

EDM 2011

**4th International Conference on
Educational Data Mining**

PROCEEDINGS OF THE
FOURTH INTERNATIONAL CONFERENCE ON
EDUCATIONAL DATA MINING

Eindhoven, July 6-8, 2011

**Mykola Pechenizkiy, Toon Calders,
Cristina Conati, Sebastian Ventura,
Cristobal Romero and John Stamper**

Mykola Pechenizkiy, Toon Calders, Cristina Conati, Sebastian Ventura,
Cristobal Romero and John Stamper

International Conference on Educational Data Mining (EDM) 2011
Proceedings of the 4th International Conference on Educational Data Mining
Mykola Pechenizkiy, Toon Calders, Cristina Conati, Sebastian Ventura,
Cristobal Romero and John Stamper (eds.)
Eindhoven, 6-8 July, 2011

A catalogue record is available from the Eindhoven University of Technology Library

ISBN: 978-90-386-2537-9

Cover: Flying pins with the background the Chamber of Commerce located at the TU/e campus.

Printing and binding: TU/e printservice

Preface

The 4th International Conference on Educational Data Mining (EDM 2011) brings together researchers from computer science, education, psychology, psychometrics, and statistics to analyze large datasets to answer educational research questions. The conference, held in Eindhoven, The Netherlands, July 6-9, 2011, follows the three previous editions (Pittsburgh 2010, Cordoba 2009 and Montreal 2008), and a series of workshops within the AAAI, AIED, EC-TEL, ICALT, ITS, and UM conferences.

The increase of e-learning resources such as interactive learning environments, learning management systems, intelligent tutoring systems, and hypermedia systems, as well as the establishment of state databases of student test scores, has created large repositories of data that can be explored to understand how students learn. The EDM conference focuses on data mining techniques for using these 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.

This year's conference includes short papers as a new submission category targeting original and unpublished research with merit in terms of originality and importance rather than maturity and technical validation. In the paper track, we received 60 long and 20 short papers, each of which was reviewed by three experts in the field, resulting in 20 long and 17 short papers accepted for presentation at the conference (some of the long paper submissions have been accepted as short paper). We also received 22 posters, targeting work in progress and last minute results with high potential to foster new developments and interesting discussions during the conference's poster presentation sessions. These sessions included the presentation of 30 posters, 14 from the original pool of poster submissions and the reminder from the pool of paper submissions.

All accepted submissions appear in these proceedings. The conference also includes invited talks by Barry Smyth (University College, Dublin, Ireland), John Stamper (Carnegie Mellon University, USA) and Erik-Jan van der Linden (MagnaView B.V., the Netherlands), with abstract in these proceedings.

We would like to thank Eindhoven University of Technology for the sponsorship and hosting of EDM'2011. We would like to thank the Netherlands Organization for Scientific Research (NWO), Belgium-Netherlands Association for Artificial Intelligence (BNVKI) and the Dutch Research School for Information and Knowledge Systems (SIKS), University of Cordoba and PSLC DataShop.

We also want to acknowledge the amazing work of the program committee members and additional reviewers, who with their enthusiastic contributions gave us invaluable support in putting this conference together.

Our special thanks to the local organizing team and additional thanks to Evgeny Knutov and Jorn Bakker for their technical support on putting these proceedings together.

Last but not least we would like to thank Arnon Hershkovitz who has served as the Web Chair of EDM series from its first edition.

June 2011

Cristina Conati and Sebastian Ventura – Program Chairs
Mykola Pechenizkiy and Toon Calders – Conference Chairs
Cristobal Romero and John Stamper – Posters Chairs

Organization

CONFERENCE CHAIRS

Mykola Pechenizkiy
Toon Calders

Eindhoven University of Technology, The Netherlands
Eindhoven University of Technology, The Netherlands

PROGRAM CHAIRS

Cristina Conati
Sebastian Ventura

University of British Columbia, Canada
University of Cordoba, Spain

POSTERS CHAIRS

Cristobal Romero
John Stamper

University of Cordoba, Spain
Carnegie Mellon University, USA

WEB CHAIR

Arnon Hershkovitz

Tel Aviv University, Israel

LOCAL ORGANIZING TEAM

Jorn Bakker
Ekaterina Vasilyeva

Paul De Bra
Yongming Luo

Lam Hoang

Evgeny Knutov
George Fletcher

Riet van Buul
Ine van der Ligt

STEERING COMMITTEE

Esma Aïmeur
Ryan Baker
Tiffany Barnes
Joseph E. Beck
Michel C. Desmarais
Neil Heffernan
Cristobal Romero
Kalina Yacef

University of Montreal, Canada
Worcester Polytechnic Institute, USA
University of North Carolina at Charlotte, USA
Worcester Polytechnic Institute, USA
Ecole Polytechnique de Montreal, Canada
Worcester Polytechnic Institute, USA
Cordoba University, Spain
University of Sydney, Australia

PROGRAM COMMITTEE

Esma Aimeur
Elizabeth Ayers
Ryan S.J.d. Baker
Tiffany Barnes
Joseph Beck
Bettina Berendt
Gautam Biswas
Jesús G. Boticario
Min Chi
Cristophe Choquet
Richard Cox
Michel Desmarais
Sidney D'Mello
Mingyu Feng
Davide Fosatti
Eva Gibaja
Daniela Godoy

University of Montreal, Canada
Carnegie Mellon University, USA
Worcester Polytechnic Institute, USA
University of North Carolina at Charlotte, USA
Worcester Polytechnic Institute, USA
Katholieke Universiteit Leuven, Belgium
Vanderbilt University, USA
U.N.E.D., Spain
University Of Pittsburgh, USA
Université du Maine, France
University of Sussex, UK
Ecole Polytechnique de Montreal, Canada
University of Memphis, USA
SRI International, USA
Carnegie Mellon University, Qatar
Universidad de Córdoba, USA
Universidad Nacional del Centro de la Provincia de Buenos Aires, Argentina

Neil Heffernan	Worcester Polytechnic Institute, USA
Arnon Hershkovitz	Worcester Polytechnic Institute, Israel
Roland Hubscher	Bentley University, USA
Sebastian Iksal	Université du Maine, France
Juday Kay	University of Sydney, Australia
Jihie Kim	University of Southern California, USA
Mirjam Köck	Johannes Kepler University, Austria
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
Riccardo Mazza	University of Lugano/University of Applied Sciences of South. Switzerland
Gordon McCalla	University of Saskatchewan, Canada
Agathe Merceron	Beuth University of Applied Sciences, Germany
Julia Mingallon Alfonso	Universitat Oberta de Catalunya, Spain
Jack Mostow	Carnegie Mellon University, USA
Rafi Nachmias	Tel Aviv University, Israel
Roger Nkambou	Université du Québec à Montréal (UQAM), Canada
Alvaro Ortigosa	Universidad Autónoma de Madrid, Spain
Alexandros Paramythios	Johannes Kepler University, Austria
Philip I. Pavlik	Carnegie Mellon University, USA
Mykola Pechenizkiy	Eindhoven University of Technology, Netherlands
Cristobal Romero	Cordoba University, Spain
Carolyn Rose	Carnegie Mellon University, USA
Erin Shaw	University of Southern California, USA
John Stamper	Carnegie Mellon University, USA
Jun-Ming Su	National Chiao Tung University, Taiwan
Steven Tanimoto	University of Washington, USA
Sebastian Ventura	Cordoba University, Spain
Stepehn Weibelzahl	National College of Ireland, Ireland
Kalina Yacef	University of Sydney, Australia
Michael Yudelson	University of Pittsburgh, WPI
Amelia Zafra	Universidad de Córdoba, Spain
Osmar Zaiane	University of Alberta, Canada

ADDITIONAL REVIEWERS

Yue Gong	Sujith Gowda	John Kinnebrew	Daniel Mack
Zach Pardos	Terry Peckham	Michael Sao Pedro	Soo Won Seo
Benjamin Shih	Vilaythong Southavilay	Fodé Touré	Jianfei Wu
Jaebong Yoo			

Sponsors



Technische Universiteit
Eindhoven
University of Technology



Netherlands Organisation for Scientific Research



PSLC DATASHOP
a data analysis service for the learning science community



UNIVERSIDAD DE CÓRDOBA

<http://www.tue.nl>
<http://www.unimaas.nl/bnvki>
<http://www.nwo.nl>
<http://www.siks.nl>
<https://pslcdatashop.web.cmu.edu/>
<http://www.ucd.es>

Table of Contents

Invited Talks (abstracts)

Social Information Discovery <i>Barry Smyth</i>	3
On exploration and mining of data in educational practice <i>Erik-Jan van der Linden, Martijn Wijffelaars, Thomas Lammers</i>	5
EDM and the 4th Paradigm of Scientific Discovery - Reflections on the 2010 KDD Cup Competition <i>John Stamper</i>	7

