R. A Model for Content Sequencing in Intelligent Tutoring Systems Based on the Ecological Approach and Its Validation Through Simulated Students. *Proceedings of the 23rd International FLAIRS Conference*, 2010. Daytona Beach, Florida.

[2] McCalla, G. The Ecological Approach to the Design of E-Learning Environments: Purpose-based capture and use of information about learners. *Journal of Interactive Media in Education: Special Issue on the Educational Semantic Web*, 2004, 7, p. 1–23

A Preliminary Investigation of Hierarchical Hidden Markov Models for Tutorial Planning

Kristy Elizabeth Boyer^{*}, Robert Phillips¹, Eun Young Ha, Michael D. Wallis¹,
Mladen A. Vouk, and James C. Lester

Department of Computer Science, North Carolina State University

¹Dual Affiliation with Applied Research Associates, Inc., Raleigh, North Carolina

^{*}Corresponding author: keboyer@ncsu.edu

For tutorial dialogue systems, selecting an appropriate dialogue move to support learners can significantly influence cognitive and affective outcomes. The strategies implemented in tutorial dialogue systems have historically been based on handcrafted rules derived from observing human tutors, but a data-driven model of strategy selection may increase the effectiveness of tutorial dialogue systems. Tutorial dialogue projects including CIRCSIM-TUTOR [1], ITSPOKE [2], and KSC-PAL [3] have utilized corpora to inform the behavior of a system. Our work builds on this line of research by directly learning a hierarchical hidden Markov model (HHMM) for predicting tutor dialogue acts within a corpus. The corpus was collected during a human-human tutoring study in the domain of introductory computer science [4]. We annotated the dialogue moves with dialogue acts (Table 1). The subtask structure and student problem-solving action correctness were also annotated manually.

Table 1. Dialogue act annotation scheme

Dialogue Act Tag	Description
ASSESSING QUESTION (AQ)	Request for feedback on task or conceptual utterance.
EXTRA-DOMAIN (EX)	Asides not relevant to the tutoring task.
GROUNDING (G)	Acknowledgement/thanks.
LUKEWARM CONTENT FEEDBACK (LCF)	Negative assessment with explanation.
LUKEWARM FEEDBACK (LF)	Lukewarm assessment of task action or conceptual utterance.
NEGATIVE CONTENT FEEDBACK (NCF)	Negative assessment with explanation.
NEGATIVE FEEDBACK (NF)	Negative assessment of task action or conceptual utterance.
POSITIVE CONTENT FEEDBACK (PCF)	Positive assessment with explanation.
POSITIVE FEEDBACK (PF)	Positive assessment of task action or conceptual utterance.
QUESTION (Q)	Task or conceptual question.
STATEMENT (S)	Task or conceptual assertion.

We trained first-order Markov (bigram) models, HMMs, and HHMMs on the annotated sequences. In ten-fold (five-fold for the HHMMs due to data sparsity) cross-validation, the HHMM (partially depicted in Figure 1) predicted tutor dialogue acts with an average 57% accuracy, significantly higher than the 27% accuracy of bigram models ($p < 0.0001$) and better than the 48% accuracy of HMMs without hierarchical structure ($p < 0.05$).³

³ p -values reported are from one-tailed two sample t -tests for equality of means with pooled variance

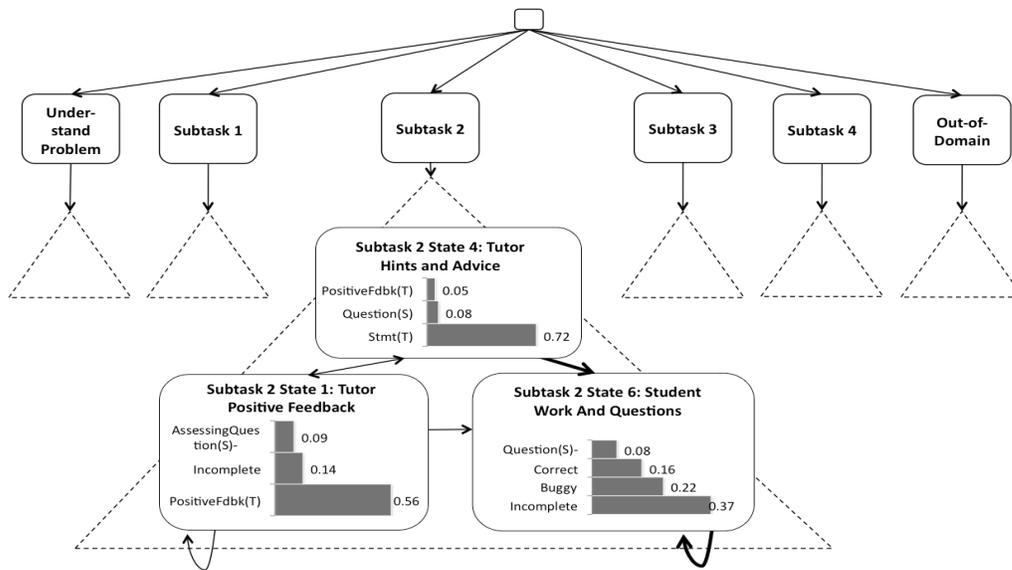


Figure 1. Subset of learned HHMM

Because of HHMMs' capacity for explicitly representing hidden dialogue structure and hierarchical task structure, they perform better than bigrams and HMMs for predicting tutor moves in our corpus. The models' performance points to promising future work that includes utilizing additional lexical and syntactic features along with fixed student characteristics within a hierarchical hidden Markov modeling framework. More broadly, the results highlight the importance of considering task structure when modeling a complex domain such as those that often accompany task-oriented tutoring. Finally, a key direction for data-driven dialogue management is to learn unsupervised dialogue act and task classification models.

Acknowledgments. This work is supported in part by the NC State Department of Computer Science and NSF grants CNS-0540523, REC-0632450, IIS-0812291, and a Graduate Research Fellowship. Any opinions, findings, conclusions, or recommendations expressed in this report are those of the participants, and do not necessarily represent official views, opinions, or policy of the National Science Foundation.

References

- [1] M. Evens and J. Michael. *One-on-One Tutoring by Humans and Computers*. Mahwah, New Jersey: Lawrence Erlbaum Associates, 2006.
- [2] K. Forbes-Riley and D. Litman. Adapting to Student Uncertainty Improves Tutoring Dialogues, *Proceedings of the 14th International Conference on Artificial Intelligence and Education*, pp. 33-40, 2009.
- [3] C. Kersey, B. Di Eugenio, P. Jordan and S. Katz. KSC-PaL: A Peer Learning Agent that Encourages Students to take the Initiative, *Proceedings of the NAACL HLT Workshop on Innovative Use of NLP for Building Educational Applications*, pp. 55-63, 2009.
- [4] K.E. Boyer, R. Phillips, A. Ingram, E. Y. Ha, M. D. Wallis, M. A. Vouk and J. C. Lester (in press). Characterizing the Effectiveness of Tutorial Dialogue with Hidden Markov Models, to appear in *Proceedings of the 10th International Conference on Intelligent Tutoring Systems*, 2010.

Higher Contributions Correlate with Higher Learning Gains

Carol Forsyth¹; Heather Butler², Arthur C. Graesser¹, Diane Halpern²
Keith Millis³, Zhiquaing Cai¹, Jonathan Wood¹
{cmfrsyth,graesser,zcai}@memphis.edu, kmillis@niu.edu,
{dhalpern,heather.butler}@cmu.edu

¹Institute for Intelligent Systems, The University of Memphis

²The Department of Psychology, Claremont McKenna College

³The Department of Psychology, Northern Illinois University

Abstract: Students interacted with an Intelligent Tutoring System called *Operation ARIES!*, which involves two agents interacting with the human in natural language dialogues. We investigated the conditions in which the length of the students' contributions is correlated with learning. Word count and the proportional learning gains scores were correlated, especially in the later phases of the curriculum. The link between student contribution length and learning supports previous findings in human one-on-one tutoring.

1 Introduction

Previous research investigating one-on-one human tutoring sessions have reported a positive correlation between the amount of information a student contributes and learning gains [1,5]. Intelligent Tutoring Systems (ITS) with natural language interaction are often designed to use the same pedagogical techniques as human tutors [2]. Therefore, the present research will use an ITS to investigate the hypothesis that the more information the students contribute, the more they will learn. The students will interact with *Operation ARIES! (Acquiring Research Investigative and Evaluative Skills)*, which teaches twenty-one topics related to scientific inquiry skills and requires the students to have three-way conversations (dialogs) with two animated pedagogical agents.

2 Description

2.1 Procedure

The experiment used a within-subjects design in which each college student completed the same task of learning about science inquiry skills through interacting with *ARIES*. The participants were 11 undergraduates enrolled in three diverse colleges, who were paid \$100 for participating in the study which occurred over several weeks lasting 6-12 hours total. They were given a pretest on the *ARIES* material, followed by a video that explains the plot of the game aspects of *ARIES*, followed by training over multiple sessions and finally a posttest. During interaction with *ARIES*, the subjects are scaffolded by two artificial agents in order to teach the topics presented within *ARIES*. For each chapter the students read an electronic text of approximately 8 pages, answer 6 multiple

choice questions, and hold trialogs with agents on content related to the questions. Upon completion of interaction with *ARIES*, the students completed a posttest consisting of seventeen open-ended questions designed to cover the topics presented and were scored using a rubric which allowed for a maximum score of 34 points.

2.3 Analysis

The pretests and posttests were graded by a rubric which was designed to score the open-ended questions based off of the expectations presented within the trialogs. In order to account for the varying levels of prior knowledge, proportional learning gains scores (PLG) were calculated using the formula $(\text{posttest} - \text{pretest}) / (1 - \text{pretest})$ [4] allowing for the students to be clustered into different levels of learning. Among the 11 students, there were a total of 1755 human turn contributions. These turn contributions were entered into Coh-Metrix [3] in order to extract the word count index as well as other characteristics of language that will be reported at the conference. An analysis of variance was performed on word count as a function of chapters, with the PLG clusters held as a random factor. There was a significant increase in words as a function of chapter number [$F(19,1693)=3.032, p<.01$], a significant effect of PLG cluster [$F(2,1693)=5.291, p<.01$], as well as significant Chapter X PLG interaction [$F(37,1693)=1.659, p<.01$]. The pattern of interaction showed that students with higher PLG scores had more words per turn, particularly for later chapters whereas students with lower PLG scores had fewer words and no change over chapters. We conclude that the size of the student's contribution positively correlates with learning, but only in later phases of the curriculum.

[1]Chi, M.T.H., Siler, S., Yamauchi, T., Jeong, H. & Hausmann, R. Learning from Tutoring. *Cognitive Science*, 2001, 25, p 471- 534.

[2]Graesser, A. C., D'Mello, S., Cade, W. Instruction based on tutoring. In R.E. and P.A. Alexander (Eds.) *Handbook of Research on Learning and Instruction*, 2009. London: Routledge Press.

[3]Graesser, A.C., McNamara, D.S., Louwerse, M.M., & Cai, Z. Coh-Metrix: Analysis of text on cohesion and language. *Behavioral Research Methods, Instruments, and Computers*, 2004, 36, p 193-202.

[4]Jackson, G.T., Graesser, A.C., & McNamara, D.S. What students expect may have more impact than what they know or feel. In V. Dimitrova, R. Mizoguchi, B. Du Boulay, B., & A.C. Graesser (Eds.) *Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling*, 2009. Amsterdam: IOS Press.

[5]Litman, D.J, Rose, C.P., Forbes-Riley, K., VanLehn, K., Bhembe, D., and Silliman, S. Spoken versus typed human and computer dialogue tutoring. *International Journal in Education*, 2006, 16, p145-170.

Pinpointing Learning Moments; A finer grain $\mathbf{P}(\mathbf{J})$ model

Adam B. Goldstein¹, Ryan S.J.d. Baker², Neil T. Heffernan¹

¹Department of Computer Science, Worcester Polytechnic Institute
100 Institute Road, Worcester MA 01609, USA
abgoldstein@gmail.com, nth@wpi.edu

²Department of Social Science and Policy Studies, Worcester Polytechnic Institute
100 Institute Road, Worcester MA 01609, USA
rsbaker@wpi.edu

1 Introduction

Educational data mining and knowledge engineering methods have led to increasingly precise models of students' knowledge as they use intelligent tutoring systems. The first stage in this progression was the development of Bayes Nets and Bayesian frameworks that could infer the probability that a student knew a specific skill at a specific time from their pattern of correct responses and non-correct responses (e.g. errors and hint requests) up until that time [cf. 2, 4, 5].

However, while the extensions made in recent years to educational data mining have created the potential for more precise assessment of student knowledge at a specific time, these models do not tell us *when* the knowledge was acquired. Baker, Goldstein, and Heffernan proposed the idea of a model that can infer the probability that a student learned a skill at a specific step during the problem-solving process. This model, $\mathbf{P}(\mathbf{J})$ for JustLearned, was shown to be a consistent predictor of high eventual \mathbf{L}_n values if a spike in $\mathbf{P}(\mathbf{J})$ is seen. This ability can potentially allow for engineering intelligent tutoring systems to bias content in a way that can induce these moments of learning. The prior approach achieved a correlation coefficient of 0.446, which leaves considerable room for improving accuracy towards achieving more effective use. Below we will discuss the application of $\mathbf{P}(\mathbf{J})$ to multiple intelligent tutoring systems as well as our most recent and more accurate attempt at creating a $\mathbf{P}(\mathbf{J})$ model in the ASSISTment tutoring system.

2 The $\mathbf{P}(\mathbf{J})$ Model

The original analysis of $\mathbf{P}(\mathbf{J})$ used data from 232 students' use of a Cognitive Tutor curriculum for middle school mathematics [3], during the 2002- 2003 school year. These students made 581,785 transactions (either entering an answer or requesting a hint) on 171,987 problem steps covering 253 skills. In [1] it was demonstrated that a model as described above can be created. This model calculates $\mathbf{P}(\mathbf{JustLearned})$, $\mathbf{P}(\mathbf{J})$ for short, which is the probability that a student just learned a skill after a certain problem step. This concept can be expressed in terms of BKT as $\mathbf{P}(\sim \mathbf{L}_n \wedge \mathbf{T} \mid \mathbf{A}_{+1+2})$. For each problem step, [1] used a set of 25 features describing the first action on problem step \mathbf{N} . These features had in turn been used in prior work to develop automated detectors of off-task behavior [2] and gaming the system. This attempt at a $\mathbf{P}(\mathbf{J})$ model achieved a correlation coefficient of 0.446 when running Linear Regression, which provides an acceptable level

of prediction, but is recognizably limited.

3 Refinements to P(J)

Although the original P(J) model was capable of identifying potential moments where learning opportunities happen, the features set constructed for the model did not have an optimally high correlation coefficient. This can perhaps be attributed to features that were designed to detect guessing and slipping rather than moments of learning, as well as the model only involving the first action of the step whereas learning may occur in subsequent actions.

To improve accuracy, and to demonstrate the ability to create a more accurate model of P(J), we redesigned our feature set and used data from ASSISTments, an intelligent tutoring system led by Professor Neil Heffernan at WPI. This new data set is pulled from 4187 students from New England middle and high schools that were using the system from 2008-2010. This data includes 55 unique skills and 418,513 logged actions. One benefit of using ASSISTments is the fact that it has extensive information about scaffolding problem steps, as well as – importantly – subsequent actions after a student's first attempt at answering. Our new feature set includes 40 features, including information about time spent on scaffolding, number of hints used in scaffolding, and so on. This new model achieves a correlation coefficient of 0.61 when running Linear Regression in Rapid Miner with 6 fold student-level cross validation.

The additional correlation shows our ability to more accurately predict P(J), a construct which has been shown to have potential to recognize moments of student learning.

4 References

- [1] Baker, Ryan S.J.d, Goldstein, Adam B., Heffernan, Neil T. : Detecting the Moment of Learning. *Intelligent Tutoring Systems*, 10, 26-35 (2010).
- [2] Corbett, A.T., Anderson, J.R.: Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction*, 4, 253-278 (1995).
- [3] Koedinger, K.R.: Toward evidence for instructional principles: Examples from Cognitive Tutor Math 6. In: *Proceedings of PME-NA XXXIII (the North American Chapter of the International Group for the Psychology of Mathematics Education)* (2002).
- [4] Martin, J., VanLehn, K.: Student Assessment Using Bayesian Nets. *International Journal of Human-Computer Studies*, 42, 575-591 (1995).
- [5] Shute, V.J.: SMART: Student modeling approach for responsive tutoring. *User Modeling and User-Adapted Interaction*, 5 (1), 1-44 (1995).

Predicting Task Completion from Rich but Scarce Data

José P. González-Brenes and Jack Mostow

{joseg, mostow}@cs.cmu.edu

Project LISTEN, School of Computer Science, Carnegie Mellon University

We present a data-driven model for predicting task completion in Project LISTEN's Reading Tutor, which takes turns picking stories and listens to the child read aloud [1]. However, children do not always finish stories, and we would like to understand why, or at least detect when they are about to stop. So our EDM challenge is to learn a model to predict task completion – a widely used metric of dialogue systems' performance. Such a model could help detect imminent disengagement in time to address it, and identify factors that influence task completion, including tutor behaviors, thereby providing useful guidance to make tutors engage students longer and more effectively.

The richness of multimodal tutorial interaction over time makes the space of possible features to describe it large relative to the amount of data. When the number of features is large compared to the amount of data, classifier learners tend to overfit the data, so we need a method that learns robust models from few training examples with many features.