Full Papers

Factorization Models for Forecasting Student Performance <i>Nguyen Thai-Nghe, Tomáš Horváth and Lars Schmidt-Thieme</i>	11
Analyzing Participation of Students in Online Courses Using Social Network Analysis Techniques <i>Reihaneh Rabbany Khorasgani, Mansoureh Takaffoli and Osmar Zaïane</i>	21
A Machine Learning Approach for Automatic Student Model Discovery <i>Nan Li, William Cohen, Kenneth R. Koedinger and Noboru Matsuda</i>	31
Conditions for effectively deriving a Q-Matrix from data with Non-negative Matrix Factorization <i>Michel Desmarais</i>	41
Student Translations of Natural Language into Logic: The Grade Grinder Translation Corpus Release 1.0 <i>Dave Barker-Plummer, Richard Cox and Robert Dale</i>	51
Instructional Factors Analysis: A Cognitive Model For Multiple Instructional Interventions <i>Min Chi, Kenneth Koedinger, Geoff Gordon, Pamela Jordan and Kurt Vanlehn</i>	61
The Simple Location Heuristic is Better at Predicting Students Changes in Error Rate Over Time Compared to the Simple Temporal Heuristic <i>Adaeze Nwaigwe and Kenneth Koedinger</i>	71
Items, skills, and transfer models: which really matters for student modeling? <i>Yue Gong and Joseph Beck</i>	81
Avoiding Problem Selection Thrashing with Conjunctive Knowledge Tracing <i>Kenneth Koedinger, Philip I. Pavlik Jr., John Stamper, Tristan Nixon and Steven Ritter</i>	91
Less is More: Improving the Speed and Prediction Power of Knowledge Tracing by Using Less Data <i>Bahador Nooraei B., Zachary Pardos, Neil Heffernan and Ryan Baker</i>	101

Analysing frequent sequential patterns of collaborative learning activity around an interactive tabletop	111
<i>Roberto Martinez Maldonado, Kalina Yacef, Judy Kay, Ahmed Kharrufa and Ammar Al-Qaraghuli</i>	
Acquiring Item Difficulty Estimates: a Collaborative Effort of Data and Judgment	121
<i>Kelly Wauters, Piet Desmet and Wim Van Den Noortgate</i>	
Spectral Clustering in Educational Data Mining	129
<i>Shubhendu Trivedi, Zachary Pardos, Gábor Sárközy and Neil Heffernan</i>	
Does Time Matter? Modeling the Effect of Time with Bayesian Knowledge Tracing	139
<i>Yumeng Qiu, Yingmei Qi, Hanyuan Lu, Zachary Pardos and Neil Heffernan</i>	
Learning classifiers from a relational database of tutor logs	149
<i>Jack Mostow, José González-Brenes and Bao Hong Tan</i>	
A Framework for Capturing Distinguishing User Interaction Behaviors in Novel Interfaces	159
<i>Samad Kardan and Cristina Conati</i>	
How to Classify Tutorial Dialogue? Comparing Feature Vectors vs. Sequences	169
<i>José González-Brenes, Jack Mostow and Weisi Duan</i>	
Automatically Detecting a Students Preparation for Future Learning: Help Use is Key	179
<i>Ryan S.J.D. Baker, Sujith Gowda and Albert Corbett</i>	
Ensembling Predictions of Student Post-Test Scores for an Intelligent Tutoring System	189
<i>Zachary Pardos, Sujith Gowda, Ryan S.J.D. Baker and Neil Heffernan</i>	
Improving Models of Slipping, Guessing, and Moment-By-Moment Learning with Estimates of Skill Difficulty	199
<i>Sujith M. Gowda, Jonathan P. Rowe, Ryan S.J.D. Baker, Min Chi and Kenneth R. Koedinger</i>	

Short Papers

A Method for Finding Prerequisites Within a Curriculum	211
<i>Annalies Vuong, Tristan Nixon and Brendon Towle</i>	
Estimating Prerequisite Structure From Noisy Data	217
<i>Emma Brunskill</i>	
What can closed sets of students and their marks say?	223
<i>Dmitry Ignatov, Serafima Mamedova, Nikita Romashkin, and Ivan Shamshurin</i>	
How university entrants are choosing their department? Mining of university admission process with FCA taxonomies.	229
<i>Nikita Romashkin, Dmitry Ignatov and Elena Kolotova</i>	

What's an Expert? Using learning analytics to identify emergent markers of expertise through automated speech, sentiment and sketch analysis	235
<i>Marcelo Worsley and Paulo Blikstein</i>	
Using Logistic Regression to Trace Multiple Subskills in a Dynamic Bayes Net	241
<i>Yanbo Xu and Jack Mostow</i>	
Monitoring Learners Proficiency: Weight Adaptation in the Elo Rating System	247
<i>Kelly Wauters, Piet Desmet and Wim Van Den Noortgate</i>	
Modeling students activity in online discussion forums: a strategy based on time series and agglomerative hierarchical clustering	253
<i>Germán Cobo, David García-Solórzano, Eugènia Santamaría, Jose Antonio Morán, Javier Melenchón and Carlos Monzo</i>	
Prediction of Perceived Disorientation in Online Learning Environment with Random Forest Regression	259
<i>Gökhan Akçapinar, Erdal Cosgun and Arif Altun</i>	
Analysing Student Spatial Deployment in a Computer Laboratory	265
<i>Vladimir Ivancevic, Milan Celikovic and Ivan Lukovic</i>	
Predicting School Failure Using Data Mining	271
<i>Carlos Marquez-Vera, Cristobal Romero and Sebastin Ventura</i>	
A Dynamical System Model of Microgenetic Changes in Performance, Efficacy, Strategy Use and Value during Vocabulary Learning	277
<i>Philip I. Pavlik Jr. and Sue-Mei Wu</i>	
Desperately Seeking Subscripts: Towards Automated Model Parameterization	283
<i>Jack Mostow, Yanbo Xu and Mdahaduzzaman Munna</i>	
Automatic Generation of Proof Problems in Deductive Logic	289
<i>Behrooz Mostafavi, Tiffany Barnes and Marvin Croy</i>	
Comparison of Traditional Assessment with Dynamic Testing in a Tutoring System	295
<i>Mingyu Feng, Neil Heffernan, Zachary Pardos and Cristina Heffernan</i>	
Evaluating a Bayesian Student Model of Decimal Misconceptions	301
<i>George Goguadze, Sergey Sosnovsky, Seiji Isotani and Bruce McLaren</i>	
Exploring user data from a game-like math tutor: a case study in causal modeling	307
<i>Dovan Rai and Joseph Beck</i>	

Posters

Goal Orientation and Changes of Carelessness over Consecutive Trials in Science Inquiry	315
<i>Arnon Hershkovitz, Ryan S.J.D. Baker, Janice Gobert and Michael Wixon</i>	
Towards improvements on domain-independent measurements for collaborative assessment	317
<i>Antonio R. Anaya and Jesús G. Boticario</i>	

A Java desktop tool for mining Moodle data	319
<i>Rafael Pedraza Perez, Cristobal Romero and Sebastián Ventura</i>	
Using data mining in a recommender system to search for learning objects in repositories	321
<i>Alfredo Zapata Gonzalez, Victor Hugo Menéndez Domínguez, Manuel Prieto and Cristobal Romero</i>	
E-learning Web Miner: A data mining application to help instructors involved in virtual courses	323
<i>Diego García-Saiz and Marta Zorrilla Pantaleón</i>	
Computerized Coding System for Life Narratives to Assess Students?' Personality Adaption	325
<i>Qiwei He, Bernard Veldkamp and Gerben Westerhof</i>	
Partially Observable Sequential Decision Making for Problem Selection in an Intelligent Tutoring System	327
<i>Emma Brunskill and Stuart Russell</i>	
Mining Teaching Behaviors from Pedagogical Surveys	329
<i>Joana Barracosa and Claudia Antunes</i>	
Variable Construction and Causal Modeling of Online Education Messaging Data: Initial Results	331
<i>Stephen Fancsali</i>	
The Hospital Classrooms Environments Challenge	333
<i>Carina González and Pedro A. Toledo</i>	
Combining study of complex network and text mining analysis to understand growth mechanism of communities on SNS	335
<i>Osamu Yamakawa, Takahiro Tagawa, Hitoshi Inoue, Koichi Yastake and Takahiro Sumiya</i>	
Logistic Regression in a Dynamic Bayes Net Models Multiple Subskills Better!	337
<i>Yanbo Xu and Jack Mostow</i>	
Studying the problem-solving strategies in the early stages of learning programming	339
<i>Edgar Cambranes-Martinez and Judith Good</i>	
Brick: Mining Pedagogically Interesting Sequential Patterns	341
<i>Anjo Anjewierden, Hannie Gijlers, Nadira Saab and Robert De Hoog</i>	
Intelligent evaluation of social knowledge building using conceptual maps with MLN	343
<i>Lorenzo Moreno, Carina González, Román Estévez and Beatrice Popescu</i>	
Identifying Influence Factors on Students Success by Subgroup Discovery	345
<i>Florian Lemmerich, Marianus Ifland and Frank Puppe</i>	
Analyzing University Data for Determining Student Profiles and Predicting Performance	347
<i>Dorina Kabakchieva, Kamelia Stefanova and Valentin Kisimov</i>	
The EDM Vis Tool	349
<i>Matthew Johnson, Michael Eagle, Leena Joseph and Tiffany Barnes</i>	