Consider the supervised learning problem with training data $S = \{(x^{(i)}, y^{(i)})\}$, $i = 1 \dots n$, where each data point is a p -dimensional vector $x^{(i)}$, and $y^{(i)}$ is its label. The number of features p may exceed the number of data points ($p \gg n$). A binary logistic regression model has the following form, where the vector θ contains the p parameters of the model:

$$p(y = 1 | x; \theta) = \frac{1}{1 + \exp(-\theta^T x)} \quad (1)$$

ℓ_1 -regularized logistic regression [2] finds the vector θ^* that maximizes this expression:

$$\theta^* = \arg \max_{\theta} \left[\sum_i^n \log p(y^{(i)} | x^{(i)}; \theta) \right] - \left[\lambda \|\theta\|_1 \right] \quad (2)$$

Here the term in the first box represents how well the model fits the training data according to Equation (1), and the second term penalizes the model by the sum of its parameters' absolute values ($\|\theta\|_1$). By discouraging non-zero parameters – which select the features actually used – this penalty can prevent overfitting. The hyper-parameter λ controls the trade-off between bias and variance, and can be set by internal cross-validation using a held out set of training data. For $\lambda = 0$, Equation (2) reduces to conventional logistic regression. As λ increases, the model's complexity is penalized more strongly, reducing the number of features it uses.

Our data points to test this method are 2112 story readings by 161 children, lasting four or more sentences. We want to distinguish completed readings from unfinished readings. We truncate each positive example to match the number of sentences to the one of a negative example, so as to sample potential stopping points, not just the end of the story. Negative examples are the entire unfinished readings, which can end anywhere.

We use both static and dynamic features. Static features, e.g. student grade (K-6) and story length, remain static over a story reading. Dynamic features, e.g. number of sentences read, words read per minute, or clicks logged, change throughout a reading, so we compute separate values for 1, 2, 3, and 4 sentences from the beginning and end of the reading. To avoid cheating, we exclude features of the last sentence read, e.g. whether the child clicked to exit. Altogether we have 17,163 raw, squared, and threshold features.

Figure 1 shows classification accuracy for balanced subsets of different sizes, with the same number of positive and negative examples drawn randomly from the 2112 readings. We used 10-fold cross-validation splitting data randomly. We also tried splitting across students, but this doesn't affect accuracy on the full set, yet it is noisy for small subsets due to students with sparse data. The error bars represent the 90% confidence interval. As Figure 1 shows, the method achieves 60% accuracy by training on only 500 examples, increasing to 70% with 600 examples, and asymptoting at 78% above 1500 examples. The three most predictive features are derived from the percentage of the story completed so far, consistent with the intuition that children are likelier to finish shorter stories.

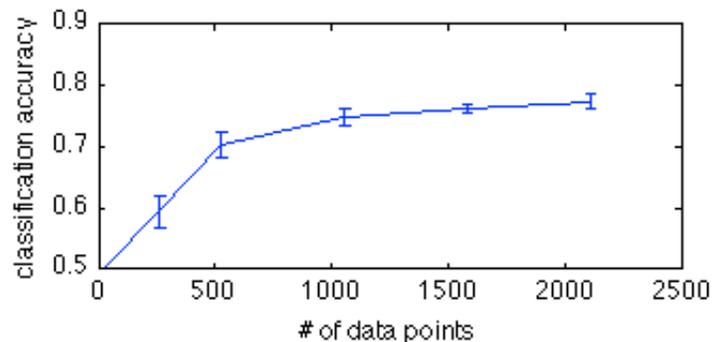


Figure 1: Classification Accuracy on Data Sets of Different Sizes

This paper has presented a novel model to predict students' task completion in a multimodal tutor, using a method that can train models from data with many dimensions but few examples. The method, used successfully elsewhere, should interest the EDM community because of its potential to cope with the curse of dimensionality inflicted by the richness of tutorial interaction.

Acknowledgements: This work was supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305A080628 to Carnegie Mellon University. The opinions expressed are those of the authors and do not necessarily represent the views of the Institute or U.S. Department of Education.

References

- [1] Aist, G. and J. Mostow. Improving story choice in a reading tutor that listens. *Proceedings of the Fifth International Conference on Intelligent Tutoring Systems (ITS'2000)*, 645. 2000. Montreal, Canada.
- [2] Ng, A.Y. Feature selection, L1 vs. L2 regularization, and rotational invariance. *International Conference on Machine Learning*, 78. 2004. New York, NY.

Hierarchical Structures of Content Items in LMS

Sharon Hardof-Jaffe, Arnon HersHKovitz, Ronit Azran, and Rafi Nachmias
{sharonh2,arnonher,roni122,nachmias}@post.tau.ac.il
Knowledge Technology Lab, School of Education, Tel Aviv University, Israel

Of the many applications enabled by new technologies, the most commonly used in higher education are Learning Management Systems (LMSs), e.g. Moodle, BlackBoard, which enable a wide range of Web-supported courses. LMSs enable the instructors to develop websites for their courses to support face-to-face teaching by means of different tools. Although most of the LMSs offer an enriched environment that goes beyond the usual content management tools (including communication tools and course management modules), these systems are mainly used for transferring information and increasing accessibility of learning materials [1-3]. Usually, the content modules in these systems enable the construction of a hierarchical repository of information items; consequently, the instructor is able to create folders and upload files creating variety of repository structures, which are presented to students in the course site.

The main purpose of this research is to empirically study the types of online hierarchical structures of content items presented to university students in Web-supported courses. Three research questions are addressed in the study: 1) What is the extent of content items presented to university students in online repositories within Web-supported courses? 2) Which types of hierarchical structures of content items are empirically revealed? 3) What are the associations between the types of structures and *Number of Items*, *Course Size*, and *Content Consumption*? Three groups of variables were defined, describing characteristics of each course, as following:

Repository Size Variables

- A. *Number of Items*: total number of content items in the repository
- B. *Number of Folders*: total numbers of folders in the repository

Repository Structure Variables

- C. *Average Folder Size*: *Number of Items* divided by *Number of Folders* ($=A/B$)
- D. *Largest Folder*: number of items in the largest folder in the repository
- E. *Largest Folder Share*: ratio of *Largest Folder* to *Number of Items* ($=D/A$)
- F. *Hierarchical Depth*: maximal repository depth (i.e., length of a path from the root)
- G. *Visible Width*: number of folders located immediately under the root; this number represents the width of the repository as presented to the students.
- H. *Width-depth Proportion*: ratio of *Visible Width* to *Hierarchical Depth* ($=E/F$).

Course Characteristics (Independent)

- I. *Course Size*: number of registered students
- J. *Content Consumption*: average consumption of content item per student

The research was carried out on a full sample of Fall term courses in Tel Aviv University (academic year 2008/9) which were accompanied by a Website within the HighLearn LMS (by Britannica Knowledge Inc.), N=1,747. Raw data was extracted using SQL queries on HighLearn databases. In this data file, each row corresponds to a single content item within the system, and documents the unique ID of the course to which the item belongs and its full path within the repository. The data file consisted of 72,753 rows (i.e., content items) of 1,747 courses. Revealing types of repository structures was done using Two-step Cluster Analysis on a reduced population which included only courses the repositories of which consisted of 15 content items or more, N=1,203.

Results suggest that *Number of Items* is largely varied between 1 and 1,029, with an average of 41.64 files (SD=69.10). The mean of *Number of Folders* was found to range between 1 and 185, with an average of 10.69 (SD=16.78). This average demonstrates a large growth in content items delivery at Tel Aviv University, comparing to earlier studies of the very same LMS [1,4].

Regarding the repository hierarchical structures, five types were found: 1) **Main-folder Structure**: no depth, almost all files piled, large folders (n=67); 2) **Extensive Filing**: high depth, small folders (n=120); 3) **Flat Small Folders**: flat hierarchy, small folders (n=222); 4) **Pile in Hierarchy Filing**: pile exists, small folders (n=354); 5) **Pile in Flat Filing**: flat hierarchy, big pile exists (n=440).

Association was found between the repository structure and *Number of Items*, according to which large repositories are associated with extensive filing. It was also found that *Course Size* is statistically significantly different between courses demonstrating Main-folder and Extensive Filing structures: The average *Course Size* took the highest value in the Extensive Filing courses (63.49, SD=61.95), and the lowest in the Main-folder courses (34.78, SD=34.43). In addition, association was found between the repository structure and its consumption, as measured by *Content Consumption*. On average, lowest *Content Consumption* was demonstrated in the Extensive Filing courses (0.77, SD=0.51), and the highest – in the Pile in Flat Filing courses (1.46, SD=0.72).

LMSs are often being studied using usage analysis for various purposes [5]. In this study, we used automatically collected data describing the structures of content items presented in Web-supported courses. However, it is not clear that this research falls into one of the 3 classical categories of Web mining (usage mining, content mining, and structure mining) [6]. As EDM research widens its horizons and examines a wide range of data originated in many different learning contexts, the categorization of Web mining studies might be re-examined.

References:

- [1] Shemla, A., and Nachmias, R. (2007). Current state of Web supported courses at Tel-Aviv University. *International Journal of E-Learning*, 6(2), 235-246.
- [2] Lonn, S., and Teasley, S. (2009). Saving time or innovating practice: Investigating perceptions and uses of Learning Management Systems. *Computers & Education*, 53(3), 686-694.
- [3] Roqueta, M. (2008). Learning management systems: A focus on the learner. *Distance Learning*, 5(4), 59-66.
- [4] Nachmias, R., and Ram, J. (2009). Research Insights from a Decade of Campus-Wide Implementation of Web-Supported Academic Instruction at Tel Aviv University. *he International Review of Research in Open and Distance Learning*, 10(2).
- [5] Romero, C., Ventura, S., and Garcia, E. (2008). Data mining in course management systems: Moodle case study and tutorial. *Computers & Education*, 51(1), 368-384.
- [6] Kosala, R., and Blockeel, H. (2000). Web mining research: A survey. *SIGKDD Explorations*, 2(1), 1-15.

Is Students' Activity in LMS Persistent?

Arnon Hershkovitz and Rafi Nachmias

{arnonher,nachmias}@post.tau.ac.il

Knowledge Technology Lab, School of Education, Tel Aviv University, Israel

Introduction. The most common method of blending the Internet in higher education today is by implementing Web-supported instruction, in which traditional face-to-face courses have auxiliary materials, usually using Learning Management Systems (LMS), e.g. WebCT, Moodle. Research of LMS in higher education has barely involved the examination of the individual's behavior over the learning period. Furthermore, although a large body of research exists regarding persistence in fully online learning configurations, only little was studied regarding the online persistence in Web-supported configurations. When empirically examining usage of Web learning environments, it has been noticed that two phenomenon are repeatedly occurring regarding volume and trends of activity: a) **Many are little active, while some are extensively active;** b) **Overall decrease in visiting** (usually with some spikes of access immediately before exams, assignment submission deadlines, or any other important events during the course) [1-4].

This study aims on identifying individuals' over-time patterns of online activity in Web-supported courses, both by volume and trends of activity. As our examination of patterns of persistence crosses courses, it might also promote the revealing of differences between courses regarding students' persistence within them.

Population. Log files of 58 Moodle one-semester-course websites offered by Tel Aviv University (TAU) in the academic year 2008/9 were analyzed (a full sample of the Moodle-supported courses; only logged activity from during the calendared term period were taken). Moodle's log files consist of actions taken within the course websites' modules, including: text pages, resources, forums, and users. Actions might be: viewing, adding, updating, or deleting. In total, 163,685 records of 1189 students were logged, and there were 1897 student enrollments which served as the basic analysis units (interdependence in the population was found to be insignificant).

Variables and Process. Five measures were calculated to describe students' activity in volume (*Cumulative Activity*, *Total Activity*) and trend (*First Tertile Proportion*, *Second Tertile Proportion*, *Activity per Day*). The main procedure involves the application of a Decision Tree algorithm on the trends-related variables, for finding patterns of persistence in students' behavior, and for defining rules of belonging to these patterns. We choose the variable *Activity per Day* as the independent variable the prediction of which should be given by the tree, and the two other variables – i.e., *First/Second Tertile Proportion* – as the variables according to which the tree will be constructed. CHAID method was used as an attribute selection measure based on the statistical chi-square test for independence, with a significance level of 0.05 for splitting nodes and merging categories, and a 10-fold cross validation.

Results and Discussion. Analyzing *Total Activity*, it was re-demonstrated that most of the students present low activity, while only a little are very active. Regarding the

activity by tertiles, it was found that on average, *First Tertile Proportion* is 0.28 (STD=0.36), and *Second Tertile Proportion* is 0.62 (STD=0.34). For a consistent student, the activity of whom is equally distributed over the term, we would expect values of these two variables to be 1/3, 2/3, respectively. T-tests for comparing the means of the tertile-related variables with those of a consistent user's behavior confirmed that the differences are statistically significant, with $t(1896)=6.65$, $p<0.01$, for *First Tertile Proportion*, and $t(1896)=5.63$, $p<0.01$, for *Second Tertile Proportion*.

Running the Decision Tree algorithm, we got seven patterns of learners' activity: a) **Persistent Users**, active occasionally throughout the term; b) **High-extent Persistent Users**, active often throughout the term; c) **Mid-Late Users**, active during second, last thirds of the term; d) **High-extent Mid-Late Users**, active during second, last thirds of the term with high intensity; e) **Late Users**, active almost only during last third of the term; f) **Retain Users**, active almost only during first third of the term; and g) **Low-extent Users**, with overall low volume of activity. The largest groups were Late Users (with 23% of the students), Retain Users (20%) and High-extent Persistent Users (19.8%).

The three most prominent groups found are consistent with research about online learning: 1) Late Users behavior corresponds to the phenomena of students visit courses' websites towards the term-end exams (or other important dates during the semester) [3]; 2) Retain Users' rate of about twenty percents might be compared to studies which examined retain from online learning configurations. However, retention rate in online learning is largely varied between studies, and can get up to 84% [5]; 3) High-extent Persistent Users is a behavior which might be associated with high motivation (either internal, i.e., to learn as much as they can from this engagement, or external, e.g., to gain more points in the exam) or satisfaction, as was previously demonstrated regarding online courses [6].

Currently, as universities provide instructors and students with LMS for facilitating blended learning, it is important to understand how these systems are being used in practice. Instructors may use these results for making their teaching more efficient; education researchers might clarify the large interpersonal differences among students regarding online persistence; and university policy-makers may deepen their knowledge of the cost-effectiveness ratio of these systems.

References

- [1] Nachmias, R., & Segev, L. (2003). Students' use of content in Web-supported academic courses. *The Internet and Higher Education*, 6(2), 145-157.
- [2] Masters, K., & Oberprieler, G. (2004). Encouraging equitable online participation through curriculum articulation. *Computers & Education*, 42(2004), 319-332.
- [3] Sheard, J., Ceddia, J., Hurst, J., & Tuovinen, J. (2003). Inferring student learning behaviour from Website interactions: A usage analysis. *Education and Information Technologies*, 8(3), 245-266.
- [4] Lovatt, J., Finlayson, O. E., & James, P. (2007). Evaluation of student engagement with two learning supports in the teaching of 1st year undergraduate chemistry. *Chemistry Education Research and Practice*, 8(4), 390-402.
- [5] Neuhauser, C. (2002). Learning style and effectiveness of online and face-to-face instruction. *The American Journal of Distance Education*, 16(2), 99-113.
- [6] Levy, Y. (2007). Comparing dropouts and persistence in e-learning courses. *Computers & Education*, 48(2), 185-204.

EDM Visualization Tool: Watching Students Learn

Matthew W. Johnson and Dr. Tiffany Barnes
{mjokimoto@gmail.com} {TBarnes@uncc.edu}

Computer Science Department, University of North Carolina at Charlotte

Abstract. This poster describes a visualization tool for educators that allows the exploration of educational data. We display an entire classes sequence of actions to the user using a tree-graph. Our preliminary results suggest that EDM visualization tools are a promising area for future research in EDM.

Introduction

This poster describes a visualization tool that allows educators to visualize the process in which students solved procedural problems, in logic, using an intelligent tutoring system. The purpose of this tool is to allow educators to be able to navigate, explore and gain insights about student performance. This allows educators to better understand the strengths and deficiencies of students, so that lectures or homework adjustments can be made to better aid student learning.

The field of InfoVis has much to offer educators and data repositories of educational data, like Carnegie Mellon's Data Shop. Fekete et. al. shows us that InfoVis is well equipped for exploring data to learn more, make new discoveries, and gain insight[3]. Card et al. defines the purpose of visualization to “amplify cognition” about data [2]. In our case amplifying an educator's cognition about the way their students solve problems, a main advantage Intelligent tutoring systems have over traditional homework methods.

Related Work

This work is an extension of the work of John Stamper and Tiffany Barnes [6,1]. We extend their work to include an interactive visualization tool which centralizes and streamlines their data processing, and adds exploration and navigation interactions. In our visualization we made use of Shneidermans's seven tasks of visualization[5]: overview, zoom, filter, details-on-demand, relate, history and extract; often considered standard in information visualization.

Romero and Ventura surveyed EDM techniques in [4] “...information obtained from usage statistics is not always easy to interpret to the educators and then other techniques have to be used...Infovis techniques.” They also concluded that educational data mining tools require “good visualization facilities to make their results meaningful to educators and e-learning designers”.

EDM Visualization Tool

The EDM Vis tool is a software tool that presents student work to educators in a simple way. First, students use a logic tutor where each 'state' of the user is recorded, along with each action, common in intelligent tutoring systems. Actions take the user from one state

to another. In the case of Tic-Tac-Toe an action would be placing your 'X' or 'O' piece on the board, the state would be the resulting configuration of X's and O's. These states and actions are then displayed as a tree-graph of nodes and edges respectively.

We use logic tutor data which stores the set of premises as a single state, in a root node. Consecutive states are generated based upon the actions that were taken by the student(s). The depth of each node represents the number of steps taken. Edge width is based on the frequency of students who performed the same action(s). Filtering and other interactions allow educators to observe trends, common mistakes and gain insights into their students' ways of thinking.