Towards Modeling Forgetting and Relearning in ITS: Preliminary Analysis of ARRS Data	351
<i>Yutao Wang and Neil Heffernan</i>	
Quality Control and Data Mining Techniques Applied to Monitoring Scaled Scores	353
<i>Alina Von Davier</i>	
eLAT: An Exploratory Learning Analytics Tool for Reflection and Iterative Improvement of Technology Enhanced Learning	355
<i>Anna Lea Dyckhoff, Dennis Zielke, Mohamed Amine Chatti and Ulrik Schroeder</i>	
Predicting graduate-level performance from undergraduate achievements	357
<i>Judith Zimmermann, Kay H. Brodersen, Jean-Philippe Pellet, Elias August and Joachim M. Buhmann</i>	
Mining Assessment and Teaching Evaluation Data of Regular and Advanced Stream Students	359
<i>Irena Koprinska</i>	
Investigating Usage of Resources in LMS with Specific Association Rules	361
<i>Agathe Merceron</i>	
Towards parameter-free data mining: Mining educational data with <i>yacaree</i>	363
<i>Jose Balcázar, Diego García-Saiz and Marta Zorrilla</i>	
Factors Impacting Novice Code Comprehension in a Tutor for Introductory Computer Science	365
<i>Leigh Ann Sudol DeLyser and Jonathan Steinhart</i>	
Investigating the Transitions between Learning and Non-learning Activities as Students Learn Online	367
<i>Paul Salvador Inventado, Roberto Legaspi, Merlin Suarez and Masayuki Numao</i>	
Learning parameters for a knowledge diagnostic tools in orthopedic surgery	369
<i>Sebastien Lallé and Vanda Luengo</i>	
Problem Response Theory and its Application for Tutoring	371
<i>Petr Jarušek and Radek Pelánek</i>	
Towards Better Understanding of Transfer in Cognitive Models of Practice	373
<i>Michael Yudelson, Philip I. Pavlik and Kenneth R. Koedinger</i>	

INVITED TALKS

(abstracts)

Social Information Discovery

Barry Smyth, University College Dublin, Ireland

The world of web search is usually viewed as a solitary place. Although millions of searchers use services like Google and Yahoo everyday, their individual searches take place in isolation, leaving each searcher to fend for themselves when it comes to finding the right information at the right time. Recently, researchers have begun to question the solitary nature of web search, proposing a more collaborative search model in which groups or users can cooperate to search more effectively.

For example, students will often collaborate as part of class projects, bringing together relevant information that they have found during the course of their individual searches. Indeed, despite the absence of explicit collaboration features from mainstream search engines, there is clear evidence that users implicitly engage in many different forms of collaboration as they search, although, these collaboration "work-arounds" are far from ideal. Naturally, this has motivated researchers to consider how future web search engines might better support different types of collaboration to take advantage of this latent need; for example, how might students collaborate as they search rather than defer the sharing of information as a post-search activity.

In this talk we focus on some of the ways in which web search may become a more social and collaborative experience. This will include lessons learned from both the theory and practice of a more collaborative approach to web search and we will describe recent attempts to bring collaboration support to mainstream search engines. We will consider a number of educational use-cases during the course of this talk to describe how instructors and learners can take full advantage of this more social perspective on web search.

On exploration and mining of data in educational practice

Erik-Jan van der Linden, MagnaView B.V., the Netherlands

Martijn Wijffelaars, MagnaView B.V., the Netherlands

Thomas Lammers, MagnaView B.V. and
Eindhoven University of Technology, the Netherlands

Educational institutions are confronted with increasing pressure from authorities and governments to justify their spending of public means. This, in turn, has led to increased internal use of the huge amounts of data in information systems on results, careers, absence, etc. Experience with a data analysis product that is actively used in 20+ schools (secondary education) indicates that visual presentation and user interaction are crucial to have analyses of large datasets lead to real improvement. Intricate and finely-tuned interaction between methods from the field of data mining and these interactive techniques may further aid schools.

EDM and the 4th Paradigm of Scientific Discovery - Reflections on the 2010 KDD Cup Competition

John Stamper, Carnegie Mellon University, USA

Technology advances have made the ability to collect large amounts of data easier than ever before. These massive datasets provide both opportunities and challenges for many fields and education is no different. Understanding how to deal with extreme amounts of student data in the EDM field is a growing problem. The 2010 KDD Cup Competition, titled "Educational Data Mining Challenge", included data for over 10,000 students. The students completed over 30 million problem steps collected over a year long courses from Carnegie Learning Inc.'s Cognitive Tutors. We believe these are the largest educational dataset at this level of granularity to be released publicly. The competition drew broad interest from the data mining community, but it was also clear that many in the research community could not handle datasets of this size. In this talk, John will discuss the 2010 KDD Cup and the impact of larger and larger amounts of data coming available for educational data mining and how this will drive the direction of educational research in the future.

FULL PAPERS

Factorization Models for Forecasting Student Performance

Nguyen Thai-Nghe, Tomáš Horváth and Lars Schmidt-Thieme, University of Hildesheim, Germany

Predicting student performance (PSP) is one of the educational data mining task, where we would like to know how much knowledge the students have gained and whether they can perform the tasks (or exercises) correctly. Since the student's knowledge improves and cumulates over time, the sequential (temporal) effect is an important information for PSP. Previous works have shown that PSP can be casted as rating prediction task in recommender systems, and therefore, factorization techniques can be applied for this task. To take into account the sequential effect, this work proposes a novel approach which uses tensor factorization for forecasting student performance. With this approach, we can personalize the prediction for each student given the task, thus, it can also be used for recommending the tasks to the students. Experimental results on two large data sets show that incorporating forecasting techniques into the factorization process is a promising approach.

1. INTRODUCTION

Predicting student performance, one of the tasks in educational data mining, has been taken into account recently [Toscher and Jahrer 2010; Yu et al. 2010; Cetintas et al. 2010; Thai-Nghe et al. 2011]. It was selected as a challenge task for the KDD Cup 2010¹ [Koedinger et al. 2010]. Concretely, predicting student performance is the task where we would like to know how the students learn (e.g. generally or narrowly), how quickly or slowly they adapt to new problems or if it is possible to infer the knowledge requirements to solve the problems directly from student performance data [Corbett and Anderson 1995; Feng et al. 2009], and eventually, we would like to know whether the students perform the tasks (exercises) correctly (or with some levels of certainty). As discussed in Cen et al. [2006], an improved model for predicting student performance could save millions of hours of students' time and effort in learning algebra. In that time, students could move to other specific fields of their study or doing other things they enjoy. From educational data mining point of view, an accurate and reliable model in predicting student performance may replace some current standardized tests, and thus, reducing the pressure, time, as well as effort on "teaching and learning for examinations" [Feng et al. 2009; Thai-Nghe et al. 2011].

To address the problem of predicting student performance, many papers have been published but most of them are based on traditional classification/regression techniques [Cen et al. 2006; Feng et al. 2009; Yu et al. 2010; Pardos and Heffernan 2010]. Many other works can be found in Romero et al. [2010]. Recently, [Thai-Nghe et al. 2010; Toscher and Jahrer 2010; Thai-Nghe et al. 2011] have proposed using recommendation techniques, e.g. matrix factorization, for predicting student performance. The authors have shown that *predicting student performance can be considered as rating prediction* since the *student*, *task*, and *performance* would become *user*, *item*, and *rating* in recommender systems, respectively. We know that learning and problem-solving are complex cognitive and affective processes that are different to shopping and other e-commerce transactions, however, as discussed in Thai-Nghe et al. [2011], the factorization models in recommender systems are implicitly able to encode latent factors of students and tasks (e.g. "slip" and "guess"), and especially in case where we do not have enough meta data about students and tasks (or even we have not enough background knowledge of the domain), this mapping is a reasonable approach.

¹<http://pslcdatashop.web.cmu.edu/KDDCup/>

Author's address: Information Systems and Machine Learning Lab (ISMLL), University of Hildesheim, Marienburger Platz 22, 31141 Hildesheim, Germany. Emails: {nguyen, horvath, schmidt-thieme}@ismll.de

Moreover, from the pedagogical aspect, we expect that students (or generally, learners) can improve their knowledge over time, thus, the temporal/sequential information is an important factor in predicting student performance. Thai-Nghe et al. [2011] proposed using three-mode tensor factorization (on student/task/time) instead of matrix factorization (on student/task) to take the temporal effect into account.

Inspired from the idea in Rendle et al. [2010], which used matrix factorization with Markov chains to model sequential behavior of the user in e-commerce area, and also inspired from the personalized forecasting methods [Thai-Nghe et al. 2011], we propose a novel approach, *tensor factorization forecasting*, to model the sequential effect in predicting student performance. Thus, we bring together the advantages of both forecasting and factorization techniques in this work. The proposed approach can be used not only for predicting student performance but also for recommending the tasks to the students, as well as for the other domains (e.g. recommender systems) in which the sequential effect should be taken into account.

2. RELATED WORK

Many works can be found in [Romero and Ventura 2006; Baker and Yacef 2009; Romero et al. 2010] but most of them relied on traditional classification/regression techniques. Concretely, Cen et al. [2006] proposed a semi-automated method for improving a cognitive model called Learning Factors Analysis that combines a statistical model, human expertise and a combinatorial search; Thai-Nghe et al. [2009] proposed to improve the student performance prediction by dealing with the class imbalance problem, using support vector machines (i.e., the ratio between passing and failing students is usually skewed); Yu et al. [2010] used linear support vector machines together with feature engineering and ensembling techniques for predicting student performance. These methods work well in case we have enough meta data about students and tasks.

In student modeling, Corbett and Anderson [1995] proposed the Knowledge Tracing model, which is widely used in this domain. The model assumes that each skill has four parameters: 1) initial (or prior) knowledge, which is the probability that a particular skill was known by the student before interacting with the tutoring systems; 2) learning rate, which is the probability that student's knowledge changes from unlearned to learned state after each learning opportunity; 3) guess, which is the probability that a student can answer correctly even if he/she does not know the skill associated with the problem; 4) slip, which is the probability that a student makes a mistake (incorrect answer) even if he/she knows the required skills. To apply the knowledge tracing model for predicting student performance, the four parameters need to be estimated either by using Expectation Maximization method [Chang et al. 2006] or by using Brute-Force method [Baker et al. 2008]. Pardos and Heffernan [2010] propose a variant of knowledge tracing by taking individualization into account. These models explicitly take into account the “slip” and “guess” latent factors.

Recently, researchers have proposed using recommender system techniques (e.g. matrix factorization) for predicting student performance [Thai-Nghe et al. 2010; Toscher and Jährer 2010]. The authors have shown that predicting student performance can be considered as rating prediction since the *student*, *task*, and *performance* would become *user*, *item*, and *rating* in recommender systems, respectively; Extended from these works, Thai-Nghe et al. [2011] proposed tensor factorization models to take into account the sequential effect (for modeling how student knowledge changes over time). Thus, the authors have modeled the student performance as a 3-dimensional recommender system problem on (*student*, *task*, *time*).

In this work, the problem setting is similar to our previous work [Thai-Nghe et al. 2011], however, we introduce two new methods - tensor factorization forecasting models - for predicting student performance.

3. PREDICTING STUDENT PERFORMANCE (PSP)

The problem of predicting student performance is to predict the likely performance of a student for some exercises (or part thereof such as for some particular steps) which we call the *tasks*. The task could be to solve a particular step in a *problem*, to solve a whole problem or to solve problems in a *section* or *unit*, etc.

Detailed descriptions can be found in [Thai-Nghe et al. 2011]. Here, we are only interested in three features, e.g. student ID, task ID, and time ID.

More formally, let S be a set of students, I be a set of tasks, and $P \subseteq \mathbb{R}$ be a range of possible performance scores. Let $\mathcal{D}^{train} \subseteq (S \times I \times P)^*$ be a sequence of observed student performances and $\mathcal{D}^{test} \subseteq (S \times I \times P)^*$ be a sequence of unobserved student performances. Furthermore, let

$$\pi_p : S \times I \times P \rightarrow P, \quad (s, i, p) \mapsto p$$

and

$$\pi_{s,i} : S \times I \times P \rightarrow S \times I, \quad (s, i, p) \mapsto (s, i)$$

be the projections to the performance measure and to the student/task pair. Then the problem of student performance prediction is, given \mathcal{D}^{train} and $\pi_{s,i}(\mathcal{D}^{test})$, to find

$$\hat{p} = \hat{p}_1, \hat{p}_2, \dots, \hat{p}_{|\mathcal{D}^{test}|}$$

such that

$$err(p, \hat{p}) := \sum_{l=1}^{|\mathcal{D}^{test}|} (p_l - \hat{p}_l)^2$$

is minimal with $p := \pi_p(\mathcal{D}^{test})$. Some other error measures could also be considered.

As discussed in Thai-Nghe et al. [2011], the problem of predicting student performance can be *i) casted as rating prediction task in recommender systems* since s, i and p would be *user, item* and *rating*, respectively, and *ii) casted as forecasting problem* (illustrated in Figure 1b-top) to deal with the potentially sequential effects (e.g. describing how students gain experience over time) which is discussed in this work. An illustration of predicting student performance which takes the data sequence into account is presented in Figure 1a. Figure 1b-bottom is an example of representing student performance data in a three-mode tensor.

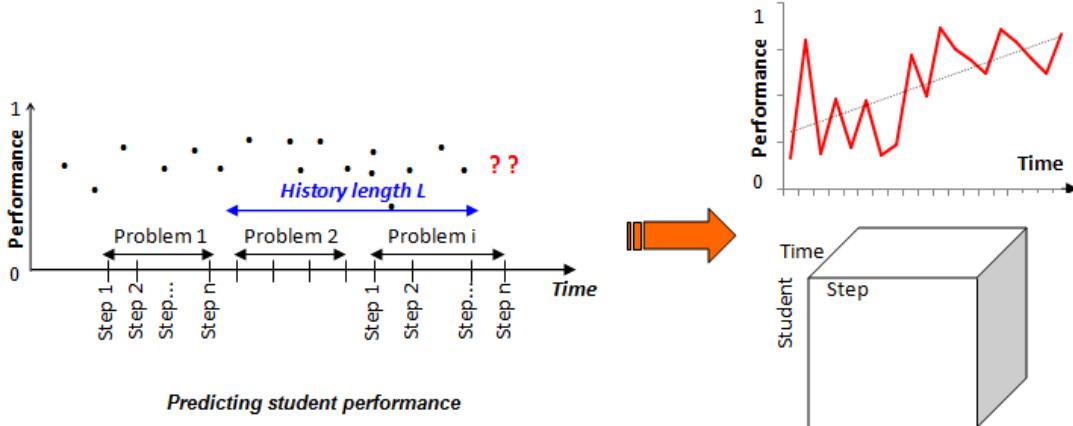


Fig. 1. An illustration of casting predicting student performance as forecasting problem, which uses all historical performance data controlled by the history length L to forecast the next performance

4. TENSOR FACTORIZATION FORECASTING

In this work, we will use three-mode tensor factorization which is a generalization of matrix factorization. Given a three-mode tensor \mathbf{Z} of size $U \times I \times T$, where the first mode describes U students, the second mode describes I tasks (problems), and the third mode describes the time. Then \mathbf{Z} can be written as a sum of rank-1 tensors by using CANDECOM-PARAFAC [Carroll and Chang 1970; Harshman 1970; Kolda and Bader 2009]:

$$\mathbf{Z} \approx \sum_{k=1}^K \lambda_k w_k \circ h_k \circ q_k \quad (1)$$

where \circ is the outer product; $\lambda_k \in \mathbb{R}^+$; and each vector $w_k \in \mathbb{R}^U$, $h_k \in \mathbb{R}^I$, and $q_k \in \mathbb{R}^T$ describes the latent factor vectors of the student, task, and time, respectively (see the articles [Kolda and Bader 2009; Dunlavy et al. 2011] for details). In this work, these latent factors are optimized for root mean squared error (RMSE) using stochastic gradient descent [Bottou 2004].

As mentioned in the literature, “the more the learners study the better the performance they get”, and the knowledge of the learners cumulates over time, thus the temporal effect is an important factor to predict the student performance. We adopt the ideas in the previous works [Dunlavy et al. 2011]², [Thai-Nghe et al. 2011; Thai-Nghe et al. 2011] to incorporate forecasting model into the factorization process, which we call tensor factorization forecasting.

For simplification purpose, we apply the moving average approach (the unweighted mean of the previous n data points [Brockwell and Davis 2002]) with a period L on the time mode. The performance of student u given task i is predicted by:

$$\hat{p}_{uiT^*} = \sum_{k=1}^K w_{uk} h_{ik} \Phi_{T^*k} \quad (2)$$

where

$$\Phi_{T^*k} = \frac{\sum_{t=T^*-L}^{T^*-1} q_{tk} p_t}{L} \quad (3)$$

where T^* is the current time in the sequence; q_{tk} and p_t are the time latent factor and the student performance of the previous time, respectively; L is the number of steps in the history to be used by the model (refer back to Figure 1 to see the value of L). We call this method **TFMAF** (Tensor Factorization - Moving Average Forecasting).

As shown in [Toscher and Jahrer 2010; Thai-Nghe et al. 2011], the prediction result can be improved if one employs the biased terms into the prediction model. In educational setting, those biased terms are “student bias” which models how good a student is (i.e. how likely is the student to perform a task correctly), and “task bias” which models how difficult/easy the task is (i.e. how likely is the task to be performed correctly). To take into account the “student bias” and “task bias”, the prediction function (2) now becomes:

$$\hat{p}_{uiT^*} = \mu + b_u + b_i + \sum_{k=1}^K w_{uk} h_{ik} \Phi_{T^*k} \quad (4)$$

where μ is the global average (average performance of all students and tasks in \mathcal{D}^{train}):

$$\mu = \frac{\sum_{p \in \mathcal{D}^{train}} p}{|\mathcal{D}^{train}|} \quad (5)$$

²This work used tensor factorization for link prediction

b_u is student bias (average performance of student u deviated from the global average):

$$b_u = \frac{\sum_{p^u \in \mathcal{D}^{train}} (p^u - \mu)}{|p^u \in \mathcal{D}^{train}|} \quad (6)$$

and b_i is task bias (average performance on task i deviated from the global average):

$$b_i = \frac{\sum_{p^i \in \mathcal{D}^{train}} (p^i - \mu)}{|p^i \in \mathcal{D}^{train}|} \quad (7)$$

Moreover, in e-commerce area, Rendle et al. [2010] have used matrix factorization with Markov chains to model sequential behavior by learning a transition graph over items that is used to predict the next action based on the recent actions of a user. The authors proposed using previous “basket of items” to predict the next “basket of items” with high probabilities that the users might want to buy. However, in educational environment, one natural fact is that the performance of the students not only depend on the recent knowledge (e.g. the knowledge in the previous problems or lessons, which act as “previous basket of items”) but also depend on the cumulative knowledge in the past that the students have studied. Thus, we need to adapt this method by using all previous performances which are controlled by history length L (see Figure 1) for forecasting the next performance.