Results & Future Work

The EDM Vis tool is still in development but unofficial results show we can gain insights about student progress. Our first insight was that only ten percent of students were able to find the shortest solution, or expert path. We also noticed problematic areas that we were not previously aware of. Next we will make a standard file format that can support data from the Data Shop at Carnegie Mellon, allowing access to more people and data. Also we will allow annotations to be made in the Vis tool which will export separate files, to be read into our logic tutor. This will extend the visualization tool to creation, allowing educators to gain insights then act using their new knowledge. Lastly, the EDM Vis tool is built for visualizing sequential data, so single step problems, like short answer questions, are unsupported. Developing other visualization tools to support other data types seems to be a meaningful avenue for future research.

References

1. Barnes, T., Stamper, J., Toward automatic hint generation for logic proof tutoring using historical student data. *Proceedings of the 9th International Conference on Intelligent Tutoring Systems*, 2008. p. 373-382.
2. Card, S.K., Mackinlay, J.D., Shneiderman, B., Using vision to think. In *Readings in information visualization: using vision to think* (Eds.) 1999. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.
3. J.-D. Fekete, J. van Wijk, J. Stasko, C. North, The Value of Information Visualization, in *Information Visualization: Human-Centered Issues and Perspectives*, Springer, 2008, pp. 1-18.
4. Romero, C., Ventura, S., Educational data mining: A survey from 1995 to 2005 *Expert Systems with Applications*. July 2007. Vol. 33, no. 1, pp. 135-146.
5. Shneiderman, B., The eyes have it: A task by data type taxonomy for information visualizations. *Visual Languages*, IEEE Symposium, 1996. p. 336.
6. Stamper, J., Barnes, T., Lehmann, L., Croy, M., The hint factory: Automatic generation of contextualized help for existing computer aided instruction. *Proceedings of the 9th International Conference on Intelligent Tutoring Systems Young Researchers Track*, 2008. p. 71-78.

Inferring the Differential Student Model in a Probabilistic Domain using Abduction Inference in Bayesian Networks

Nabila Khodeir¹, Nayer Wanas², Nevin Darwish³, Nadia Hegazy¹
nabilakhodeir@yahoo.com, nayerw@microsoft.com, ndarwish@ieee.org, nhegazy@mcit.gov.eg

¹Informatics Dep., Electronics Research Institute, Tahrir St., Giza, Egypt

²Cairo Microsoft Innovation Lab, 306 Chorniche El-Nile, Maadi, Cairo, Egypt

³Dept. of Computer Engineering, Faculty of Engineering, Cairo University, Giza, Egypt

Abstract. In this paper we aim to estimate the differential student knowledge model in a probabilistic domain within an intelligent tutoring system. The suggested algorithm aims to estimate the actual student model through the student answers to questions requiring diagnosing skills. Updating and verification of the model are conducted based on the matching between the student and model answers. Two different approaches to updating namely coarse and refined model are suggested. Results suggest that the refined model, although takes more computational resources, provides a slightly better approximation of the student model.

1 Introduction

The student model is a core component in any intelligent or adaptive tutoring system that represents many of the student features such as knowledge and individual traits [1]. Differential model is one from several models that have been suggested for the student knowledge modeling [5]. The differential model represents both the student knowledge and the differences between student and expert knowledge, which represents the gap in the student's knowledge [2].

We propose a method to build a differential student model in a probabilistic domain based. A Bayesian network is used to represent both the domain and the student model. Based on the domain structure we generate problems that require diagnostic skills to be solved. The discrepancies between the answers generated by the student model and the answers provided by the student are used to update the student model.

2 Estimating of the Student knowledge Model

A diagnostic question is generated and presented to the student. Subsequently, the answer provided by the student to the question is compared to that generated by the initial student model using an abduction inference through the Bayesian network [3,4]. If the answers match the student model doesn't require regulation. On the other hand, if there is a discrepancy between the answers the student model is adjusted. Since the answer is a ranked list of hypotheses, the difference between the two answers can be either (i) missing or (ii) adding or (iii) incorrect order hypotheses or (iv) any combinations of these differences. The addition or absence of a hypothesis is related to difference in the relations between domain items, On the other hand, the difference between hypotheses

order is referred to the differences in the weight of these relations. Upon identifying the type of difference, the student model should be updated accordingly. Two different approaches are used to update the student model, namely (i) coarse model update and (ii) refined model update. The coarse model update is conducted by adding or removing relations of the differences in the hypotheses and swapping between relations weights for the differences in the hypotheses order. Refined model update, on the other hand, is performed using successive increase or decrease in the weights of the different hypotheses according to the nature of the differences. Model verification is evaluated to verify that the new model generates answer which matches the answer provided by the student.

Simulated students and different groups of questions are used to evaluate the algorithm. The questions are evaluated against the initial student model, generating the original match. The performance of the algorithm is evaluated after each question. Moreover, the set of questions are all tested against the new adjusted model for verification. The performance of the algorithm is evaluated using the accuracy of estimating the student answers to the questions. Coarse and refined updating model approaches are successful in estimating the student model with an accuracy of over 84% for individual match, and 70% for verification match.

3 Discussion and Conclusion

An experimental evaluation of the approaches has been conducted. The results suggested that the refined model updating exhibits similar performance with respect to accuracy compared to the coarse updating. However, it estimates models that are closer to the actual model by at least 8% compared to the coarse updating model. Applying the questions in any order doesn't have a significant effect on the overall performance of the algorithm. On the other hand, the proximity of the initial model selected to that of the student, or to the knowledge model improves the performance of this approach significantly.

References

- [1] Brusilovsky, P., Millan, E. User Models for Adaptive Hypermedia and Adaptive Educational Systems, *In Brusilovsky, P., Kobsa, A., Nejdl, W.(eds.) The Adaptive Web, LNCS 4321*, 2007, pp. 3-53.
- [2] Burton, R, Brown, J A tutoring and student modelling paradigm for gaming environments, *SIGCSE-SIGCUE*, 1976, pp. 236-246
- [3] Elvira Project, <http://www.ia.uned.es/~elvira/index-en.html>
- [4] Nilsson, D. An efficient algorithm for finding the M most probable configurations in probabilistic expert systems, *Stat. and Comp.* 1998, 8(2), pp. 159-173
- [5] Pahl, C., Kennyand, C. Interactive Correction and Recommendation for Computer Language Learning and Training, *IEEE Trans. KDE 2009*, 21(6), pp.854-866.

Using LiMS (the Learner Interaction Monitoring System) to Track Online Learner Engagement and Evaluate Course Design

Leah P. Macfadyen¹ and Peter Sorenson²

leah.macfadyen@ubc.ca

¹Science Centre for Learning and Teaching, The University of British Columbia

²Quizzicle, LLC, Hamden, CT, USA

Abstract. This poster will describe the Learner Interaction Monitoring System (LiMS), designed to capture data demonstrating learner online engagement with course materials. The poster presentation will explain how the LiMS ‘event capture model’ collects detailed real-time data on learner behavior in self-directed online learning environments, and interprets these data by drawing on behavioral research. We believe that LiMS offers education and training managers in corporate contexts a valuable tool for the evaluation of learner performance and course design. By permitting more detailed demonstration of ROI in education and training, LiMS allows managers to make the case for web based courseware that reflects appropriate and evidence-based instructional design, rather than budgetary constraints.

1 Introduction

Businesses are increasingly recognizing the need to support continuous professional development in their workforce, and many have recognized the benefits of online training and education. Well-designed online training courses promise to make education and training available in ways that fit the busy work and life schedules of employees, and almost two decades of research into online learning has demonstrated that there is no measurable significant difference in learning outcomes between face to face and online learning modalities [1]. In the corporate context, where exam results and course grades rarely exist as measures of learner achievement or effective online course design, it falls to training and education managers to identify reliable and valid approaches to evaluating both course design and learner performance, in order to demonstrate a significant return on the sizable investments needed to implement high quality online learning.

Unfortunately, few easily implementable approaches exist. As [2] notes, “in spite of the best efforts of organizations and the professional trainers’ associations, there are significant problems in evaluating the true impact of [online] training”. Similarly, while the value of feedback from and to learners in educational settings is well-established (see, for example, [3]), most corporate training departments lack the tools to gather accurate learner feedback (direct or indirect) about their online learning experience or activities. Such information is critical in evaluating whether training courses are meeting corporate educational needs and goals. In the absence of evaluative tools that return meaningful and easily interpretable data, corporate training departments are most likely to deliver web based courseware that simply reflects budgetary restrictions.

2 Description

To meet this need, we have developed the LiMS application. LiMS is a two-part web-based plug-in application that can interface with any web-based course delivery platform to transform the online learning environment into an active observer of learner engagement with course materials. Unlike the minimalist tracking tools packaged with standard Learning Management Systems (LMSs), LiMS purposely captures fine-grained data on learner activity and behaviors within the learning environment, turning the course itself into an active receiver of indirect learner feedback. By collecting user course engagement events, such as mouse clicks or movements acting upon individual page elements such as buttons, checkboxes and lists, LiMS ensures that a learner's detailed course interaction data is captured. Going far beyond the capture of simple event sequencing, LiMS also captures data reflecting the variable behavioral characteristics of those actions such as duration, timing and response latency. Importantly, LiMS implementation and continuing development builds on existing research to permit pedagogically meaningful interpretation of captured data. At the completion of each online training experience, LiMS assigns a 'behavioral grade' to the learner reflecting their approach to the training material when compared to a standard established by LiMS itself. LiMS adjusts the learner's grade using an algorithm that computes a final assigned 'grade' reflecting their behavioral approach to online training materials. A descriptive profile of the learner is generated based on the course grade and the behavioral data, and is posted on the student's report page. LiMS implementation can then be customized to allow educational designers to ask targeted questions about learner choices within a course, or to track learner behavior in relation to key course material items or events of interest. In relation to learner behaviors, for example, educators may wish to ask questions such as: Are learners spending sufficient time reading particular core course text materials? Do my learners display differential response latency to key questions, and can this provide insight into comprehension or decision making style? Additional comparison measures permits benchmarking against peers or course norms.

LiMS therefore offers education and training managers in corporate contexts a valuable tool for the evaluation of learner performance and course design. By allowing more detailed demonstration of ROI in education and training, LiMS allows managers to make the case for web based courseware that reflects appropriate and evidence-based instructional design, rather than budgetary constraints.

References

- [1] Russell, T. L. *The No Significant Difference Phenomenon*, 1999. North Carolina State University.
- [2] Murray, L. W. & Efendioglu, A. M. Valuing the investment in organizational training. *Industrial and Commercial Training*, 2007, 39, p. 372-379.
- [3] Chickering, A. W. & Gamson, Z. Seven Principles For Good Practice in Undergraduate Education. *AAHE Bulletin*, 1987, 39, p. 3-7.

Observing Online Curriculum Planning Behavior of Teachers

Keith E. Maull¹, Manuel Gerardo Saldivar^{2,3} and Tamara Sumner^{1,3}

{keith.maull, manuel.saldivar, tamara.sumner}@colorado.edu

¹Computer Science Department, University of Colorado, Boulder

²School of Education, University of Colorado, Boulder

³Institute of Cognitive Science, University of Colorado, Boulder

Abstract. Curriculum planning is perhaps one of the most important tasks teachers must perform before instruction. While this task is facilitated by a wealth of existing online tools and resources, teachers are increasingly overwhelmed with finding, adapting and aligning relevant resources that support them in their planning. Consequently, ripe research opportunities exist to study and understand online planning behavior in order to more generally characterize planning behavior. In this paper, we introduce a web-based curriculum planning tool and study its use by middle and high school Earth science teachers. We examine the web analytics component of the tool and apply clustering algorithms to model and discover patterns of the use within the system. Our initial results provide insights into the use of the tool over time and indicate teachers are engaging in behavior that show affinity for the use of interactive digital resources as well as social sharing behaviors. These results show tremendous promise in developing teacher-centric analysis techniques to improve planning technologies and techniques to study online curriculum planning patterns.

The use of the Internet in the classroom, applied either as a direct instructional tool or as a student learning tool for research and self-directed learning, has become essential to teachers and learners alike. Much empirical research indicates that Americans in general and K-12 students in particular are using technology in their day-to-day lives more than ever before; communication technologies that leverage the Internet are particularly popular with young people [2]. A large body of education research indicates that the best learning experiences are those that make direct connections to students' existing knowledge and life experiences [1]. Thus, it is vital that K-12 education leverage Internet technology not only because it offers instructional benefits in and of itself but because it can bridge students' in-class experiences with their out-of-class lives, thus making learning personally relevant.

Tools supporting teachers through planning, organizing and integrating instruction around the complexities of individual student skill, curriculum goals, district-wide standards, etc. lack maturity, perhaps because the fluid nature of planning in general or the changing demands of the classroom. Despite the myriad of teacher resources in the form of shared ideas and re-usable lesson plans, activities, etc., successfully integrating these resources yet requires a fair amount of customization. Teachers often become overwhelmed by the customization task that it becomes *more* time consuming to re-use and re-purpose existing materials than develop their own.

This poster describes the application context, research questions, initial experiments and results of an online curriculum planning and development tool called the Curriculum Customization Service (CCS). The tool was deployed for use by 6th and 9th grade middle

and high school teachers within the Denver Public School system, and usage observations were made over the course of a semester of tool use. We detail the tool : its motivation, interface and content, as well as the web analytics data generated by the end user interactions. We describe the initial exploration and selection of data features and the application of clustering algorithms to analyze system usage. Our research focuses on developing and applying tools and techniques for observing and classifying teachers' online behavior in educational applications, offering a unique view port into educators' online usage patterns and behaviors.

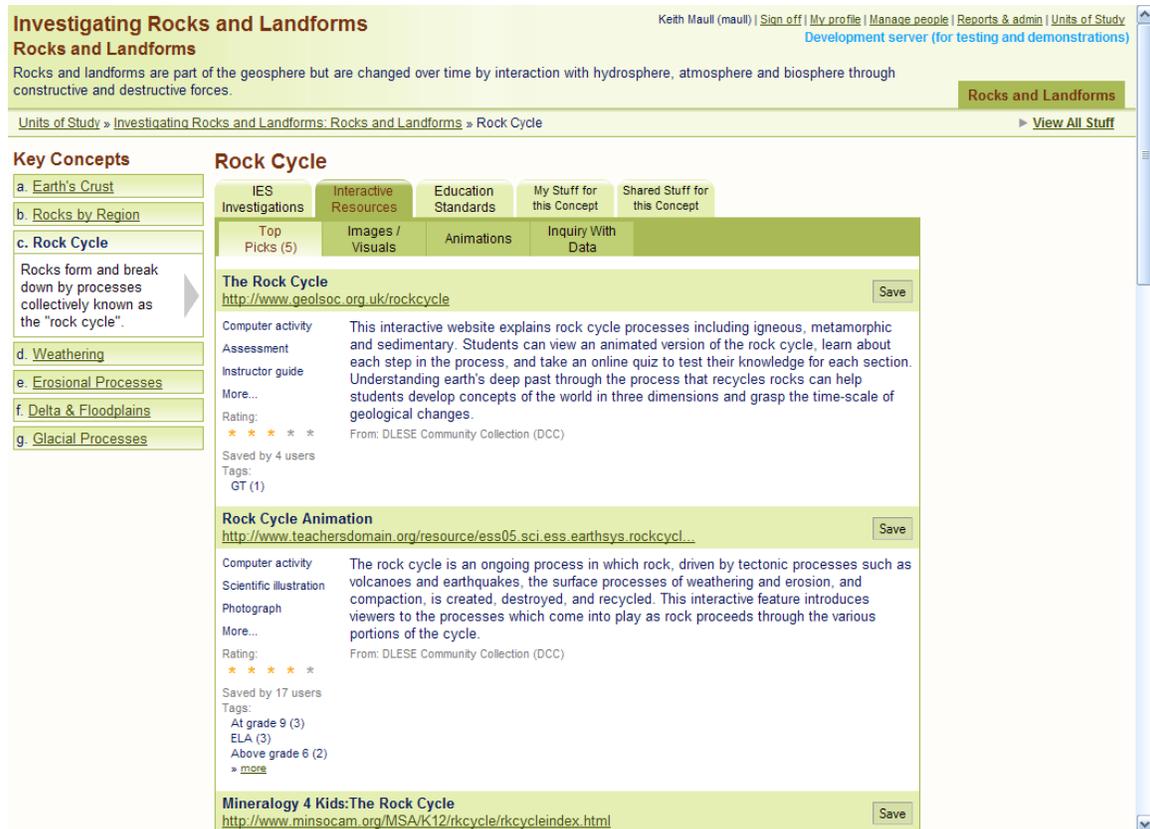


Figure 1 : The CCS Interface

References

- [1] A. Brown, J. Bransford, and R. Cocking. *How people learn: brain, mind, experience, and school*. Washington, DC: National Academy Press, 2000.
- [2] A. Lenhart, K. Purcell, A. Smith, and K. Zickuhr. *Social media & mobile internet use among teens and young adults*. Pew Internet & American Life Project, Washington, DC, 2010.

When Data Exploration and Data Mining meet while Analysing Usage Data of a Course¹

André Krüger^{1,2}, Agathe Merceron² and Benjamin Wolf²
{akrueger, merceron, bwolf}@beuth-hochschule.de
¹Aroline AG, Berlin, Germany
²Beuth University of Applied Sciences, Berlin, Germany

1 Introduction

Learning Management Systems (LMS) are web-based systems that are increasingly used in education. If lecturers want to gain a deeper insight on whether and how students use the learning resources they put at their disposal in a course, user data stored by the LMS needs to be explored. To help explore these data, we have designed and implemented an application called *ExtractAndMap* [1] that structures and exports all data stored at various places and under various forms by a LMS into a consistent data base that can be used for numerous queries and further data mining. Using queries on the data base it is possible to answer questions like: “how many students have attempted self-evaluation exercise of week 1?”, “how many students have attempted self-evaluation exercise of week 2?” and so on, till the end of the semester. When looking at the results it may well be that these numbers diminish as shown in Figure 3 p. 7 in [1].

An experienced lecturer will read in those figures the following familiar experience: at the beginning of the semester students are enthusiastic and are not yet overloaded with a lot of homework in different courses. Therefore, many of them attempt self-evaluation exercises. As the semester progresses, always less students attempt the exercises till a stable group remains that sticks to its working habits and consistently attempt all evaluation-exercises. Is this hypothesis correct? The sole answers to the above questions tackling each exercise separately cannot tell for sure. In this contribution we show that a deeper exploration with queries handling several exercises together like “how many students have attempted exercise A and exercise B?” can be enough to infer the association rule “if students attempt exercise A, then they also attempt exercise B”.

2 Inferring Association Rules from Data Exploration

For definitions about association rules and interestingness measures like *confidence*, *lift* and *cosine* the reader is referred to [2]. Data exploration involves quite often simple counting, like how many transactions contain X , Y or X and Y , thus, giving $P(X)$, $P(Y)$ or $P(X, Y)$. If exploration shows that $P(X)$ and $P(X, Y)$ are equal, then we have:

$$\text{conf}(X \rightarrow Y) = 1, \text{ lift}(X \rightarrow Y) = \frac{P(X, Y)}{P(X) \cdot P(Y)} = \frac{1}{P(Y)} \text{ and}$$

$$\text{cosine}(X \rightarrow Y) = \frac{\sqrt{P(X)}}{\sqrt{P(Y)}}, \text{ where } X \rightarrow Y \text{ is the association rule “if } X, \text{ then } Y\text{”}. \text{ In}$$

practice $P(X)$ and $P(X, Y)$ are going to be almost equal, not exactly equal. Table 1

1 This work is partially supported by the European Social Fund for the state Berlin.

shows how confidence, lift and cosine evolve. The rows show $P(X, Y)$ as a fraction of $P(X)$. $P(X, Y)=0.97P(X)$ means that $P(X, Y)$ equals 97% of $P(X)$. $P(Y)$ is taken to be 0.8 in the third column and 0.7 in the fourth column. $P(X)$ is taken to be first 2/3 of $P(Y)$ in column 5 and 3/4 of $P(Y)$ in the last column. We recall that confidence is a number between 0 and 1 (highest is 1), that lift rates a rule as interesting if its value is above 1, and that cosine is a number between 0 and 1 and rates a rule as interesting if its value is above 0.66.

Table 1. Evolution of confidence, lift and cosine

	<i>conf.</i>	<i>lift</i> $P(Y)=0.8$	<i>lift</i> $P(Y)=0.7$	<i>cosine</i> $2/3 P(Y)$	<i>cosine</i> $3/4 P(Y)$
$P(X, Y)=P(X)$	1	1.25	1.43	0.81	0.87
$P(X, Y)=0.97P(X)$	0.97	1.21	1.39	0.79	0.84
$P(X, Y)=0.95P(X)$	0.95	1.19	1.36	0.77	0.82
$P(X, Y)=0.90P(X)$	0.90	1.12	1.29	0.65	0.78

Summing up, when $P(X)$ is almost equal to $P(X, Y)$, data exploration is enough to infer the association rule $X \rightarrow Y$, there is no need to use a data mining algorithm for association rules extraction in such a case.

3 Conclusion and Future Work

We have used this result while analyzing the data of the course “Programming 1” in our university, see [1]. The associations found show that a group of students emerges that keep doing self-evaluation exercises during the semester. A future work is to continue conducting case studies with courses taught in different topics and designed in different ways to further enhance our catalogue of questions that can be interesting for teachers, and investigating further connections between data exploration and data mining.

References

- [1] Krueger A., Merceron, A., Wolf, B. A Data Model to Ease Analysis and Mining of Educational Data. *Third International Conference on Educational Data Mining*, 2010. Pittsburgh, USA.
- [2] Merceron, A., Yacef, K. Interestingness Measures for Association Rules in Educational Data. *First International Conference on Educational Data Mining*, 2008. Montreal, Canada, p. 57-66.

AutoJoin: Generalizing an Example into an EDM query

Jack Mostow and Bao Hong (Lucas) Tan
mostow@cs.cmu.edu, btan@andrew.cmu.edu

Project LISTEN, School of Computer Science, Carnegie Mellon University

Abstract. This paper describes an implemented method to generalize an example tutor interaction into a query to retrieve similarly related sets of events. It infers *WHERE* clauses to equate repeated values unlikely to match by accident.

The Session Browser [1] shown in **Figure 1** is an EDM tool to view data retrieved by querying a database of events logged by a tutor. It displays retrieved events in a context tree of enclosing events, with a 1-line summary of the database record for each event.

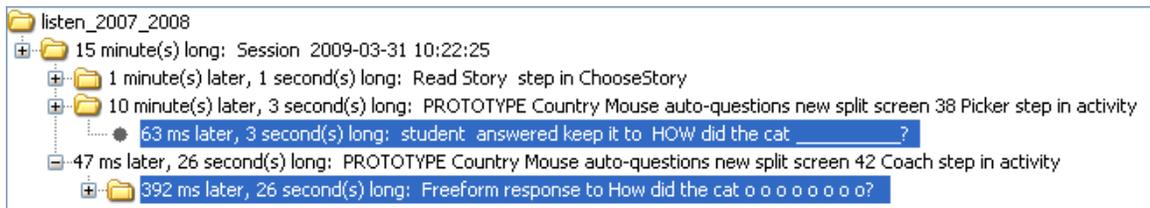


Figure 1: Event context tree highlighting two events selected by the user to AutoJoin

This brief example occurred in an activity to teach children to ask themselves questions about the text they read. The first highlighted event summarizes a child’s multiple-choice response to a prompt to fill in the rest of a question about the text. The second event describes the child’s spoken response to a prompt to speak the completed question aloud.

Often we want a query to retrieve examples similar to a current example. Complex queries are hard to construct, so we developed AutoJoin to generate them automatically. AutoJoin generalizes the two highlighted events into a query that finds “similar” cases, in this case a multiple choice step immediately followed by a free-form response step:

```
SELECT * FROM
  multiple_choice_question mcq,
  sentence_encounter se
WHERE mcq.activity_directory = se.activity_directory
      /* '..\data\stories\PROTOTYPE Country Mouse auto-questions new
split screen' */
AND mcq.end_time = se.step_start_time /* '20090331103359' (03/31/2009
10:33:59 AM) */
AND mcq.machine_name = se.machine_name /* 'LISTEN07-211' */
AND mcq.story_encounter_start_time = se.story_encounter_start_time
      /* '20090331102336' (03/31/2009 10:23:36 AM) */
AND mcq.user_id = se.user_id /* 'mKJ9-5-2001-07-23' */;
```

(The example here is simple for brevity; AutoJoin can generalize from more events too.)

AutoJoin constructs a query as a join of the tables where the selected events were logged. It abbreviates each table by its initials, e.g. `multiple_choice_question` as `mcq` and infers **WHERE** clauses by equating field values that show up more than once in those events. The comments, in green, show which values it abstracted into variables; they help the user understand the query and fix over-generalizations easily by uncommenting.

AutoJoin assumes that (1) unlikely matches matter, but (2) their specific values do not, so it (1) infers that the variables must be equal, but (2) abstracts away their specific values. When (1) is wrong, it under-generalizes; when (2) is wrong, it over-generalizes. To avoid under-generalizing, it uses heuristics to prevent meaningless matches. It gauges the likelihood of accidental match from how often the matching value occurs in each field. For instance, NULL values are frequent in many fields due to various reasons, one of which is its use as a no-value indicator. Thus NULL values in two such fields are likely to match by accident. In contrast, data types such as strings and dates have a large range of possible values. Assuming that they are unlikely to match by accident saves time by not bothering to estimate the frequency of the matching value in the two table columns.

Computing how often a given value occurs in a table column takes time for a large table, so AutoJoin estimates it from a sample of 400 rows. This sample is fast to retrieve but may be unrepresentative, especially if the column is sorted. Sampling blocks of 20 rows at 20 randomly chosen offsets is more reliable, but slow enough to be worth caching.

Another heuristic to avoid under-generalizing classifies certain columns as “non-cross-match” and compares them only with columns of the same name in other tables. For example, the table `story_encounter` has non-cross-match column `story_count`, and `sentence_encounter` has non-cross-match columns `sentence_count` and `word_count`. Since `story_count`, `sentence_count`, and `word_count` count different types of things, we assume it does not make sense to compare them. In contrast, comparing `start_time` and `end_time` from different tables does make sense.

The “non-cross-match” heuristic eliminates additional spurious matches, but relies on the naming convention it exploits, and on the user’s knowledge of which fields make sense to compare. In contrast, estimating the frequency of matching values does not depend on column names or knowledge of the database schema, so it’s a more general method.

We have identified the EDM task of generalizing from a single example of tutorial interaction, described AutoJoin’s simple but powerful heuristics, and reported their implementation in the Session Browser. By systematically generating clauses we might omit or mistype, it helps us build complex joins much faster than by hand. It’s limited in what it can notice, and it sometimes generalizes accidental matches, so we check its output, but checking is faster than writing queries. AutoJoin may also apply to many non-EDM domains. It seems too simple to be novel, but we have not found it elsewhere.

Acknowledgements: This work was supported by the Institute of Education Sciences, U.S. Department of Education, through Grant R305B070458, R305A080157, and R305A080628 to Carnegie Mellon University. The opinions expressed are those of the authors and do not necessarily represent the views of the Institute or the U.S. Department of Education. We thank Session Browser users for testing AutoJoin, and the educators, students, and LISTENers who helped generate, collect, and analyze our data.

References (also see Publications page at www.cs.cmu.edu/~listen)

- [1] Mostow, J., J. Beck, A. Cuneo, E. Gouvea, C. Heiner, and O. Juarez. Lessons from Project LISTEN's Session Browser. In C. Romero, et al., Editors, *Handbook of Educational Data Mining*. Taylor & Francis Group, in press.

Conceptualizing Procedural Knowledge Targeted at Students of Different Skill Levels

Martin Možina, Matej Guid, Aleksander Sadikov, Vida Groznik, Jana Krivec, and Ivan Bratko

Contact email: martin.mozina@fri.uni-lj.si

University of Ljubljana, Faculty of Computer and Information Science, Slovenia

Abstract. Conceptualizing procedural knowledge is one of the most challenging tasks of building systems for intelligent tutoring. We present a novel algorithm that enables teachers to accomplish this task (semi)automatically. Furthermore, it is desired to adapt the level of conceptualization to the skill level of particular students. We argue that our algorithm facilitates such adaptation in a straightforward fashion. We demonstrate this feature of the algorithm with a case study.

1 Conceptualization of Procedural Knowledge

In symbolic problem solving domains (like physics, mathematics, or games like chess), a particular domain is defined with a basic domain theory and a solution to be achieved. The task is to find a sequence of steps that bring us from the beginning state of the problem (definition of the problem) to the goal state (the solution). The basic domain theory (or basic declarative knowledge of the domain) is usually simple and easy to remember and, in principle, sufficient for solving problems; e.g. knowing rules of chess could in theory enable optimal play. However, finding a solution using only declarative knowledge would require far too extensive searching. A human student is incapable of searching very deep, therefore we need to teach him also the procedural knowledge – how to solve problems.

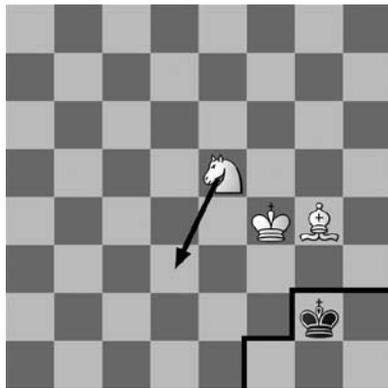
The “complete” procedural knowledge would be a function mapping from each problem state to an action that leads to the solution. For example, in chess endgames a tablebase specifies best moves for all possible positions. Tablebases can be used easily because they only require trivial amount of search. But now the problem is the space complexity – it is impossible for humans to memorize such tablebases that typically contain millions of positions. There is a way, however, that enables humans to solve problems in such chess endgames quite comfortably. Humans use some intermediate representation of the domain that lies between the basic domain theory and the “complete” procedural knowledge. We call such an intermediate representation a “conceptualized domain”.

We propose a goal-oriented conceptualization of domains. A goal-oriented rule has the following structure: *IF preconditions THEN goal (depth)*. The rule's *preconditions* and *goal* are both expressed in terms of attributes used for describing states. The term *preconditions* specifies applicability of the rule, while a *goal* specifies the values of attributes in the state to be achieved. The *depth* property of a rule is the maximum allowed number of steps in achieving the *goal*. We developed an interactive algorithm that combines specialized minimax search with the ABML principle [2] for (semi) automatic construction of such rules, where the teacher and the algorithm in turns improve the model. The *depth* parameter is set prior to learning and can be used to dictate

the complexity of learned rules. A higher *depth* will result in fewer rules with more general goals and vice-versa. Due to space limitations, we will skip the description of the algorithm (for details see [1]) and demonstrate its idea on a case study.

2 Case Study: KBNK Chess Endgame

KBNK (king, bishop, and knight vs. a lone king) is regarded as the most difficult of the elementary chess endgames. Most books mention only a basic strategy, however, it is hardly enough for successfully checkmating the opponent. Our aim was to conceptualize procedural knowledge in this domain for chess players at club level. Our chess teacher evaluated that they are able to calculate chess variations about 3 moves (6 plies) ahead.



Computer: “I suggest the following goal: the distance between black king and the edge of the board should decrease. However, it does not seem to work well in this position. *What goal would you suggest for white in this position? What are the reasons for this goal to apply in this position?*”

The teacher gave the following answer: “Pushing black king to the edge of the board is fine. However, I find the following goal to be more instructive for the student: *Build a barrier and squeeze the defending king into the corner. Currently such barrier is not yet established.* The move expected from the student is 1.Ne5-d3 achieving the goal.”

Figure 1: Interaction between computer and teacher: explanation of a critical example.

An example interaction between the method and the teacher is shown in Fig. 1. The teacher is presented with a critical example, i. e., the example where the current set of rules suggested a bad goal (“push black king to the edge of the board” can be achieved, but is not leading to solution). The teacher was therefore asked to provide a better goal for this position, which was then used in the construction of a new set of goal-based rules. The process was completed when all critical examples were explained by the expert.

The final rules¹ were presented to three chess teachers (among them a selector of Slovenian women's squad and a selector of Slovenian youth squad) to evaluate their appropriateness for teaching chess-players. They all agreed on the usefulness of the presented concepts and found the derived strategy suitable for educational purposes at the level targeted for. Among the reasons to support this assessment was that the instructions “clearly demonstrate the intermediate subgoals of delivering checkmate.”

References

- [1] Možina M., Guid M., Sadikov A., Bratko I.: Goal-Based Rule Learning. Technical report, University of Ljubljana, 2009, <http://www.ailab.si/matej/KBNK/GBRL.pdf>.
- [2] Možina, M., Žabkar, J., Bratko, I. Argument based machine learning. *Artificial Intelligence*, 2007, 171(10-15), pp. 922-937.

¹The complete rule-based model for KBNK and example games containing automatically generated instructions can be found in a web appendix at <http://www.ailab.si/matej/KBNK/>.

Data Reduction Methods Applied to Understanding Complex Learning Hypotheses

Philip I. Pavlik Jr.

ppavlik@andrew.cmu.edu

Human Computer Interaction Institute, Carnegie Mellon University

Abstract. Modern learning science researchers are facing a flood of data as it becomes easier and easier to collect multiple streams of information from students before, during, and after learning experiments. While oftentimes these experiments do experimentally manipulate specific variables to improve responses on a posttest, these experiments are also interested in how the many related student factors explain who responds to the treatment and why. This poster introduces a recent experiment and explains how the data were analyzed using a combination of exploratory factor analysis (using SPSS) and exploratory structural equation modeling (using Tetrad) to partially refute a theoretical hypothesis and reveal a new explanation for further testing.

1 Introduction

A recent line of experimentation with the FaCT System [1] for vocabulary drill practice in Chinese is focusing on how metacognitive and motivational factors may be important in various aspects of student learning. Specifically, this line of research is trying to dig deeper into various reliable correlations that were uncovered in earlier work. In this earlier work students were asked for self reports of strategy use (e.g. forming verbal linkages rather than just repeating items). A consistent finding in 2 experiments was that students in the less difficult conditions (higher percent correct) reported using more of these verbal strategy links rather than using a more repetitive strategy [2]. My initial hypothesis was that this pattern of results suggested that higher percent correct (more review practice of the same flashcards) reduces the cognitive load on the user, thereby allowing the individual to better use learning strategies. Since strategy use for learning paired-associates has often been a successful way to improve learning [3], I wanted to test whether the process based explanation above was correct, since if it should be supported, it would indicate that any further ways we could decrease cognitive load might encourage further strategy use by students thereby leading to further learning gains.

The data we collected before the study included the full Motivated Strategies for Learning Questionnaire (MSLQ). During the vocabulary practice we gathered all data on recall and performance, and also surveyed students on a 5-point Likert scale to ask them to rate how difficult practice was recently, how useful practice was recently, and how much they were able to use strategies for learning recently. Further, during this middle portion they were randomized into either easier or more difficult practice conditions (more or less review). Following practice they responded to a 32-item questionnaire that specifically addressed their reactions to the practice sessions.