The Φ_{T^*k} in equation (3) now becomes:

$$\Phi_{T^*k} = \frac{\sum_{t=T^*-L}^{T^*-1} h'_{tk} q_{tk} p_t}{L} \quad (8)$$

where h'_{tk} is the latent factor of the previous solved task in the sequence. We call this method **TFF** (Tensor Factorization Forecasting).

5. EVALUATION

In this section, we first present two real-world data sets, then we describe the baselines for comparison. We show how we set up the models, and finally, the results of tensor factorization forecasting are discussed.

5.1 Data sets

We use 2 real world data sets which are collected from the Knowledge Discovery and Data Mining Challenge 2010³. These data sets, originally labeled “Algebra 2008-2009” and “Bridge to Algebra 2008-2009” will be denoted “Algebra” and “Bridge” for the remainder of this paper. Each data set is split into a train and a test partition as described in Table I. The data represents the log files of interactions between students and the tutoring system. While students solve math related problems in the tutoring system, their activities, success and progress indicators are logged as individual rows in the data sets.

Table I. Original data sets

Data set	#Attributes	#Instances
Algebra-2008-2009 train	23	8,918,054
Algebra-2008-2009 test	23	508,912
Bridge-to-Algebra-2008-2009 train	21	20,012,498
Bridge-to-Algebra-2008-2009 test	21	756,386

The central element of interaction between the students and the tutoring system is the *problem*. Every problem belongs into a hierarchy of *unit* and *section*. Furthermore, a problem consists of many individual

³<http://pslcdatashop.web.cmu.edu/KDDCup/>

steps such as calculating a circle's area, solving a given equation, entering the result and alike. The field *problem view* tracks how many times the student already saw this problem. The other attributes we have not used in this work. Target of the prediction task is the *correct first attempt* (CFA) information which encodes whether the student successfully completed the given step on the first attempt (CFA = 1 indicates correct, and CFA = 0 indicates incorrect). The prediction would then encode the certainty that the student will succeed on the first try.

As presented in Thai-Nghe et al. [2010], these data sets can be mapped to *user*, *item*, and *rating* in recommender systems. The student becomes the *user*, and the correct first attempt (CFA) becomes the *rating*, bounded between 0 and 1. The authors also presented several options that can be mapped to the *item*. In this work, the *item* refers to a *solving-step*, which is a combination (concatenation) of *problem hierarchy* (PH), *problem name* (PN), *step name* (SN), and *problem view* (PV). The information of *student*, *task*, and *performance* is summarized in Table II.

Table II. Information of students, tasks (solving-steps), and performances (CFAs)

Data set	#Student (as User)	#Task (as Item)	#Performance (as Rating)
Algebra	3,310	1,416,473	8,918,054
Bridge	6,043	887,740	20,012,498

5.2 Evaluation metric and model setting

Evaluation metric: The root mean squared error (RMSE) is used to evaluate the models.

$$RMSE = \sqrt{\frac{\sum_{ui \in \mathcal{D}^{test}} (p_{ui} - \hat{p}_{ui})^2}{|\mathcal{D}^{test}|}} \quad (9)$$

Baselines: We use the *global average* as a baseline, i.e. predicting the average of the target variable from the training set. The proposed methods are compared with other methods such as *student average* (user average in recommender systems), *biased-student-task* (this method originally is user-item-baseline in Koren [2010]). Moreover, we also compare the proposed approach with *matrix factorization* (MF) since previous works [Toscher and Jahrer 2010; Thai-Nghe et al. 2010] have shown that MF can produce promising results. For MF, the mapping of *user* and *item* as the following:

$$\begin{aligned} student &\mapsto user; \\ Problem\ hierarchy\ (unit,\ section),\ problem\ name,\ step\ name,\ problem\ view &\mapsto item; \\ performance &\mapsto rating \end{aligned}$$

Hyper parameter setting: Hyper parameter search was applied to determine the hyper parameters⁴ for all methods (e.g. optimizing the RMSE on a holdout set). We will report later the hyper parameters for some typical methods (in Table IV). Please note that we have not performed the significance test (t-test) because the real target variables of the two data sets from KDD Challenge 2010, until now, have not been published yet. We have to submit the results to the KDD Challenge 2010 website to get the RMSE score. Thus, all the results reported in this study are the RMSE score from this website (it is still opened for submission after the challenge). Of course, one can use the internal split (e.g. splitting the training set to sub-train and sub-test) but we have not experimented in this way since we would like to see how good the results of our approach are compared to the other approaches on the given data sets.

⁴Using similar approach described in [Thai-Nghe et al. 2010]

Dealing with cold-start problem: To deal with the “new user” (new student) or “new item” (new task), e.g., those that are in the test set but not in the train set, we simply provide the global average score for these new users or new items. However, using more sophisticated methods, e.g. in [Gantner et al. 2010], can improve the prediction results. Moreover, in the educational environment, the cold-start problem is not as harmful as in the e-commerce environment where the new users and new items appear every day or even hour, thus, the models need not to be re-trained continuously.

5.3 Results

To justify why forecasting method can be a choice in predicting student performance (especially embedding in the factorization process) and how the sequential (temporal) information affects to the performance of the learners, we plot the student performance on the y -axis and the problem ID (in sequence) on the x -axis. However, in the experimental datasets, the true target variable (the actual performance) for each single step is encoded by binary values, i.e., 0 (incorrect) and 1 (correct), thus, the student performance does not show the trend line when we visualize these data sets.

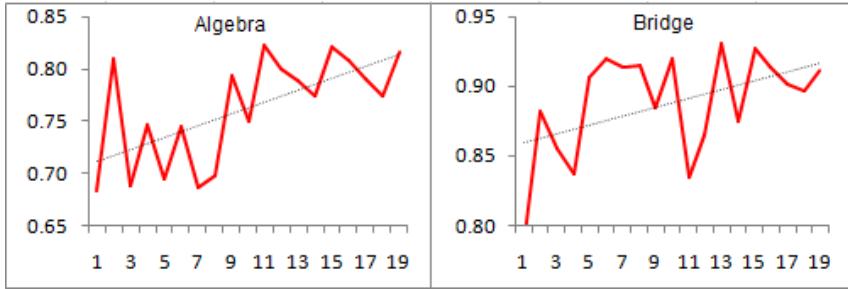


Fig. 2. Sequential effect on the student performance: y -axis is the average of correct performances and x -axis is the sequence of problems (ID) aggregated from the steps. Typical results of Unit 1 and Section 1 of Algebra and Bridge datasets

We aggregate the performance of all steps in the same problem to a single value and plot the aggregated performance to Figure 2. From this, we can see the sequential effect on the sequence of solving problems (from left to right). The average performance increases with the trend line, which implicitly means that forecasting methods are appropriate to cope with predicting student performance. Please note that by aggregating, we will come up with new data sets and the task now is to predict/forecast the whole problem instead of predicting/forecasting the single step in that problem. This work is, however, out of the scope of this paper, so we leave the experimental results on these new aggregated data sets for future work.

Also, in these specific data sets, the actual target variable (the actual performance) is encoded by 0 (incorrect) and 1 (correct), so we modify the equations (3) and (8) to avoid the zero value of the factor product. The Φ_{T^*k} in equation (3) now becomes:

$$\Phi_{T^*k} = \frac{\sum_{t=T^*-L}^{T^*-1} q_{tk}((p_t - 0.5) \cdot 2)}{L} \quad (10)$$

and the Φ_k in equation (8) now becomes:

$$\Phi_{T^*k} = \frac{\sum_{t=T^*-L}^{T^*-1} h'_{tk} q_{tk}((p_t - 0.5) \cdot 2)}{L} \quad (11)$$

However, other modifications on these specific data sets can also be used.