2 Analysis and Results

Analysis began with exploratory factor analysis on both questionnaires. Because I had a limited sample size (49 students) it was clear that we would not be able to meaningfully

discover all the MSLQ factors found in prior research [4], which suggests that 15 subscales can be differentiated from the 81 items. I used principal components FA (SPSS factor analysis procedure) with oblimin rotation (which allows correlated factors to be found) to initially reduce the MSLQ data, and then maximum likelihood FA (which produces fit statistics) to find a best fitting model of the MSLQ factors that included mostly high communality items. Four factors were found. Data from the post-questionnaire was similarly reduced to 3 factors. The factor scores for the 4 MSLQ and 3 post-questionnaire along with all the other relevant data (performance data, Likert survey averages during practice, difficulty condition the student was in) were entered into the GES (greedy eliminative search) algorithm in Tetrad software program. This particular structural equation modeling (SEM) algorithm was ideal for my purposes because it provided a theoretically unmotivated way to search for a best way to reduce the patterns of the data into something understandable. Our hypothetical model (easiness leads to strategies) which required only a single edge from difficulty during learning to strategies during learning fit more poorly ($d.f. = 81$, $X^2 = 87.6$, $p = 0.298$) than an identical model with no requirements on this edge in the model ($d.f. = 80$, $X^2 = 81.83$, $p = 0.422$).

The SEM results in the poster will show the differences between the competing models and also illustrate the power of this approach for drawing hypotheses from large sets of poorly organized data or from data without strong preexisting theoretical relations. In particular this sort of analysis reveals one way contextual data about student data in repositories such as DataShop can be used to make conclusions about learning.

Acknowledgements

This research was supported by Ronald Zdrojkowski and was also made possible with the assistance and funding of the Pittsburgh Science of Learning Center (NSF-SBE) #0836012.

References

- [1] Pavlik Jr., P.I., Presson, N., Hora, D.: Using the FaCT System (Fact and Concept Training System) for Classroom and Laboratory Experiments. *Inter-Science Of Learning Center Conference*, 2008. Pittsburgh, PA,
- [2] Pavlik Jr., P.I.: The microeconomics of learning: Optimizing paired-associate memory. *Dissertation Abstracts International: Section B: The Sciences and Engineering*, 2005, 66(10-B), p. 5704.
- [3] Atkinson, R.C., Raugh, M.R.: An application of the mnemonic keyword method to the acquisition of a Russian vocabulary. *Journal of Experimental Psychology: Human Learning & Memory*, 1975, 1(2), p. 126-133.
- [4] Artino, A.R., Jr.: Review of the Motivated Strategies for Learning Questionnaire. 2005.

Analysis of a causal modeling approach: a case study with an educational intervention

Dovan Rai and Joseph E. Beck
 {dovan, josephbeck}@wpi.edu
 Computer Science Department, Worcester Polytechnic Institute

1 Introduction

This paper explores the application of causal models to understanding data generated from a computer tutor. Mily's World (<http://users.wpi.edu/~dovan/coordinates.html>) is a flash-based learning environment for coordinate geometry and featuring *game-like properties* such as a cover story and pictures. We were primarily interested in what class of students benefitted from this type of intervention, as well as which students preferred this style of instruction to traditional materials. Fifty eight students used the tutor and we collected survey data and their log records from the tutor. We analyzed the data using Tetrad (<http://www.phil.cmu.edu/projects/tetrad>), free software designed for causal modeling.

2 Causal model search

We used the PC algorithm in Tetrad, which is designed to search for causal explanations of observational or mixed observational and experimental data. The causal model thus obtained is shown in Figure 1 where the rectangular nodes are the data of each student.



Figure 1 Search Model: causal model after making PC search

Based on our domain knowledge, we hand-crafted a causal model shown in Figure 2, where we added four latent nodes (oval nodes) that we believe are influencing the observables.



Figure 2 Hand-crafted model (latent variables are ovals)

We also generated a correlation graph where a link between two nodes indicates they are reliably correlated with $P < 0.05$. We then compared the PC causal model and the correlation graph with our hand-crafted model. For each link in the hand-crafted model, if the automatically generated model had it it was a true positive; if the link was missing it was a false negative. Similarly, if the model has correctly identified the absence of link, that would be a true negative. The causal model is more stringent than correlation graph as it would put a

link between nodes only if they retain their association after controlling for all other nodes. We found that the correlation graph has more false positives whereas the causal model is more susceptible to false negatives.

2.1 True positive with correct direction and true negatives

By exploiting conditional dependencies, the PC model correctly identifies true positives with correct direction (LikeStory&picturesOfMily → likeMily ← realWorld ExamplesHelpful) and true negatives (link likeMath–avgHints is gone once controlled for “% correct”). This ability to automatically partial out other influences is difficult, at best, to replicate in traditional statistics packages.

2.2 False negatives: weaker statistical power due to small sample size

When we have small sample size, doing partial correlations can give false negatives due to limited statistical power. Having more samples reduces false negatives without adding false positives. Multicollinearity is an extreme case, where we might falsely conclude that there is no linear relationship between an independent and a dependent variable. For example: picturesHelpfulForMath is correlated with both likeMily and LikeStory&picturesOfMily. But since, likeMily and LikeStory&picturesOfMily are highly correlated between themselves (.471**), picturesHelpfulForMath is conditionally independent to both of them (see Figure 1).

2.3 Search with domain knowledge

To overcome the problem of multiple “Markov equivalent” graphs that can be built from the same data, we add domain knowledge to direct our search to pick the most compatible model.

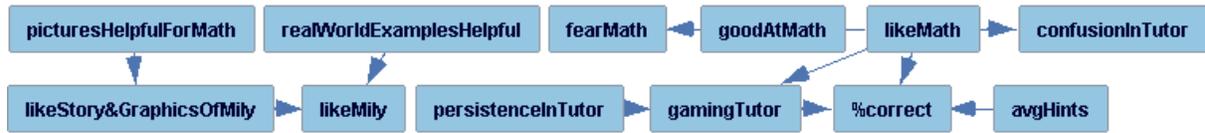


Figure 3 Causal model with domain knowledge

We see from Figure 2 and Figure 3 that adding domain knowledge not only fixes the arrow orientations (likeMath → %correct), but also adds new edges such as likeMath → gamingTutor. One interesting finding is that adding domain knowledge has fixed the problem of multicollinearity (picturesHelpfulForMath → LikeStory&picturesOfMily) as adding temporal knowledge restricts nodes to only influence things which occurred later.

3 Conclusions

In this paper, we have presented a case study of applying causal modeling, using the Tetrad software, to understand what factors influence how students respond to our educational intervention. We found that a problem that arises from having a small sample results in more false negatives in our causal model. That is, there are true relationships that we lack the statistical power to detect. We also found that by adding domain knowledge, we are not only able to correct the arrow orientations but we can also overcome issues such as multicollinearity to come up with the most plausible model from the set of equivalent models.

Peer Production of Online Learning Resources: A Social Network Analysis¹

Beijie Xu and Mimi M. Recker

beijie.xu@aggiemail.usu.edu, mimi.recker@usu.edu

Department of Instructional Technology & Learning Sciences, Utah State University

Abstract. This paper describes methods for collecting user activity data in a peer production educational system, the Instructional Architect (IA), and then takes a social network perspective in analyzing these data. In particular, rather than focusing on content produced, it focuses on the relationship between users (teachers), and how they can be analyzed to identify important users and like-minded user groups. Our analyses and results provide an example for how to select the most important factors in analyzing the dynamics of an online peer production community using social network analysis metrics, such as in-degree, out-degree, betweenness, clique, and community.

1 Introduction

The increased pervasiveness of networked computing coupled with a vibrant participatory web culture has spawned new models of innovation and creation. In education, the scalable deployment of media-rich online resources supports peer production in ways that promise to radically transform teaching and learning. Recent research, however suggests these peer production models may only succeed when they are aimed at focused tasks, coupled with incentives to harness the work of the best collaborators. More is not simply better, and for educational peer production models to succeed, we need more nuanced understandings of how people participate in such environments to efficiently and effectively collaborate around learning resources.

Social network analysis (SNA) is a well-established method for studying interactions among human organizations [1]. It has also been applied in educational research. In particular, patterns of social relationship revealed by SNA, coupled with results from other qualitative evaluation methods such as content analysis, interviews, survey, reports, and sociometry, are frequently used in longitudinal study of the participatory aspects of computer-supported collaborative learning (CSCL).

In our own work, we have developed a simple, web-based authoring tool, called the Instructional Architect ([IA.usu.edu](http://ia.usu.edu)), which supports teacher peer production. In this study, we examine the teacher users in the IA system and conduct a social network analysis to begin to characterize teachers' networked relationships.

2 The Instructional Architect and its Social Networks

The Instructional Architect allows teachers to freely find, gather, and produce instructional activities for their students using online learning resources. Teachers can share these resulting activities, called *IA projects*, by making them publically available on

¹ For full paper, graphs, and references, please visit <http://edm.usu.edu/publications/sna.pdf>

the Web. These IA projects can then be *viewed*, *copied*, and *adapted* by other IA users, in ways that support innovative teacher peer production.

For each registered user, we determined the networks between users based on the following two pairs of relationships: 1) user A *viewed* user B's IA project, and user B's IA project was *viewed* by user A, 2) user A *copied* user B's IA project, and user B's IA project was *copied* by user A. Thus, the vertices in each network represent IA users, and the link directions and values represent the number of viewer/viewed or copier/copied actions between two users. These two networks (termed *viewer* network and *copier* network respectively) are represented as weighted, directed graphs.

3 Data Analysis

The present study consists of the view and copy actions occurring between September 2008 and February 2010. The view and copy networks were represented within the freely-available SNA software Visone, which also computes key SNA measures for each network. The graph of viewer network is much denser than the copier network. From a user perspective, viewing represents an action with a much lower “cognitive” cost (a simple click) compared to a copy action (which represents a decision to use/adapt the content). Not surprisingly, this difference is reflected in the number of participating users and the density of the two networks.

We studied the relationship between user production of IA projects, and viewing and copying actions. Users with a large number of views are not necessarily those who create a large number of IA projects. Conversely, the mean number of IA projects created does not saturate and exhibits an increasing trend as the function of copy action. Thus, the number of copies is a more accurate signal than the number of views in estimating project creation magnitude, serving as a better metric for describing meaningful user's activity within the IA network.

Finally, we applied a *clique* analysis on the copy network – the more important network of the two. A *clique* is a subgraph in a network in which every two vertices are connected by an edge. When the number of vertices in such a subgraph is k , it is called a *k-clique*. A clique represents closely tied subset of the network. A *k-clique-community* is defined as the union of all k -cliques that can be reached from each other through a series of adjacent k -cliques. We detected 11 k -cliques inside the copy network. These cliques suggest that some small subsets of users share common interests such that they could make use of each other's IA projects. The largest community in the copy network is a 6-clique-community formed by four adjacent 3-cliques. Since this community represents a closely tied subset of the copy network, not surprisingly, all six users teach the same subject area – language arts, and five of them teach both math and science, and four of them teach social studies. In sum, the clique analysis helped identify teachers with shared interests.

References

- [1] Knoke, D. & Yang, S. *Social Network Analysis*, 2008. London: Sage.

Class Association Rule Mining from Students' Test Data

Cristóbal Romero¹, Sebastián Ventura¹, Ekaterina Vasilyeva² and Mykola Pechenizkiy²
cromero@uco.es, sventura@uco.es, e.vasilyeva@tue.nl, m.pechenizkiy@tue.nl

¹Department of Computer Science, Córdoba University, Córdoba, Spain

²Department of Computer Science, Eindhoven University of Technology, The Netherlands

Abstract. In this paper we propose the use of a special type of association rules mining for discovering interesting relationships from the students' test data collected in our case with Moodle learning management system (LMS). Particularly, we apply Class Association Rule (CAR) mining to different data matrices such as the *score-matrix*, the *relationship-matrix* and the *knowledge-matrix*. These matrices are constructed based on the data relate to students' performance in the test and on the domain knowledge provided by the instructor. We describe how to obtain these matrices and then we have applied a CAR mining algorithm.

1 Introduction

A Class Association Rule (CAR) is a special type of Association Rule (AR) that describes an implicative co-occurring relationship between a set of items and a pre-defined class, expressed in the form of an “IF antecedent (input-attributes) THEN consequent (class)” rule [1]. AR mining finds all rules that satisfy some minimum support and minimum confidence constraints, that is, the target of mining is not predetermined. However, in CAR mining there is one and only one pre-determined target, i.e., the class. So, CAR is a type of target-constraint association rule. Such kind of *focused* rules mining results in a set of independent and comprehensible rules having one (desired) element in the consequent. Such rules usually represent discovered knowledge at a high level of abstraction and can be used directly in the decision making process.

Modern assessment tools and testing systems in particular allow accumulating a lot of useful performance and usage related data (possibly at different levels of granularity). This may include (but is not limited to) actual students' answers and their correctness, final scores, used/execution time (total and for each question) and some statistics about items/questions such as Facility Index or % Correct (F.I.), Standard Deviation (S.D.), Discrimination Index (D.I.), etc. Although in many cases, still only the final scores are used by an instructor to evaluate students' knowledge or performance [2], with recent developments in educational data mining research different ideas for intelligent analysis of assessment data were proposed. In this paper, we show the potential utility of applying CAR mining over the test-related data for providing an instructor interesting relationships discovered from these data presented in the *score-matrix* and *knowledge-matrix*. This information can be turned by an instructor into valuable knowledge for making decision on how to improve both the test and the course.

2 Experimentation

We have used data collected with Moodle's quiz module tool. Starting from these data and some background information provided by the instructor we have created three different data matrices (see Figure 1).

SCORE-MATRIX							RELATIONSHIPS-MATRIX					KNOWLEDGE-MATRIX				
	Item1	Item2	Itemj	Time	Score	Item1	Item2	Itemj	Concept1	...	Conceptk	Score
Student1	S11	...						R11	...				K11			
Student2				
...																
...								
Studenti					...	Sij						Rkj			Kik	

Figure 1. Matrices created from test's data.

We have applied the Apriori-CAR mining algorithm over the previously described data matrices. In the first experiment, we have used the *score-matrix*, and we have selected/filtered as input-attributes (antecedent) only the item answers, and as class the final score. In this way, we can see the relationships between items and how they can predict/determine the final score obtained by students. In the second experiment, we have used the *knowledge-matrix*, and we have selected the knowledge of concepts as input-attributes (antecedent) and the final score as a class attribute. In this way, we can discover the relationships between concepts and between the level of knowledge of these concepts and the final score obtained by students.

3 Conclusions and Future Work

In this paper, we proposed to use a special type of association rules over the assessment data in a particular scenario. We mined different test data matrices rather than only the typical score-matrix. Particularly we used an item-concept relationship matrix created by the instructor and a student-concept knowledge level matrix automatically created based on the information from the other two matrices. Finally, it is important to notice that concepts themselves may need to be presented as a hierarchy rather than a 'flat' set of independent concepts. Mining interesting patterns in such settings is one of the directions of our further work.

References

- [1] Liu, B., Hsu, W, Ma, Y. Integrating classification and association rule mining. *Int. Conference on Knowledge Discovery and Data Mining, KDD'98*, pp. 80-86.
- [2] Romero, C., Ventura, S., Salcines, E. Data mining in course management systems: Moodle case study and tutorial. *Computer & Education*, 2008, 51(1), pp. 368-384.

Modeling Learning Trajectories with Epistemic Network Analysis: A Simulation-based Investigation of a Novel Analytic Method for Epistemic Games

André A. Rupp, Shauna J. Sweet, & Younyoung Choi
ruppandr@umd.edu
EDMS Department, University of Maryland

Epistemic games have been developed in recent years to help players develop domain-specific expertise that characterizes how professionals in a particular domain reason, communicate, and act [1]. For example, learners may learn what it is like to think and act like journalists, artists, business managers, or engineers by using digital learning technologies to solve realistic complex performance tasks. This is accomplished by designing the game in such a way that completing it mimics the core experiences that learners outside the gaming environment would have in a *professional practicum* in the field. As one might expect, traditional measurement models with latent variables designed for traditional large-scale assessments struggle to jointly accommodate the complexities of the data that arise from these games. Thus, there are currently no off-the-shelf statistical models that can be applied directly to epistemic games to satisfy the desired scaling and reporting purposes; alternative modeling approaches grounded in non-parametric methods appear to be more promising in this regard.

In this poster, we report on a comprehensive simulation study for investigating one candidate method that has recently been proposed in the literature called *epistemic network analysis* (ENA) [3]. The method is purely descriptive at this point and has been applied to real data collected in several different epistemic games. However, it has not been thoroughly investigated using simulation studies that use conditions representing a wide variety of realistic game-play scenarios. In our work we specifically investigate the sensitivity of different ENA statistics to capturing the different learning trajectories of players who play different types of epistemic games.

In order to simulate data we used principles from modern latent variable models, specifically models in *item response theory* (IRT) and *diagnostic classification models* (DCMs) [2]. In these models, contributions of learner and task characteristics to response probabilities are statistically separated by specifying separable parameters for each. The following table provides an overview of the simulation design for our study. Note that we are specifically investigating various ENA outcome statistics on different metrics. For example, we are using the raw ENA statistics as well as the percentage overlap of empirical confidence bands, computed using the 100 replications, across the entire game play of ENA statistics. The latter approach in particular provides us with a non-parametric approach for sorting learners according to their learning trajectory profiles that can be adapted for real-data analyses.

Factor	Level Specification	# of Levels
Learner Characteristics		
Type of trajectory	Linear, curvilinear, piecewise	40
Mixture of trajectory	Yes, no	2
General Game Characteristics		
# of Slices / Tasks	90, 120	2
# of Skills / SKIVE elements	5, 15	12
Different Reference Patterns	Low, high	2
Specific Task Characteristics		
Task difficulty	Varying from low to high	7
Task specificity	Low, moderate, high	3
Game structure / Q-matrix	Constant vs. varying complexity and skill focus	4
Total # of Conditions		161,280
Number of Replications per Condition		100
Total # of Computations		161,280,000

The simulation design further helps us to quantify the relative influence of different design factors on the variation in the raw ENA statistics or secondary derived statistics such as percentage overlap of ENA statistics. For example, the following table shows how most of the variation in one global statistic, the weighted density, is accounted for by the similarity of the underlying learning trajectories independent of the different game conditions as captured by the task parameters.