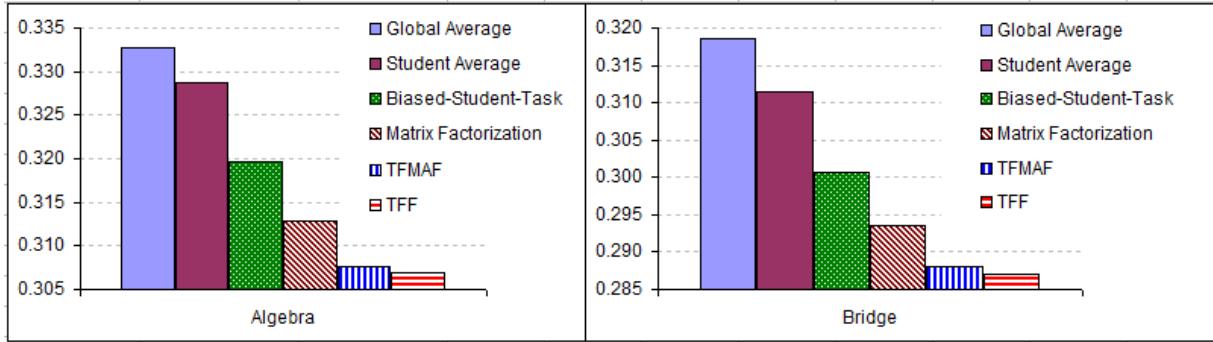


Fig. 3. RMSE results of taken into account the temporal effect using tensor factorization which factorize on student/solving-step/time.

Figure 3 presents the RMSE of the tensor factorization forecasting methods which factorize on the student (as *user*), solving-step (as *item*), and the sequence of solving-step (as *time*). The results of the proposed methods show improvement compared to the others. Moreover, compared with matrix factorization which does not take the temporal effect into account, the tensor factorization methods have also improved the prediction results. These results may implicitly reflect the natural fact that we mentioned before: “the knowledge of the student improves over time”. However, the results of TFF has a small improvement compared to TFMAF method.

Table III presents the RMSE of the proposed methods and the well-known Knowledge Tracing [Corbett and Anderson 1995] which estimates the parameters by using Brute-Force (BF) [Baker et al. 2008], on Bridge data set. Since this data set is quite large, it is intractable when using Expectation Maximization (EM) method [Chang et al. 2006]. The tensor factorization forecasting models have significant improvements compared to the Knowledge Tracing model. However, the comparison with other methods, e.g. Performance Factors Analysis [Pavlik et al. 2009] and Prior Per Student [Pardos and Heffernan 2010], is leaved for future work.

Table III. RMSE of Knowledge Tracing vs. Tensor Factorization Forecasting models

Data set	Knowledge Tracing (BF)	TFMAF	TFF
Algebra	0.30561	0.30398	0.30159
Bridge	0.30649	0.28808	0.28700

For referencing, we report the hyper parameters found via cross-validation and approximation of running time in Table IV. Although the training time of TFF is high (e.g. ≈ 15 hours on Algebra) but in educational environment where the models need not to be retrained continuously, this running time is not an issue.

Table IV. Hyper parameters and running time. β is learning rate, λ is regularization term, K is the number of latent factors, #iter is the number of iterations, and L is the history length.

Method	Data set	Hyper parameters	Train (min.)	Test (sec.)
Matrix Factorization	Algebra	$\beta=0.005$, #iter=120, K=16, $\lambda=0.015$	16.83	0.15
	Algebra	$\beta=0.015$, #iter=30, K=16, $\lambda=0.015$, L=8	108.84	9.17
	Algebra	$\beta=0.001$, #iter=60, K=16, $\lambda=0.015$, L=10	908.71	15.11
TFMAF	Bridge	$\beta=0.01$, #iter=80, K=64, $\lambda=0.015$	40.15	0.34
	Bridge	$\beta=0.005$, #iter=20, K=64, $\lambda=0.015$, L=10	629.07	51.06
	Bridge	$\beta=0.0015$, #iter=60, K=16, $\lambda=0.005$, L=5	466.01	6.61

6. DISCUSSION AND CONCLUSION

Predicting student performance is an important task in educational data mining, where we can give the students some early feedbacks to help them improving their study results. A good and reliable model which accurately predicts the student performance may replace the current standardized tests, thus, reducing the pressure on teaching and learning for examinations as well as saving a lot of time and effort for both teachers and students.

From educational point of view, the learner's knowledge improves and cumulates over time, thus, sequential effect is an important information for predicting student performance. We have proposed a novel approach - tensor factorization forecasting - which incorporates the forecasting technique into the factorization model to take into account the sequential effect.

Indeed, factorization techniques outperform other state-of-the-art collaborative filtering techniques [Koren 2010]. They belong to the family of latent factor models which aim at mapping users (students) and items (tasks) to a common latent space by representing them as vectors in that space. The performance of these techniques are promising even we do not know the background knowledge of the domain (e.g. the student/task attributes). Moreover, we use just two or three features such as student ID, task ID and/or time, thus, the memory consumption and the human effort in pre-processing can be reduced significantly while the prediction quality is reasonable. Experimental results have shown that a combination of factorization and forecasting methods can perform nicely compared to previous works which only use factorization techniques.

Another advantage of this approach is that we can personalize the prediction for each student given the task, and thus, besides predicting student performance, one could use the proposed methods to recommend the tasks (exercises) to students when building a personalized learning system.

A simple forecasting technique, which is moving average, was incorporated into the factorization model. However, applying more sophisticated forecasting techniques, e.g. Holt-Winter [Chatfield and Yar 1988; Dunlavy et al. 2011], may produce better results.

ACKNOWLEDGMENTS

The first author was funded by the "Teaching and Research Innovation Grant" project of Cantho university, Vietnam. Tomáš Horváth is also supported by the grant VEGA 1/0131/09.

REFERENCES

- BAKER, R. S., CORBETT, A. T., AND ALEVEN, V. 2008. More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing. In *Proceedings of the 9th International Conference on Intelligent Tutoring Systems*. Springer-Verlag, Berlin, Heidelberg, 406–415.
- BAKER, R. S. AND YACEF, K. 2009. The state of educational data mining in 2009: A review and future visions. *Journal of Educational Data Mining (JEDM)* 1, 1, 3–17.
- BOTTOU, L. 2004. Stochastic learning. In *Advanced Lectures on Machine Learning*, O. Bousquet and U. von Luxburg, Eds. Lecture Notes in Artificial Intelligence, LNAI 3176. Springer Verlag, Berlin, 146–168.
- BROCKWELL, P. J. AND DAVIS, R. A. 2002. *Introduction to Time Series and Forecasting*. Springer.
- CARROLL, J. AND CHANG, J.-J. 1970. Analysis of individual differences in multidimensional scaling via an n-way generalization of eckart-young decomposition. *Psychometrika* 35, 283–319.
- CEN, H., KOEDINGER, K., AND JUNKER, B. 2006. Learning factors analysis a general method for cognitive model evaluation and improvement. In *Intelligent Tutoring Systems*. Vol. 4053. Springer Berlin Heidelberg, 164–175.
- CETINTAS, S., SI, L., XIN, Y., AND HORD, C. 2010. Predicting correctness of problem solving in its with a temporal collaborative filtering approach. In *International Conference on Intelligent Tutoring Systems*. 15–24.
- CHANG, K., BECK, J., MOSTOW, J., AND CORBETT, A. 2006. A bayes net toolkit for student modeling in intelligent tutoring systems. In *Proceedings of International Conference on Intelligent Tutoring Systems (ITS 2006)*. Springer, 104–113.
- CHATFIELD, C. AND YAR, M. 1988. Holt-winters forecasting: Some practical issues. *Special Issue: Statistical Forecasting and Decision-Making. Journal of the Royal Statistical Society. Series D (The Statistician)* 37, 2, 129–140.