Factorial ANOVA Results using Deep Structure Similarity Coding

Source		Type III SS	df	η^2
Main Effects	Similarity	285.78	2	33.74
	Difficulty	.38	2	.05
	Specificity	1.42	6	.17
Interaction Effects	Similarity*Difficulty	.94	4	.11
	Similarity*Specificity	3.97	12	.47
	Difficulty*Specificity	.26	12	.03
Residual		554.32	6786	65.44

The research is currently ongoing with the goal of having completed results submitted for publication by the end of the summer.

References

- [1] Bagley, E., & Shaffer, D. W. (2009). When people get in the way: Promoting civic thinking through epistemic gameplay. *International Journal of Gaming and Computer-mediated Simulations*, 1, 36-52.
- [2] Rupp, A. A., & Templin, J. (2008). Unique characteristics of diagnostic classification models: A comprehensive review of the current state-of-the-art. *Measurement: Interdisciplinary Research and Perspectives*, 6, 219-262.
- [3] Rupp, A. A., Gushta, M., Mislevy, R. J., & Shaffer, D. W. (2010). Evidence-centered design of epistemic games: Measurement principles for complex learning environments. *Journal of Technology, Learning, and Assessment*, 8(4). Available online at <http://escholarship.bc.edu/jtla/vol8/4>

Multiple Test Forms Construction based on Bees Algorithm

Pokpong Songmuang and Maomi Ueno
{pokpong, ueno}@ai.is.uec.ac.jp

Graduate School of Information Systems, the University of Electro-Communications

Abstract. This paper proposes a new construction method of multiple test forms that applies a Bees Algorithm and a parallel computation technique to improve the computational costs of the traditional methods.

1 Introduction

Educational assessments occasionally need “multiple test forms” in which each form consists of a different set of items but still has qualities that are equivalent to the others. In order to construct multiple test forms, e-testing, which accomplishes automated test construction, has recently become popular in research areas involving educational measurement. In order to construct multiple test forms, several methods have been proposed to construct all forms of a test to satisfy the same test constraints to ensure that all forms have equivalent qualities. However, the main problem with traditional methods of constructing multiple test forms to maximize the equivalent between test forms is the trade-off between the fitting errors to test constraints and computational costs. The main purpose of the research discussed in this paper is to solve this problem.

2 Method of Constructing Multiple Test Forms Based on Bees Algorithm in Parallel Computing

The main idea behind this research is to alleviate the trade-off by applying a parallel-computing technique that divided the computational costs between multiple processors without increasing the differences in fitting errors.

It is possible to effectively install a parallel-computing technique into random-search methods. Several studies have used random-search methods in parallel computing to solve optimization problems [1] [2].

Furthermore, some studies have compared the efficiencies of random-search algorithms [3] [4]. The results of these studies revealed Bees Algorithm (BA) performed the high performances in searching for optimal solutions. Moreover, the problems in [3] [4] and the multiple test construction are combinatorial optimization problems which are classified as NP-hard as the multiple test construction. Therefore, we propose a method of constructing multiple test forms based on BA in parallel computing to alleviate the trade-off between computational costs and differences in fitting errors. This method constructs multiple equivalent test forms by minimizing the difference in fitting errors between test forms.

The construction of multiple test forms has a time complexity of $O(c \cdot m! \cdot 2^f)$, where c is the number of test constraints, m is the number of items in an item bank, and f is the number of constructed test forms that satisfy all test constraints. To reduce the computational time, we divided the construction of test forms into two steps: Step A: Construct test forms only to minimize the fitting errors of each form to test constraints without taking into consideration the equivalence of test forms. Therefore, the

time complexity of this step is $O(c \cdot m!)$. Here, the constructed test forms are still not equivalent.

Step B: Extract the most equivalent set of test forms from the constructed test forms in Step A that minimizes the difference in fitting errors between test forms. The time complexity in this step is $O(2^f)$.

The time complexity for constructing test forms is reduced from $O(c \cdot m! \cdot 2^f)$ to $O(c \cdot m! + 2^f)$. The BA is applied as a search algorithm to both steps.

3 Evaluation

We compared the proposed method (BA) with the Big-Shadows Test (BST) method [5] and a genetic algorithm for multiple test construction (GA) to demonstrate its accuracy and speed in constructing multiple test forms. BA, BST, and GA construct multiple test forms to minimize the fitting errors indicted by the differences between the expected test information function and the test information function of the constructed test forms at five levels of abilities and to minimize the difference in the fitting errors. Before constructing test forms, we define the target values of the expected test information function. We used three actual item banks from the Japan Information Technology Engineers' Examination that had total numbers of items of 517, 978, and 2385. The total numbers of test constraints corresponding to each item bank were 32, 57, and 112 and the total numbers of test items were 20, 50, and 80. The results obtained from this experiment indicate that the proposed method improves the traditional construction of multiple test forms.

References

- [1] Borovska, P. Solving the travelling salesman problem in parallel by genetic algorithm on multicomputer cluster, *Proceedings of International Conference on Computer Systems and Technologies*, 2006. pp. II.11–1–II.11–6.
- [2] He, K., Zheng, L., Dong, S., Tang, L., Wu, J., Zheng, C. PGO: A parallel computing platform for global optimization based on genetic algorithm, *Computers & Geosciences*, 2007, 33(3), p. 357–366.
- [3] Pham, D.T., Ghanbarzadeh, A., Koc, E. Otri, S., Rahim, S., Zaidi, M. The bees algorithm, a novel tool for complex optimisation problems, *Proceedings of the 2nd Int Virtual Conf on Intelligent Production Machines and Systems*. 2006, pp. 454–461.
- [4] Wong, L. P., Low, M. Y. H., Chong, C. S. A bee colony optimization algorithm for traveling salesman problem, *Proceedings of the 2008 Second Asia International Conference on Modelling & Simulation (AMS)*. 2008, p. 818–823.
- [5] van der Linden, W. J. *Linear Models for Optimal Test Design*, 1st ed., ser. Statistics for Social and Behavioral Sciences: Springer, 2005.

Can order of access to learning resources predict success ?

Hema Soundranayagam¹ and Kalina Yacef²

{hsou2856, kalina}@it.usyd.edu.au

¹School of Arts and Sciences, Australian Catholic University, Australia

²School of Information Technologies, University of Sydney, Australia

Abstract. Learning management systems capture student's interactions with the course contents in the form of event logs, including the order in which resources are accessed. We build on past research which indicates there are learning benefits if students determine their own ordering of use of learning materials. We report our exploration of sequential data mining that aims to help teachers determine whether some patterns of access to learning resources are predictive of performance, especially where this may signal the need for remediation. We report first explorations of the data in a graphics course and these indicate that sequence of resource access varied between the low, medium and high achieving student groups.

Introduction

The sequence in which learners should make use of learning materials is important in designing online courses. Learning management systems (LMS) provide one-size-fits-all solution where every student is presented with the same set of learning materials in one particular order. Previous research [1] indicates that the order in which the online learning materials are accessed may have an important relationship with student learning.

Our research study explores the order in which resources students access as they solve set assessment tasks, such as tests, assignments and Exams. The presentation order of the problems (for instance easy questions followed by difficult ones) in these activities determines the sequence in which resources will be assessed in order to solve them. For instance, in a study by Pardos and Heffernan [2], the relationship between the sequence of problem order and learning in Intelligent Tutoring systems was explored using Bayesian methods. We explore the use of data mining techniques to analyse patterns of such access. While Pardos and Heffernan explored the relationship between problem order and performance, we analyse the order of resource usage and its links with learning.

Approach

The context of our work is a graphics course delivered in mixed mode. Each week students complete activities after reviewing the resources online. The resources consisted of a comprehensive tutorial guide and additional video and text based tutorial guides. Skills acquired during the weekly activities are tested using a mid-semester test and final exam. Student log data for file accesses during each week are extracted. Students were grouped, based on their achievement on the mid-semester test into high, medium and low achievement groups. The patterns within each group were analysed to identify distinctive sequences associated with each group. A key goal of the approach is to identify sequences that are more frequent among the weak students since such patterns might be used as an "early warning sign" by instructors.

Our dataset comprises 66 students taking a Multimedia system course in Semester, 2010, using the Blackboard learning management system. As an indicator of initial knowledge, students completed an eighteen question self assessment questionnaire at the start of the semester. Event logs(1421) over two weeks were analysed to obtain the patterns.

Preliminary results: Distinctive patterns of access were found for each group and patterns showed that high and medium groups frequently accessed the compressive tutorial in order to complete their tasks successfully.

References

- [1] K. Scheiter and P. Gerjets, The impact of problem order: Sequencing problems as a strategy for improving one's performance, *Proceedings of the 24th Annual Conference of the Cognitive Science Society*, pp. pp. 798-803, 2002.
- [2] Z. Pardos and N. Heffernan, Determining the Significance of Item Order In Randomized Problem Sets, *The Second International Conference on Educational Data Mining 2009*.

A Data Driven Approach to the Discovery of Better Cognitive Models

Kenneth R. Koedinger¹ and John C. Stamper²

¹koediger@cmu.edu, ²john@stamper.org

²Human Computer Interaction Institute, Carnegie Mellon University

Abstract. Cognitive models composed of knowledge components are an integral part of intelligent tutors and drive many of the instructional decisions that these systems make. Most of these models are designed by educators and subject experts. Today vast amounts of data, collected from many intelligent tutors, allow us to analyze and improve the current cognitive models through educational data mining. In this research, we show how we identified, in the tutor data, potential improvements to existing cognitive models and then evaluated those improvements using statistical analysis and cross validation.

1 Introduction and Experiment

Educational data mining provides a great opportunity to discover better cognitive models. A correct cognitive model is one that is consistent with student behavior. It predicts task difficulty and transfer between instruction and test. Multiple algorithms have been developed for automated discovery of the attributes or factors that make up a cognitive model. This research provides the basis for an infrastructure for automatically applying such algorithms to data sets and discovering better cognitive models. We show how data analysis tools such as those in the PSLC DataShop [2] can be used to identify areas for improvement, and then discuss how to quantitatively evaluate the new models.

We tested a number of Knowledge Component (KC) models on the Geometry Area '96 data set (from DataShop), which indicate how well 59 students performed (i.e., how often they were correct without tutor help) on 139 unique geometry task items, that is, steps in multi-step problems presented by the Geometry Cognitive Tutor. Two measures of quality, the Bayesian Information Criteria (BIC) and the root mean squared error (RMSE) on the held-out test data in a 3-fold cross validation were calculated for each KC model. The KC models represent different ways of sorting the 139 items into groups that measure student acquisition of the same KC, like all items requiring application of the circle area formula. In all cases, predictions of student performance are based on Additive Factors Model (AFM). It is important to point out that both BIC and cross-validation adjust for over-fitting, which can result from unnecessary addition of model parameters. Thus, the differences between the models are likely to be of practical significance. A new model was discovered using the visualization and analysis tools provided by the PSLC DataShop. This model is better (lower BIC and lower RMSE) than the “original” production rule cognitive model in Geometry Cognitive Tutor, which was created by cognitive scientists and domain experts [3]. It is also better than any of the models discovered by our existing automated approach to KC model discovery called Learning Factors Analysis [1]. The DataShop tools show that most of the KCs have appropriate learning curves, like circle-area and trapezoid-area, where the error rate starts high and then goes down. The curve for compose-by-addition curve, however, is flat.

This kind of curve suggests that this KC may not be correctly defined – some task items it summarizes may require different knowledge than others. The compose-by-addition KC characterizes the step in complex geometry area problems where the student must find an irregular area, like the area of a sidewalk around a pool by subtracting the area of the pool from the whole area. Closer inspection of such problems reveals that some of them provide “scaffolding” by indicating that the student should find the component areas first (whole area and pool area) before finding the target irregular shape (the sidewalk) and other problems do not provide this hint. The new model distinguishes steps in such problems where such decomposition planning is required and ones where it is not. Using data to discover and confirm the existence of such “hidden” planning knowledge is an interesting cognitive science achievement – such a hidden or cognitive component is not directly apparent in student behavior, in contrast with the original compose-by-addition KC, which is adding or subtracting two numbers. It is also important for instructional design. We have used this insight to redesign this unit of the cognitive tutor to better help students acquire this difficult and important problem decomposition skill (such non-trivial problems are frequently seen on standardized tests). A ‘close-the-loop’ in vivo experiment is being run this spring to test whether these designs yield improved robust learning as compared to the existing tutor.

2 Conclusion and Future Work

This work demonstrates the use of tools in the DataShop to discover a better cognitive model, even in a domain (Geometry) where there has been considerable attention and prior cognitive analysis. The approach described here to discover cognitive models has a heavy component of human expertise. Using data to optimize cognitive models and improve instructional systems is a tremendous opportunity for EDM. The achievement will be greater to the extent that the discovered models involve deep or integrative KCs not directly apparent in surface task structure, like the problem decomposition skill we identified in Geometry. In addition, future work should compare the statistical model structure of competing discovery algorithms to shed new light on the nature or extent of regularities or laws of learning, like the power or exponential shape of learning curves and whether or not there are systematic individual differences in student learning rates.

References

- [1] Cen, H., Koedinger, K. R., & Junker, B. (2006). Learning Factors Analysis: A general method for cognitive model evaluation and improvement. In M. Ikeda, K. D. Ashley, T.-W. Chan (Eds.) *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, 164-175. Berlin: Springer-Verlag.
- [2] Koedinger, K.R., Baker, R.S.J.d., Cunningham, K., Skogsholm, A., Leber, B., Stamper, J. (2009) A Data Repository for the EDM community: The PSLC DataShop. In Romero, C., Ventura, S., Pechenizkiy, M., Baker, R.S.J.d. (Eds.) *Handbook of Educational Data Mining*. Boca Raton, FL: CRC Press.
- [3] Koedinger, K. R. & Alevan, V. (2007). Exploring the assistance dilemma in experiments with Cognitive Tutors. *Educational Psychology Review*, 19 (3): 239-264.

Using a Bayesian Knowledge Base for Hint Selection on Domain Specific Problems

John C. Stamper¹, Tiffany Barnes², and Marvin Croy³

¹john@stamper.org, ²tiffany.barnes@gmail.com, ³mjcroy@uncc.edu

¹Human Computer Interaction Institute, Carnegie Mellon University

²Department of Computer Science, University of North Carolina at Charlotte

³Department of Philosophy, University of North Carolina at Charlotte

Abstract. A Bayesian Knowledge Base is a generalization of traditional Bayesian Networks where nodes or groups of nodes have independence. In this paper we describe a method of generating a Bayesian Knowledge Base from a corpus of student problem attempt data in order to automatically generate hints for new students. We further show that using problem attempt data from systems used to teach propositional logic we could successfully use the created Bayesian Knowledge Base to solve other problems. Finally, we compare this method to our previous work using Markov Decision Processes to generate hints.

1 Introduction and Method

One goal of our research has been to provide help and feedback to students with as little human intervention as possible. Our hint generation methods using MDPs have been successful in fulfilling this goal [1], but are limited to individual problems with previously collected data. We believe that extracting information from multiple problem MDPs and create a corpus of knowledge that can be used to solve new problems would inherently incorporate student preferences and bias hints toward the rules that students actually use. To accomplish this solution we extracted individual model components (MCs) from each of the MDPs to create a large Bayesian Knowledge Base (BKB) which can then be applied to any problem in the domain. A MC represents a piece of knowledge in the BKB, and in our case this will contain all or part of a student step. A BKB is a generalization of a BN where each node has independence from the others [3]. In our implementation each node in the BKB represents a group of individual model components where each component consists of a state, actions, and new states. We know that our data contains certain patterns that we can exploit. We would expect that some level of background knowledge will be needed in order to generalize overall knowledge in a domain. The logic domain is a good candidate for study since there are actual rules that we can discover. In fact, logic proof solvers already exist [2], but it is difficult for a solver to use the work that a student has already done as a basis for the remainder of a proof. Our goal is to determine hints that would lead students from where they are to a valid proof solution, and also to show the potential uses of such hints in other domains.

Model components (MCs) in the BKB can be at various levels of granularity. We have identified three levels from which the data from the MDP can be transformed. Level 3 MCs are simply all the state, actions, new state pairs extracted from the MDP. Level 2 MCs contain only one action, but can contain multiple new states. Level 1 MCs contain the smallest knowledge component that can be extracted and consist of the specific part of a state used by an action, the action, and the resulting part of the new state. In practice

in the logic domain, a level 1 MC represents a logic rule application. Level 3 MCs contain the most context specific information, but will be the hardest to match. For our experiments we use only the level 1 MCs. The level 1 MCs have only have the original state information needed for the specific action, and only the new state information that is derived from this action. Again, this information is normalized such that any combination of letters in a problem can be matched to the state features.

2 Experiment and Conclusion

We create a BKB from 12 problems in our logic data sets and verify that the level 1 MCs correspond to valid rules in the logic domain. These MCs can be combined to make a general rule set that can be used to solve a new proof problem. We show this by deriving a BKB from a set of problems and testing the MCs on a new problem to see if we could select rules in such a way that the problem can be solved. This is done with a leave-one-out cross validation where all but one problem is used to make the BKB and then the remaining problem is the test case for the BKB. To provide hints with the BKB method, the current state in the new problem will be selected against the BKB and all matching model components will be returned. Based on the specificity of the states and the overall value of the returned items a hint can be given using the model component that is selected as the best for that step. To test if the BKB method is able to give hints we used all the MCs from other problems to solve the current problem. For each of the problems the BKB could solve the new problem with the exception of one problem. This is the only problem in the set that requires the “Equivalence” rule in order to reach a solution. This shows that we would be able to give hints using the BKB with most new problems unless they contained rules that had not previously occurred in a problem data set.

The primary findings of this research suggest that we can generalize the MDP method into a Bayesian Knowledge Base, which contain MCs that can be used to solve new problems. The structure of the MCs can be stated in an “if-then” format that is very similar to the production rules used in the cognitive tutors. This is extremely encouraging since the ability to automate some or all production rules for a cognitive tutor would save a tremendous amount of time in tutor development.

References

- [1] Barnes, T., Stamper, J., Croy, M., Lehman, L. (2008). A Pilot Study on Logic Proof Tutoring Using Hints Generated from Historical Student Data. *In Baker, Barnes, Beck (Eds.) Procs of the 1st Intl Conf on Educational Data Mining (EDM 2008)*, pp. 197-201.
- [2] McCune, W. and Shumsky O., "Ivy: A Preprocessor and Proof Checker for First-order Logic". Chapter 16 in Computer-Aided Reasoning: ACL2 Case Studies (ed. M. Kaufmann, P. Manolios, and J Moore), Kluwer Academic Publishers, 2000.
- [3] Santos, E., Jr., & Santos, E. (1996) Bayesian Knowledge-bases. *Technical report AFIT/EN/TR96-05*, Department of Electrical and Computer Engineering, Air Force Institute of Technology.

A Review of Student Churn in the Light of Theories on Business Relationships

Jaan Ubi and Innar Liiv

Jaanbi@staff.ttu.ee

Innar.Liiv@ttu.ee

Department of Informatics, Tallinn University of Technology (TTU)

Abstract. The goal of this review is to use business theories in student retention research, which has so far been informed by economics, organizational behavior, psychology, sociology. Relationships in business networks are compared to these between students and universities, putting forth relevant characteristics for student churn/retention research. Theories regarding a taxonomy of customer churn, its determinants and consequences are also viewed in this context and implications for educational data mining (DM) are put forth.

The relationship between entities is traditionally either following a strict hierarchical fiat (HF) – if the parties belong to the same organization –, or it's essentially an arm's length transaction (ALT) – if there is a market relationship between the entities. Recent business theories describe a more nuanced reality. In the light of wider changes in research, today's corporations are found to be heterarchical; other authors speak of autonomous strategic initiatives that take place in corporations violating the principles of hierarchy. The relationships between companies are neither of ALT type. The works of Uppsala school [1] show that not only is a corporation essentially a network of units (as is elsewhere described both from multinational corporation (MNC) and its subsidiary perspective), but it is also embedded in a business network of its own. The picture is further complicated by the individualism-collectivism dimension. Big MNCs are comprised of internal markets (within one firm!) with an ongoing internal competition for world product mandates, centers-of-excellence, etc. between the subsidiary units; whereas the supply chain relationships that a firm belongs to, have been described as coevolving systems. Concentrating our attention on the relationships in business networks, we see that two firms gradually increase their commitment, as they do business with each other. A process of learning about each other's capabilities, needs and strategies takes place, as well as a formation of routines for undertaking transactions. Sides adapt to each other incrementally. Knowledge transfer is inherently present – with organizational learning taking place – the results being often tacit and intangible.

The relationship between a student and the university varies on the HF-ALT dimension. A student can simply purchase single classes from the Open University; one may be a full time student with an opportunity to call the university his *alma mater* after graduation; within the university's administrative framework, the studies can also in part be paid for by giving consults to one's peers; during the post graduate studies becoming a teaching assistant and teaching simultaneously with the studies is even more common; and finally – it is a goal of universities to populate the ranks of its faculties with the best graduates, in which case the student would administratively become a part of the organization. Entering studenthood comprises of overcoming various entry barriers. The curriculum is substantially different from that of a high school and the university studies are qualitatively harder as well – as the amount of independent work is greater, the tempo in the classes faster and as in some universities general courses can be amongst the most

difficult in the undergraduate curricula. At the same students have to learn scheduling, budgeting, develop their EQ and career. The steep entry barriers underline the commitment it takes to enter the university relationship. Therefore sunk costs are formed, which are reflected in the fact that student churn lessens considerably later on. The mutual commitment and adaptation is evident in the following: the student will be able to pursue further career goals after graduation; the student becomes familiar with the university life, procedures and administrative system; the youth helps to keep the university abreast of times; the university assesses its employees based on student feedback. And perhaps most importantly – the students adapt to the university and their academic mentor's profile. For both sides knowledge transfer and (organizational) learning ensue and as ties with the industry are created, so are the intangible assets.

Student churn taxonomy as a basis for further work

Customer churn is the propensity of customers to cease doing business. The cost of acquiring new customers is many times higher than that of retaining the existing ones. Customer (monetary) lifetime value (CLV) has been linked to customer tenure and is also something that previous research has considered in educational settings, although misinterpreting the NPV term. The most important determinants of customer churn are switching costs, satisfaction and future usage. The switching costs are transaction costs (while transferring from one university to another), learning costs (differences in curricula) and artificial costs (a scholarship keeping a student at the university). It can be inferred from the discussion above that satisfaction is a key determinant of student churn and that future usage plays an important role in the relationship. Bringing a parallel from business, according to a taxonomy [2], involuntary external churn occurs in the case of exmatriculation or accrued debt; voluntary external churn in case of relocation, switch to another university or alternative career path, for family considerations (this type can further be classified as either deliberate or incidental according to whether the locus of origin lies with the student); internal churn takes place inside the university framework – such as moving between full time and part time study plan; churn can furthermore be either customer or competitor initiated, the one at hand being mainly the former.

This review has resulted in an identification of a need to develop a classification model for several student churn definitions, which will additionally be informed by the student-university relationship information. Performance, demographic, high school and satisfaction data is available. DM and visualization will use seriation, matrix reordering, clustering and feature selection. As research has combined social network analysis with DM, student network data (the classes taken with the peers; the supervisors; the specialization; but also in the mid-term the facebook data) will be utilized. The retention effort is pinpointed by „quantifying” the students (CLV). In Estonia, projects analyzing the global effectiveness of students allocation, further reducing churn, are possible.

References

- [1] Forsgren, M. *Managing the embedded multinational*, 2005. UK: Edward Elgar.
- [2] Mattison, R. *The telco churn management handbook.*, 2005 Illinois: XiT Press.

Towards EDM Framework for Personalization of Information Services in RPM Systems

Ekaterina Vasilyeva¹, Mykola Pechenizkiy¹, Aleksandra Tesanovic²,
Evgeny Knutov¹, Sicco Verwer¹, and Paul De Bra¹
{e.vasilyeva, m.pechenizkiy, e.knutov,s.verwer, i.zliobaite}@tue.nl,
aleksandra.tesanovic@philips.com, debra@win.tue.nl

¹Department of Computer Science, Eindhoven University of Technology
²Philips Research Laboratories

Abstract. Remote Patient Management Systems (RPM), besides monitoring the health conditions of patients, provide them with different information services that currently are predefined and follow a one-size-fits-all paradigm to a large extent. In this work we focus on the problem of knowledge discovery and patient modeling by mining educational data, motivational and instructional feedback provided to patients within RPM system.

1 Introduction

Chronic diseases are the leading cause of death and healthcare costs in the developed countries. Healthcare systems are undergoing a paradigm shift from patient care in the hospital to the patient care at home [2]. It is believed that RPM systems, by providing adequate patient monitoring, instruction, education and motivation can facilitate normalization of the patients' conditions and prevent re-hospitalization. Recently, a possible architecture of the next generation personalized RPM systems was introduced, and a general process of knowledge discovery from RPM data, leading to identification of potentially useful features and patterns for patient modeling and construction of adaptation rules, was considered [1].

In this work we focus on the design of an EDM framework aimed at facilitating the adaptation and personalization of information services by discovering actionable patterns from educational material usage data as well as motivational and instructional feedback, and linking those with the conditions and quality of life of the home-monitored patients. We sketch a conceptual framework how EDM in RPM can be implemented and provide motivating examples based on our preliminary study of one real RPM dataset.

2 EDM for Personalization in RPM: First Steps and Further Work

Figure 1 (top) gives an insight how EDM technology can become an integral part of RPM systems. The output of knowledge discovery process will be utilized for patient modeling and providing input for the adaptation engine. One type of practically relevant questions related to patient modeling and adaptation includes e.g. *“what kinds of patients are likely to weigh themselves regularly if they review their weight charts”*; *“what is the relationship between the patients reviewing the charts and watching educational videos or reading motivational messages”*, *“do the patients (or what kind) restart weighting after receiving a message that they forgot to do so”*. Two examples drawn from the real RPM dataset, collected during a clinical trial, are shown at the bottom of Figure 1.

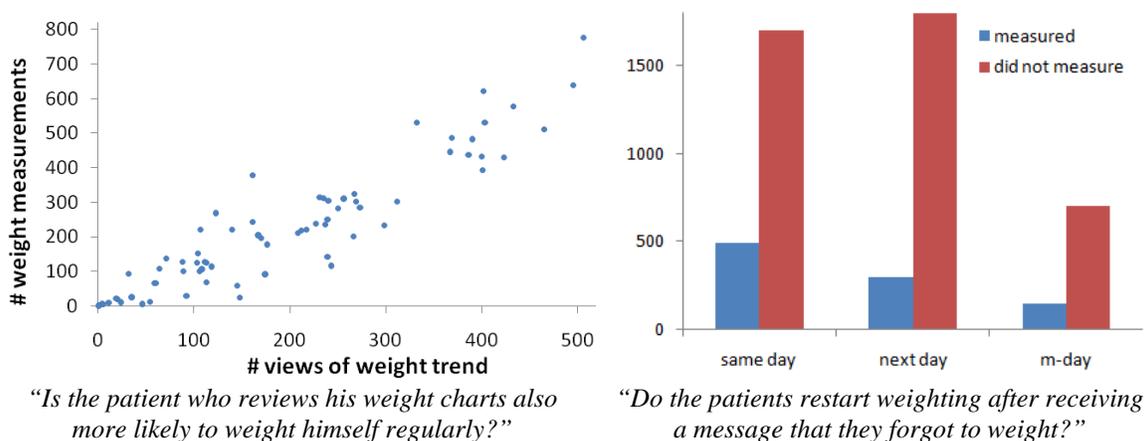
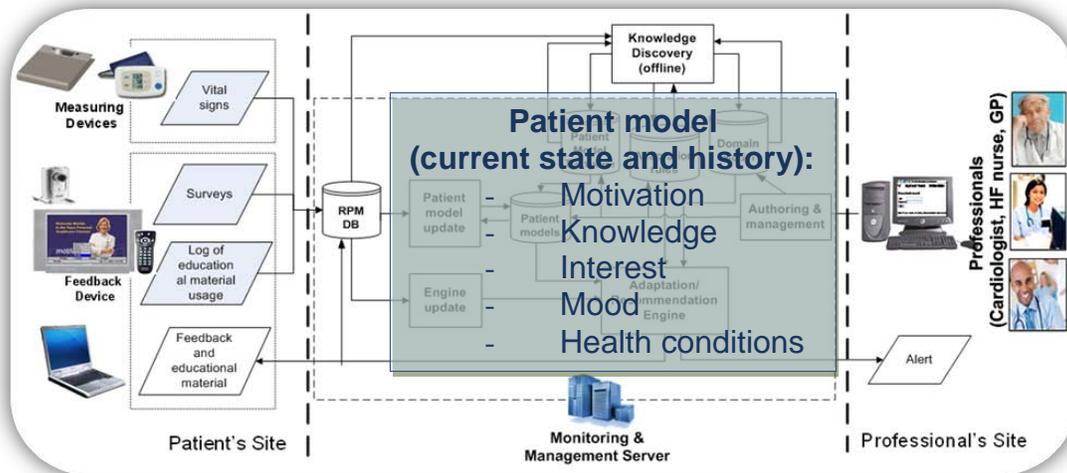


Figure 1. The role of EDM in RPM (top) and motivating examples (bottom)

The preliminary results of our exploration study suggest that there is potential for EDM to facilitate data-driven patient modeling and motivate the shift from the one-size-fits-all approach currently employed in the development of RPM systems to personalization in providing educational materials, motivational support and informational content to their users. Our further work includes the many-sided analysis of the RPM usage database with the focus on the educational content and usage data. The particular focuses include subgroup discovery and identification of signatures describing well-doing home-monitored patients and those who require more assistance of the medical staff.

Acknowledgments. This research is partly supported by KWR MIP, FP7 EU HeartCycle and NWO GAF projects.

References

- [1] Tesanovic A., Manev G., Pechenizkiy M., Vasilyeva. E. eHealth Personalization in the Next Generation RPM Systems. *Proceedings 22nd Int. Conf. IEEE CBMS 2009*. p.1-9.
- [2] Wang H. Disease management industry and high-tech adoption. *An Industry Report from Parks Associates*, Parks Associates, 2008.

A Case Study: Data Mining Applied to Student Enrollment

César Vialardi¹, Jorge Chue¹, Alfredo Barrientos¹, Daniel Victoria¹, Jhonny Estrella¹, Juan Pablo Peche¹
and Álvaro Ortigosa

{cvialar, jchue, abarrien, dvictoria, jestrella, jpeche}@ulima.edu.pe, Alvaro.ortigosa@uam.es

¹Facultad de Ingeniería de Sistemas, Universidad de Lima

²Escuela Politécnica Superior, Universidad Autónoma de Madrid

Abstract. One of the main problems faced by university students is deciding the right learning path based on available information such as courses, schedules and professors. In this context, this paper presents a recommender system based on data mining. This recommender system intends to create awareness of the difficulty and amount of workload entailed by a chosen set of courses. For the purpose of building the underlying model, this paper describes the generation of domain specific variables that are capable of representing students' past performance. The objective is to improve students' performance in general, by reducing the rate of misguided enrollment decisions.

1. Introduction

University Curricula allow students great flexibility and freedom of choice in terms of which and how many courses they can sign up for each term. The amount of workload and their ability to balance it successfully depends on these choices. Their results are also dependent on their own ability for a particular area of knowledge.

The main objective of this paper is to propose an enrollment recommender system to assist students in their decision making. The main contribution is the generation of two domain specific variables, namely the potential of student and the difficulty of courses.

A similar study was conducted by Al-Radaideh [1], in which he uses classification algorithms to evaluate the performance of students who studied the C++ course in Yarmouk University in 2005. To build a reliable classification model, it adopts the CRISP-DM methodology.

2. Domain Specific Variables

Domain specific metrics increase the representativeness of models. In this particular case we used two variables: the course difficulty and ability of a student towards a course; the latter is referred to as potential.

The course difficulty is represented by the average of the grades obtained by students. On the other hand, the potential is calculated per student for each course he may take. It is represented by the average of the grades a student has obtained in the prerequisites of a course and in previous attempts to pass the course; each grade is divided by the course difficulty.

3. Recommender System

The model that supports the recommender system was built using the Knowledge Discovery in Databases Methodology [2] using the C4.5 algorithm as classification engine [3]. It included the domain specific variables presented in Section 2 in order to improve its effectiveness. This system is integrated into the current Enrollment System.

During enrollment students choose a set of courses and obtains a forecast for each of them: PASS/FAIL. This enables them to make informed and conscious decisions hence indirectly improving their performance. (Figure 1) shows the sequence of this process in detail.

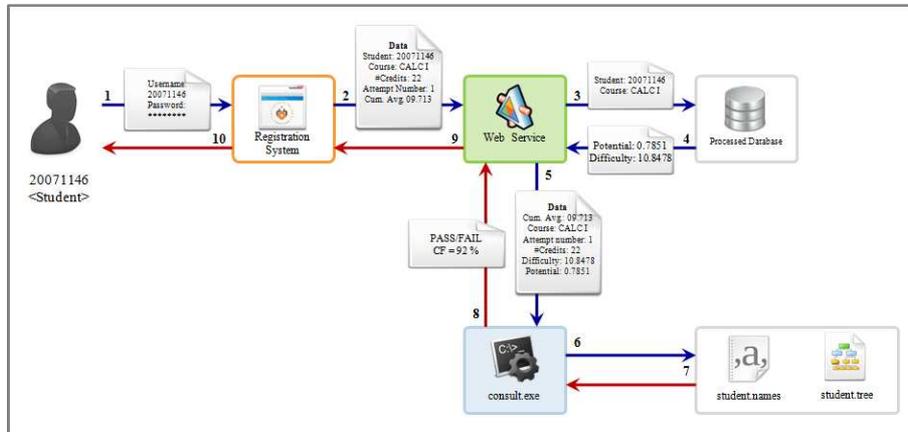


Figure 1. Recommendation Sequence Diagram

4. Conclusions and Future Work

The main benefit of the proposed system is the awareness created in students about the enrollment process: they will be aware of the fact that they are not currently skilled enough to success in courses in a particular area; it will also help them appraise the difficulty of courses and realize the possibility of unbalanced workloads. In the long run, these positive effects will lower failure rate of students hence improving their learning process and their learning paths.

Future works should strive to improve data cleaning to remove noise from the model. The model itself could be enhanced by improving the C4.5 pruning method or by adding significant attributes.

Acknowledgement

This work has been funded by Spanish Ministry of Science and Education through the HADA projects (TIN2007-64718) and the Universidad de Lima through the IDIC (Research Institute).

References

- [1] Al-Radaideh, Qasem A., Al-Shawakfa, Emad M., and Al-Najjar, Mustafa. Mining Student Data using Decision Trees. *International Arab Conference on Information Technology*, 2006.
- [2] Fayyad, U., Piatetsky-Shapiri, G., and Smyth, P. From Data mining to knowledge Discovery in Databases. *AAAI*, 1997, p. 37-54.
- [3] Quinlan, J.R. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, California, USA, 1993.