- CORBETT, A. T. AND ANDERSON, J. R. 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction* 4, 253–278.
- DUNLAVY, D. M., KOLDA, T. G., AND ACAR, E. 2011. Temporal link prediction using matrix and tensor factorizations. *ACM Trans. Knowl. Discov. Data* 5, 10:1–10:27.
- FENG, M., HEFFERNAN, N., AND KOEDINGER, K. 2009. Addressing the assessment challenge with an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction* 19, 3, 243–266.
- GANTNER, Z., DRUMOND, L., FREUDENTHALER, C., RENDLE, S., AND SCHMIDT-TIEME, L. 2010. Learning attribute-to-feature mappings for cold-start recommendations. In *Proceedings of the 10th IEEE International Conference on Data Mining (ICDM 2010)*. IEEE Computer Society.
- HARSHMAN, R. A. 1970. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-modal factor analysis. *UCLA Working Papers in Phonetics* 16, 1, 84.
- KOEDINGER, K., BAKER, R., CUNNINGHAM, K., SKOGSHOLM, A., LEBER, B., AND STAMPER, J. 2010. A data repository for the edm community: The pslc datashop. In *Handbook of Educational Data Mining*, C. Romero, S. Ventura, M. Pechenizkiy, and R. Baker, Eds. Lecture Notes in Computer Science. CRC Press.
- KOLDA, T. G. AND BADER, B. W. 2009. Tensor decompositions and applications. *SIAM Review* 51, 3, 455–500.
- KOREN, Y. 2010. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Trans. Knowl. Discov. Data* 4, 1, 1–24.
- PARDOS, Z. A. AND HEFFERNAN, N. T. 2010. Using hmms and bagged decision trees to leverage rich features of user and skill from an intelligent tutoring system dataset. *KDD Cup 2010: Improving Cognitive Models with Educational Data Mining*.
- PAVLIK, P. I., CEN, H., AND KOEDINGER, K. R. 2009. Performance factors analysis –a new alternative to knowledge tracing. In *Proceeding of the 2009 Conference on Artificial Intelligence in Education*. IOS Press, Amsterdam, The Netherlands, 531–538.
- RENDLE, S., FREUDENTHALER, C., AND SCHMIDT-TIEME, L. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web (WWW'10)*. ACM, New York, USA, 811–820.
- ROMERO, C. AND VENTURA, S. 2006. *Data Mining in E-learning*. WIT Pr Computational Mechanics.
- ROMERO, C., VENTURA, S., PECHENIZKIY, M., AND BAKER, R. S. 2010. *Handbook of Educational Data Mining*. Chapman and Hall/CRC Data Mining and Knowledge Discovery Series.
- THAI-NGHE, N., BUSCHE, A., AND SCHMIDT-TIEME, L. 2009. Improving academic performance prediction by dealing with class imbalance. In *Proceeding of 9th IEEE International Conference on Intelligent Systems Design and Applications (ISDA'09)*. Pisa, Italy, IEEE Computer Society, 878–883.
- THAI-NGHE, N., DRUMOND, L., HORVATH, T., KROHN-GRIMBERGHE, A., NANOPoulos, A., AND SCHMIDT-TIEME, L. 2011. Factorization techniques for predicting student performance. In *Educational Recommender Systems and Technologies: Practices and Challenges (In press)*, O. C. Santos and J. G. Boticario, Eds. IGI Global.
- THAI-NGHE, N., DRUMOND, L., KROHN-GRIMBERGHE, A., AND SCHMIDT-TIEME, L. 2010. Recommender system for predicting student performance. In *Proceedings of the 1st Workshop on Recommender Systems for Technology Enhanced Learning (RecSysTEL 2010)*. Vol. 1. Elsevier's Procedia CS, 2811 – 2819.
- THAI-NGHE, N., GANTNER, Z., AND SCHMIDT-TIEME, L. 2010. Cost-sensitive learning methods for imbalanced data. *Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN 2010)*.
- THAI-NGHE, N., HORVATH, T., AND SCHMIDT-TIEME, L. 2011. Personalized forecasting student performance. In *Proceedings of the 11th IEEE International Conference on Advanced Learning Technologies (ICALT 2011) (to appear)*. IEEE CS.
- TOSCHER, A. AND JAHRER, M. 2010. Collaborative filtering applied to educational data mining. *KDD Cup 2010: Improving Cognitive Models with Educational Data Mining*.
- YU, H.-F., LO, H.-Y., ..., AND LIN, C.-J. 2010. Feature engineering and classifier ensemble for kdd cup 2010. *KDD Cup 2010: Improving Cognitive Models with Educational Data Mining*.

Analyzing Participation of Students in Online Courses Using Social Network Analysis Techniques

Reihaneh Rabbany k., Mansoureh Takaffoli and Osmar R. Zaïane,
Department of Computing Science, University of Alberta, Canada
rabbanyk,takaffol,zaiane@ualberta.ca

There is a growing number of courses delivered using e-learning environments and their online discussions play an important role in collaborative learning of students. Even in courses with a few number of students, there could be thousands of messages generated in a few months within these forums. Manually evaluating the participation of students in such case is a significant challenge, considering the fact that current e-learning environments do not provide much information regarding the structure of interactions between students. There is a recent line of research on applying social network analysis (SNA) techniques to study these interactions. And it is interesting to investigate the practicability of SNA in evaluating participation of students. Here we propose to exploit SNA techniques, including community mining, in order to discover relevant structures in social networks we generate from student communications but also information networks we produce from the content of the exchanged messages. With visualization of these discovered relevant structures and the automated identification of central and peripheral participants, an instructor is provided with better means to assess participation in the online discussions. We implemented these new ideas in a toolbox, named Meerkat-ED. Which prepares and visualizes overall snapshots of the participants in the discussion forums, their interactions, and the leader/peripheral students. Moreover, it creates a hierarchical summarization of the discussed topics, which gives the instructor a quick view of what is under discussion. We believe exploiting the mining abilities of this toolbox would facilitate fair evaluation of students' participation in online courses.

1. INTRODUCTION

There is a growing number of courses delivered using e-learning environments, especially in postsecondary education, using computer-supported collaborative learning (CSCL) tools, such as Moodle ,WebCT and Blackboard . Online asynchronous discussions in these environments play an important role in collaborative learning of students. It makes them actively engaged in sharing information and perspectives by interacting with other students [Erlin et al. 2009]. There is a theoretical emphasis in CSCL on the role of threaded discussion forums for online learning activities. Even basic CSCL tools enable the development of these threads where the learners could access text, revise it or reinterpret it; which allow them to connect, build, and refine ideas, along with stimulating deeper reflection [Calvani et al. 2009]. There could be thousands of messages generated in a few months within these forums, containing long discussion threads bearing many interactions between students. Therefore the CSCL tools should provide a means to help instructors for evaluating participation of students and analyzing the structure of these interactions; which otherwise could be very time consuming, if not impossible, for the instructors to be done manually.

Up to now, current CSCL tools do not provide much information regarding the participation of students and structure of interactions between them in discussion threads. In many cases, only some statistical information is provided such as frequency of postings, which is not a useful measure for interaction activity [Erlin et al. 2009]. This means that the instructors who are using these tools, do not have access to convenient indicators that would allow them to evaluate the participation and interaction in their classes [Willging 2005]. Instructors usually have to monitor the discussion threads manually which is hard, time consuming, and prone to human error. On the other hand, there exists a large body of research on studying the participation of students in such discussion threads using traditional research methods: content analysis, interviews,

survey observations and questionnaires [de Laat et al. 2007]. These methods try to detect the activities that students are involved in while ignoring the relations between students. For example, content analysis methods, as the most common traditional methods, provide deep information about specific participants. However, they neglect the relationships between the participants while their focus is on the content, not on the structure [Willging 2005]. In order to fully understanding the participation of students, we need to understand their patterns of interactions and answer questions like who is involved in each discussion, who is the active/peripheral participant in a discussion thread [de Laat et al. 2007]. Nurmela et al. 1999 demonstrated the practicality of social network analysis methods in CSCL, as a method for obtaining information about relations and fundamental structural patterns. Moreover, there is a recent line of work on applying social network analysis techniques for evaluating the participation of students in online courses like works done by Sundararajan 2010, Calvani et al. 2009, de Laat et al. 2007, Willging 2005, Laghos and Zaphiris 2006, and Erlin et al. 2009. The major challenges these works tried to tackle are: extracting social networks from asynchronous discussion forums (might require content analysis), finding appropriate indicators for evaluating participation (from education's point of view) and measuring these indicators using social network analysis. As clarified in the related works, Section 2, none of these works provides a complete or specific toolbox for analyzing discussion threads. However, they attempted to address one of these challenges to some extent.

Here, we elaborate on the importance of social network analysis for mining structural data in the field of computer science and its applicability to the domain of education. for monitoring and evaluating participation of students in online courses. We propose Meerkat-ED, a specific and practical toolbox for analyzing interactions of students in asynchronous discussion forums of online courses. Meerkat-ED analyzes the structure of these interactions using social network analysis techniques including community mining. It prepares and visualizes overall snapshots of participants in the discussion forums, their interactions, and the leader/peripheral students in these discussions. Moreover, it analyzes the content of the exchanged messages in this discussions by building an information network of terms and using community mining techniques to identify the topics discussed. Meerkat-ED creates a hierarchical summarization of these discussed topics in the forums, which gives the instructor a quick view of what is under discussion in these forums. It further illustrates how much each student has participated in these topics, by showing his/her centrality in the discussions on that topic, the number of posts, replies, and the portion of terms used by that student in the discussions. In the following, we first introduce some basic backgrounds of social network analysis and elaborate on its applications in the context of on-line Education. We then present Meerkat-ED – our solution for social network analysis of online courses in Section 3 and illustrate its practicability on our own case study data in Section 4.

2. BACKGROUND AND RELATED WORKS

Social networks are formally defined as a set of actors or network members whom are tied by one or more type of relations [Marin and Wellman 2010]. The actors are most commonly persons or organizations, however, they could be any entities such as web pages, countries, proteins, documents, etc. There could also be many different types of relationships, to name a few, collaborations, friendships, web links, citations, information flow, etc. [Marin and Wellman 2010]. These relations represented by the edges in the network connecting the actors and may have a direction (shows the flow from one actor to the other) and a strength (shows how much, how often, how important).

Unlike proponents of attribute based social sciences, social network analysts argue that causation is not located in the individuals, but in the social structure [Marin and Wellman 2010]. Social network analysis is the study of this structure. Rooted in sociology, nowadays, social network analysis has became an interdisciplinary area of study, including researchers from anthropology, communications, computer science, education, economics, criminology, management science, medicine, political science, and other disciplines [Marin and Wellman 2010]. Social network analysis examines the structure and composition of ties in the network to provides insights into: 1) understanding the central actors in the network (prestige); 2) detecting

the individuals with the most outgoing connections (influence), the most incoming connections (prominence), and the least connections (outlier); 3) identifying the proportion of possible ties that actually exist (density); 4) tracking the actors that are involved in passing information through the network (path length); 5) finding the actors that are communicating more often with each other (community), etc. The availability and growth of large datasets of information networks makes community mining a very challenging research topic in social networks analysis. There has been a considerable amount of work done to detect communities in social networks [Palla et al. 2005], [Newman and Girvan 2004], [Chen et al. 2009], etc.