Representing Student Performance with Partial Credit

Yutao Wang, Neil T. Heffernan, Joseph E. Beck
{yutaowang, nth, josephbeck}@wpi.edu
Computer Science Department, Worcester Polytechnic Institute

1 Introduction

The educational data mining community has not been paying much attention to how much assistance a student needs. Feng and Heffernan[1] showed that we can predict student performance better by accounting for amount of assistance, but didn't reduce it to a value that could be shared with students. In this paper we want to see if we can better model student performance by replacing traditional binary measures (correct/ incorrect) with continuous partial credit, which is assigned based on the details in the student responses. The data we use is from ASSISTment, a web-based math tutoring system for 7th-12th grade students. The system helps the student learn the required knowledge by breaking the problem into sub-questions called 'scaffolding' or by giving the student hints on how to solve the question.

2 Partial Credit Algorithm & Evaluations

In the algorithm, students get a maximum score of '1' for responding correctly on the first attempt; otherwise, we assign partial credit by penalizing them for hints, wrong attempts and asking for scaffolding. We use $1 / (\# \text{ of total_hints})$ as the penalty for each hint; $(\# \text{ of correct_answers}) / (\# \text{ of total_answers} - i)$ as the penalty for the i^{th} wrong attempt in multiple choice problems; 0.1 as the penalty for each wrong attempt in open response problems and for asking for scaffolding. Table 1 shows an example of two data records, in which two students get the same binary performances but very different partial credit (only some of the features are shown because of limited space).

Table 1. Example of different Partial Credit

	...	#hints	#total hints	#wrong attempts	#correct answers	#total answers	Binary performance	Partial credit
1	...	1	5	0	1	5	0	0.8
2	...	1	5	2	1	5	0	0.22

We conducted two analyses on 52,529 data records from 72 students in order to evaluate the predictive power of partial credit in comparison to traditional binary performance.

The first analysis compares the predictive power of the mean value of all the previous performances. As a baseline we used a logistic regression model with binary performance as the dependent, and student id as an independent. We then compare using mean previous partial credit vs. mean previous binary performance as independents in the

model. The results are shown in Table 2. The model with mean previous *partial credit* gets a 0.0019 better R-squared than the one with mean previous *binary credit*. The absolute numerical improvement is small, which indicates that it is difficult to substantially improve student modeling. On the other hand, because knowing mean previous *binary performance* only brings 0.0046 better R-squared over baseline, knowing mean previous *partial credit* brings a relative 41.3% better R-squared compared to it.

Table 2. Comparison between partial credit and binary performance

	R² (analysis 1)	accuracy (analysis 1)	R² (analysis 2)	accuracy (analysis 2)
baseline	0.1857	0.6846	0.1893	0.6721
binary	0.1903	0.6871	0.1979	0.6757
partial credit	0.1922	0.6884	0.1999	0.6791

In the second analysis, instead of the mean value of previous performances, we compare the predictive power of the trends of these two measures of performance. We choose data in which students have 5 opportunities in one skill and use the previous 4 performances to create a trend line to predict the value of the 5th performance. To smooth the result and generate a bounded prediction between 0 and 1, we also apply a logistic regression model to this prediction. The result is shown in the fourth and fifth column of Table 2. Using partial credit improves R-squared by about 0.002 compared to using binary performance. This result is consistent with analysis 1.

3 Conclusions and Future work

In this paper we present a naïve algorithm to assign partial credit given detailed student responses. Partial credit performance contains much more information than binary performance, which is currently used in almost all researchers in the educational data mining field. Evaluations show that partial credit improves student model fitting by only a small absolute value but a high relative value compared to the binary performance.

One question we are interested in is how to refine the algorithm to better fit student data and infer student knowledge. Also, we are interested in finding out in which situations partial credit does better than binary performance, so we can use it as an efficient complement to all the current models that use only the binary performance.

References

- [1] Feng, M., 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)? *Educational Data Mining*, 2010. Pittsburgh, PA.

Where in the World? Demographic Patterns in Access Data¹

Mimi M. Recker¹, Beijie Xu¹, Sherry Hsi², and Christine Garrard³
mimi.recker@usu.edu, bejjie.xu@aggiemail.usu.edu
sherryh@berkeley.edu, chrisg@leupold.gis.usu.edu

¹Department of Instructional Technology & Learning Sciences, Utah State University

²Center for Technology Innovation, Lawrence Hall of Science, UC Berkeley

³RS/GIS Laboratory, Utah State University

Abstract. Standard webmetrics tools record the IP address of users' computers, thereby providing fodder for analyses of their geographical location, and for understanding the impact of e-learning and teaching. In this paper, we describe how two web-based educational systems were engineered to collect geo-referenced data. This is followed by a description of joining these data with demographic and educational datasets for the United States, and mapping different datasets using geographic information system (GIS) techniques to visually display their relationships. We conclude with results from statistical analyses of these relationships that highlight areas of significance.

1 Introduction

Web-based learning environments can be engineered to collect detailed data about their users and their activities (generally called *webmetrics*). In addition, standard webmetrics tools record the IP address of users' computers, thereby providing fodder for analyses of their geographical location, and for understanding the impact and the effectiveness of e-learning and teaching. In our study, we used two web-based learning environments – the Instructional Architect (IA) and the Exploratorium Learning Resources Collection (ELRC) – to explore how to collect geo-referenced data, and analyze such datasets by joining them with the demographic and national educational datasets, and using geographic information system (GIS) to visually display the relationships.

2 Collecting Geo-Referenced Data

The IA is an educational digital library service developed to support the instructional use of the online learning resources. With the IA, teachers are able to create web-based instructional activities, called IA projects, and give their students exclusive access to them. The ELRC is a digital library of over 700 teacher-tested science activities and instructional resources inspired and created from the Exploratorium's exhibits, public program events, and teacher professional development programs. Both websites have been engineered to collect detailed online usage data using Google Analytics (GA). As part of their standard reporting tool, GA estimates the visitors' location using the client computers' IP addresses, in a process called geo-location [1].

¹ For full paper, figures, and references, please visit <http://edm.usu.edu/publications/demo.pdf>.

3 Results

The GA reporting tool generates geographic maps of website visitors. The IA's map has darker shades in Utah, New York and Michigan, three areas that have received IA's teacher development programs; similarly, the ELRC's map has darker shade in California, where the ELRC library locates. The maps show that both groups are successful in local dissemination activities.

GA location data can be joined with other demographic and geographical datasets. In this study, we extracted population, number of schools, number of school districts, per capita income, and median family income from the ESRI 9.3 GIS software, and the National Center for Education Statistics. These were then overlaid with our GA location data. For example, Figure 1 shows IA traffic overlaid on one of the datasets –median family income. Such maps can help visually reveal relationships between site usage and demographic or school characteristics.



Figure 1. U.S. map showing IA visits (darker dots indicate more visits) overlaid with median family income over 1 year.

We also examined statistical relationships between the demographic predictors and the number of site visits per location as reported by GA. Due to the high correlation between some of the predictors, only three out of the five were selected, which were population, number of school districts, and per capita income. A negative binomial regression (to handle the skewed data) showed that *population*, and *per capita income* were statistically significant predictors of site visits to both the IA and the ELRC, and *number of school districts* was also a significant predictor for the ELRC.

We interpret these results to mean that online visitors to these sites came from, not surprisingly, more densely populated areas. In addition, the relationship with per capita income may be a function of the amount of resources (i.e., computing) available in the local schools and communities.

References

- [1] Muir, J. A., & Oorschot, P. C. V. Internet Geolocation: Evasion and Counterevasion. *ACM Computing Surveys (CSUR)*, 2009, 42(1). Article 4.

Pundit: Intelligent Recommender of Courses

Ankit Ranka¹, Faisal Anwar¹, Hui Soo Chae¹

¹ EdLab, Russell Hall 5th Floor
525 West 120th Street, New York
NY 10027, USA

ar2816@columbia.edu, fa2227@columbia.edu, hsc2001@tc.edu

Abstract. In this paper the authors describe Pundit, a course recommendation and search tool at Teachers College, Columbia University. The alpha prototype employs a novel combination of data retrieval and data mining approaches to recommend courses to users based on a match between their profiles and course contents. We utilize course management system and library e-reserves data to collect information about course content that is indexed and then matched with user profiles. In conclusion, we define an evaluation function that was used to determine the quality of recommendations.

1 Introduction

Pundit was developed as a course search and recommendation system for students at Teachers College (TC), Columbia University. The following data sources were used to design the Pundit database: 1) the TC Library e-reserve system (DocDel), 2) the TC course management system (Classweb), and 3) the TC Directory. We utilized a content-based approach [5] and information retrieval techniques [4] to build the recommendation system. A ranked list of recommended courses was generated for each user based on: 1) user profiles (e.g., degree status, program of study), 2) user behavior (e.g., courses taken), and 3) an inverted index [2] created for all TC courses.

2 Methodology

Using key words from user profiles [3], we queried the inverted index of all TC courses. Profiles include user's scholarly interests, academic background, employment history, resume, publications, and additional documents provided by users. In this alpha version of Pundit, we used over 150 user profiles from Netposse [1], an academic social networking tool at Teachers College, that helps users to identify student, faculty, and staff who share similar interests and backgrounds.

Pundit first takes extracted keywords from user's Netposse profile. Then the application generates bi-grams that correspond to the extracted keywords. Afterwards, Pundit eliminates those words that are part of bi-grams from the keyword list. Then it issues an OR query containing all the bi-grams and remaining keywords to the index. Apache Lucene's ranking function was used to calculate a score for each course. Lastly, Pundit removes all courses that were previously taken or taught by the user.

3 Evaluation

Our evaluation strategy links every user in the system with every course, s/he has taken at Teachers College. We compared this information with recommendations from Pundit to determine the number of intersections. Before doing this, we removed transcripts and/or other course related information from user profiles to eliminate any bias towards the final results. The data used for evaluating the Pundit methodology consisted of 45,713 course files, 3,403 courses, 35,215 student & faculty, 40 user profiles, 74 degree programs, and 247 department. Figure 1 below shows the results we obtained when 200 recommended courses for 40 faculty member profiles were compared against the courses faculty previously taught.

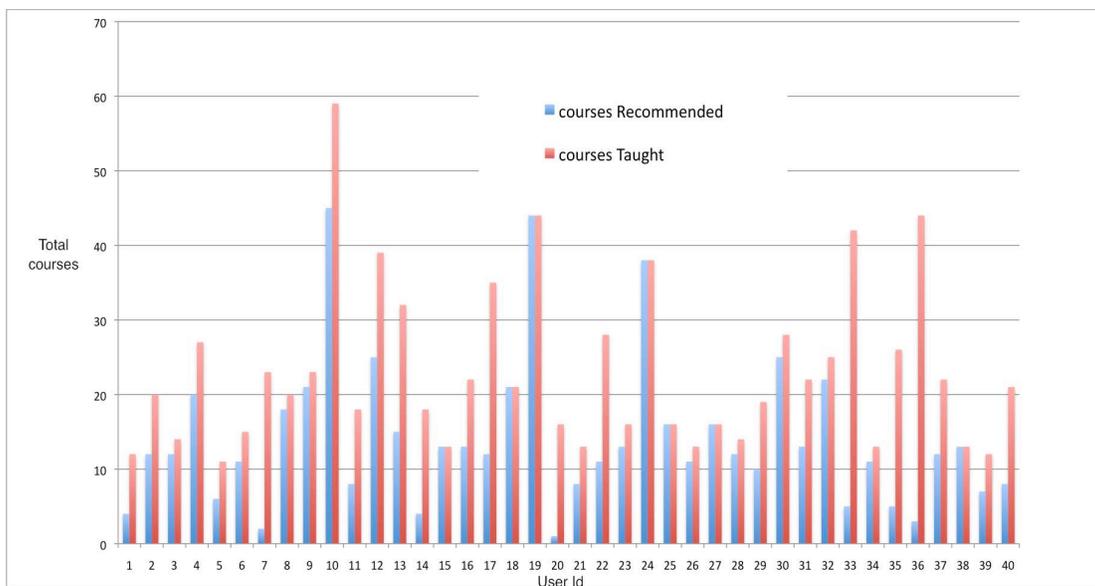


Figure 1. Courses Recommended Vs. Courses Taught

4 References

[1] Anwar, F., Chae, H., Natriello, G. Netposse: A Tool for Connecting Users in Virtual Communities. *Journal of Systems, Cybernetics, and Informatics*, 2008, 8, p. 67-71.

[2] Baeza-Yates, R., Ribeiro-Neto, B. *Modern Information Retrieval*, 1999. New York, NY: Addison Wesley.

[3] Basu, C., Hirsh, H., Cohen W. Recommendation as Classification: Using Social and Content-Based Information in Recommendation. *Proceedings of the 15th National Conference on Artificial Intelligence*, 1998, p. 714-720

[4] Manning, C. D., Raghavan, P., Schütze, H., *Introduction to Information Retrieval*. 2008. New York, NY: Cambridge University Press.

[5] Pazzani, M. J., Billsus, D. Content-Based Recommendation Systems. *Lecture Notes in Computer Science*, 2007, 4321, p. 325-341.

Author Index

Aleahmad, Turadg	267	D'Mello, Sidney	31
Aleven, Vincent	267	Domingez, Anna Katarina	91
Anderson, John R.	51	Estrella, Jhonny	333
Anwar, Faisal	339	Falakmasir, Mohammad	241
Arroyo, Ivon	1, 191	Hassan	
Ayers, Elizabeth	151	Feng, Mingyu	41
Azran, Ronit	293, 295	Ferris, Jennifer	51
Bachmann, Matt	269	Fincham, Jon M.	51
Bagadia, Muntaquim	275	Forsyth, Carol	287
Baker, Ryan S.J.d.	11, 141, 181, 289	Garrad, Christine	337
Balla, Amar	279	Gobert, Janice D.	141, 181, 269
Barnes, Tiffany	297, 327	Goldstein, Adam	289
Barrientos, Alfredo	333	Gong, Yue	61
Beck, Joseph E.	61, 269, 313, 335	González-Brenes, José P.	291
Becker, Lee	271	Gowda, Sujith M.	11
Bernauer, James	273	Graesser, Art	31, 287
Betts, Shawn	51	Groznik, Vida	309
Bian, Haiyun	277	Gupta, Anupriya	275
Biswas, Gautam	81	Ha, Eun Young	285
Bousbia, Nabila	279	Habibi, Jafar	241
Boyer, Kristy Elizabeth	285	Halpern, Diane	287
Bratko, Ivan	309	Hardof-Jaffe, Sharon	293
Brusilovsky, Peter	221	Heffernan, Neil T.	41, 61, 161, 289, 335
Butler, Heather	287	Hegazy, Nadia	299
Cade, Whitney	101, 281	Hershkovitz, Arnon	293
Cai, Zhiquaing	287	Howard, Larry	71, 81
Calvo, Rafael A.	111, 257	Hsi, Sherry	337
Chae, Hui Soo	339	Jeong, Hogyong	81
Champaign, John	231, 283	Johnson, Julie	71, 81
Choi, YounYoung	319	Johnson, Matthew W.	297
Chue, Jorge	333	Khodeir, Nabila	299
Cohen, Robin	231, 283	Kim, Sunghwan Mac	111
Cooper, David G.	191	Knutov, Evgeny	331
Croy, Marvin	327	Koedinger, Kenneth R.	201, 325
Curran, James R.	91	Kraut, Robert	267
Darwish, Nevin	299	Krivek, Jana	309
De Bra, Paul	331	Krüger, André	131, 305
Dean, Nema	151	Labat, Jean-Marc	279
Desmarais, Michel C.	21		

Lehman, Blair	101	Salvidar, Manuel Gerardo	121, 303
Lester, James C.	285	Sao Pedro, Michael A.	141, 181
Liiv, Innar	329	Schunn, Christian	211
Litman, Diane	211	Shanabrook, David H.	191
Luna, Jose María	171	Sheines, Richard	201
Macfadyen, Leah P.	301	Shih, Benjamin	201
Mathews, Moffat	221	Songmuang, Pokpong	321
Maull, Kieth E.	121, 303	Sorenson, Peter	301
Mehranian, Hasmik	1	Soundranayagam, Hema	323
Merceron, Agathe	131, 305	Southavilay, Vilaythong	257
Mills, Kieth	287	Srinivas, Suchismita	275
Mitrovic, Antonija	221	Stamper, John C.	325, 327
Montalvo, Orlando	141, 181	Sumner, Tamara	121, 303
Mostow, Jack	291, 307	Sweet, Shauna J.	319
Možina, Martin	309	Tan, Bao Hong (Lucas)	307
Nachmias, Rafi	293, 295	Tesanovic, Aleksandra	331
Nakama, Adam	141, 181	Ubi, Jaan	329
Neitzel, Carin	71	Ueno, Maomi	321
Nugent, Rebecca	151	vanVuuren, Sarel	271
Olney, Andrew	101, 281	Vasilyeva, Ekaterina	317, 331
Ortigosa, Álvaro	333	Ventura, Sebastián	171, 317
Pardo, Zachary	161	Verwer, Sicco	331
Pavlik Jr., Philip I.	311	Vialadri, César	333
Peche, Juan Pablo	333	Victoria, Daniel	333
Pechenizkiy, Mykola	317, 331	Vouk, Mladen A.	285
Pelczer, Ildiko	21	Wallis, Michael D.	285
Philips, Robert	285	Wanas, Nayer	299
Powell, Jay	273	Wang, Yutao	335
Rai, Dovan	313	Ward, Wayne H.	271
Rajibussalim	249	Wolf, Benjamin	131, 305
Ranka, Ankit	339	Wood, Jonathan	287
Rebai, Issam	279	Woolf, Beverly Park	1, 191
Recker, Mimi M.	315, 337	Xiong, Wenting	211
Romero, Cristóbal	171, 317	Xu, Beijie	315, 337
Romero, José Raúl	171	Yacef, Kalina	91, 257, 323
Rupp, Andre A.	319	Yudelson, Michael	221
Sadikov, Aleksander	309		