2.1 Social Network Analysis of Asynchronous Discussions in Online Courses

In order to apply social network analysis techniques to assess participation of students in an e-learning environment, we need to first extract the social network from the e-learning course. Then we consider which measures show an effective participation, and finally report these measures in an appropriate way. Here, we give an overview of the previous works related to each of these phases.

Extraction of Social Network. CSCL tools record log files that contain the detailed actions that occurring within them. Hence, log files include information about the activity of the participants in the discussion forums [Nurmela et al. 1999]. de Laat et al. 2007, Willging 2005, Erlin et al. 2009 and Laghos and Zaphiris 2006 used these log files to extract the social network underneath of discussion threads. Laghos et al. stated that they considered each message as directed to all participants in that discussion thread while others considered it as only directed to the previous message. Gruzd and Haythornthwaite 2008 and 2009, proposed an alternative and more complicated way of extracting social networks, called named network. They argue that using this common method (connecting a poster to the previous poster in the thread) would result in losing much of the connections. Their approach briefly is: first using named entity recognition to find the nodes of the network, then counting the number of times that each name is mentioned in posts by others to obtain the ties, and finally weighting these ties by the amount of information exchanged in the posts. However, their final reported results are not that promising and even obtaining those results requires many manual corrections during the process. Regarding what we should consider as the participation in extracting the social network, Hrastinski 2008 suggested that apart from writing, there are other indicators of participation like accessing the e-learning environment, reading posts or the quantity and quality of the writing. However, all of these methods extracted networks just based on posts (writing level).

Measuring the Effectiveness of Participation. Daradoumis et al. 2006 defined high level weighted (showing the importance) indicators to represent collaboration learning process; task performance, group functioning, social support, and help services. They further divided these indicators to skills and sub-skills, and assigned every sub-skill to an action. For example, group functioning is divided into: active participation behavior, task processing, communication processing, etc. On the other hand, communication processing is itself divided into more sub-skills: clarification, evaluation, illustration, etc. and clarification is then mapped to the action of changing description of a document or url. In the education context, Calvani et al. 2009 defined 9 indicators for measuring the effectiveness of participation to compare different groups within a

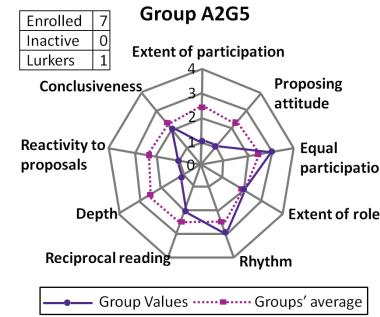


Fig. 1: This nanogram illustrates a comparison of participation of one group (blue lines) with the average participation of other groups (red lines) using the nine indicators defined by Calvani et al. 2009. Figure reproduced from [Calvani et al. 2009].

class; extent of participation (number of messages), proposing attitude (number of messages with proposal label), equal participation (variance of messages for users), extent of role (portion of roles used), rhythm (variance of daily messages per day), reciprocal reading (portion of messages that have been read), depth (average response depth), reactivity to proposal (number of direct answers to messages with proposal label) and conclusiveness (number of messages with conclusion label); all summarized in a nonagon graph which shows the group interactions relatively to the mean behavior of all groups (Figure 1). However, for measuring the effectiveness of participation, most of the previous works simply used general social network measures (different centrality measures, betweenness, etc.), available in one of the common general social network analysis toolboxes. Sundararajan 2010, de Laat et al. 2007, Willging 2005, Erlin et al. 2009 used UCINET [UCINET] and Laghos and Zaphiris 2006 used NetMiner [NetMiner].

3. SOCIAL NETWORK ANALYSIS FOR EDUCATION: MEERKAT-ED

In this section, we illustrate the practicability of social network analysis in evaluating participation of students in online discussion threads. We present our specific social network analysis toolbox, named Meerkat-ED, to analyze online courses. Meerkat-ED is designed for assessing the participation of students in asynchronous discussion forums of online courses. It analyzes the structure of interactions between students in these discussions using social network analysis techniques. It exploit community mining techniques in order to discover relevant structures in social networks generated from student communications and also information networks produced from the content of the exchanged messages. With visualization of these discovered relevant structures and the automated identification of central and peripheral participants, an instructor is provided with better means to assess participation in the online discussions.

Meerkat-ED prepares and visualizes overall snapshots of participants in the discussion forums, their interactions, and the leader/peripheral students. It creates a hierarchical summarization of the topics discussed in the forums using community mining, which gives the instructor a quick view of what is under discussion in these forums. It further illustrates how much each student has participated on these topics, by showing his/her centrality in the discussions on that topic, the number of posts, replies, and the portion of terms used by that student in discussions on the topic. Meerkat-ED builds and analyzes two kinds of networks out of the discussion forums: *social network of the students* where links represent correspondence, and *network of the phrases* used in the discussions where links represent co-occurrence of phrases in the same sentence. Interpreting the first network shows the interaction structure of the students participated in the discussions. Furthermore, centrality of students in this network corresponds to their leadership in the discussions. Interpreting terms network depicts the terms used in the discussion and the relations between these terms. Finding the hierarchical communities in this network demonstrates the topics addressed in the discussions. Choosing each of these topics outlines the students who participated in that topic and the extent of their participation.

3.1 Interpreting Students Interaction Network

Interpreting the network of interaction between students helps instructors monitor the interaction structure of students, and examine which students are the leaders in given discussions and who are the peripheral students. Here, we first describe how the network is extracted based on the information from the discussion threads. Then, we continue by bringing an analysis of leadership of the students based on their centrality in this network. The student network shows the interaction between students in the discussion forums, where the nodes represent students of the course and edges are the interaction between these students (i.e. messages exchanged). The edges are weighted by the number of messages passed between the two incident students. This network could be built both directed or undirected (chosen by the instructor); in the directed model, each message is considered connecting the author of the message to the author of its parent message. The leadership and influence of students in the discussions could be compared by examining the centrality of

nodes corresponding to them in the network; as the nodes' centrality measures their relative importance within a network. Moreover, students could be ranked more explicitly in a concentric centrality graph in which the more central/powerful the node is, the closer it is to the center (Figure 4).

3.2 Interpreting Term Network

Interpreting the term network, depicts the terms used in the discussions and the relation between these terms. Moreover, finding the hierarchical communities in this network, demonstrates the topics exchanged in the discussions. Furthermore, choosing each of these topics would outline the students who participated in that topic and the extent of their participation. In the term network, nodes represent noun phrases occurring in the discussions; and edges show the co-occurrence of these terms in the same sentence. Each co-occurrence edge contains the messages in which its incident terms occurred together; and is weighted by the number of sentences in which these terms co-occurred. For building this network, we need to first extract the noun phrases from the discussions, then build the network by setting the extracted phrases as nodes and checking their co-occurrence in all the sentences of every message for creating the edges.

We have used the OpenNlp toolbox [OpenNlp] for extracting noun phrases out of discussions. OpenNlp is a set of natural language processing tools for performing sentence detection, tokenization, pos-tagging, chunking, parsing, and etc. Using sentence detector in OpenNlp, we first segmented the content of messages to their consisting sentences. The tokenizer was used to break down those sentences to words. Having the tokenized words, we used the Part-Of-Speech tagger to determine their grammatical tags – whether they are noun, verbs, adjective, etc. Then using the chunker, we grouped these words to the phrases, and we picked the detected noun phrases, which are sequences of words surrounding at least one noun and functioning as a single unit in the syntax. For obtaining better sets of terms to represent the content of the discussions, pruning on the extracted noun phrases was necessary. We removed all the stopwords, and split the phrases that have stop word(s) within into two different phrases. For example the phrase "privacy and confidentiality" is split into two terms: "privacy", and "confidentiality". To avoid having duplicates, the first characters were converted to lower case (if the other characters of the phrase are in lowercase) and plurals to singular forms (if the singular form appeared in the content). For instance "Patients" would be "patients" then "patient". As final modification, we removed all the noun phrases that just occurred once; which would prune most of unwanted phrases.

The term Network could be further analyzed to group the terms co-occurring mostly together. These groups represent the different topics discussed in the messages and could be obtained by detecting the communities in the term network. This idea is similar to work done in Chen et al. 2008. For creating the hierarchy of the topics, we applied a community mining algorithm repeatedly to divide one of the current connected components of the network, until the size of all components is smaller than a threshold, or the division of any of the components would result in a loose partitioning. We used FastModularity [Clauset et al. 2004] as the community detection algorithm, however it could be any other community mining approach. Based on the detected term communities, the participation of students and how wide their participation are could be validated. In other words, students who participated in different topics could be considered more active than students that just talked about a smaller number of topics. This evaluation could be examined by selecting each student and checking how many topics he/she participated in.

4. CASE STUDY

In this section, we validate the feasibility of Meerkat-ED and illustrate its practical application on our own case study data. Here, Meerkat-ED is used for visualizing, monitoring and evaluating participation of students in the discussion forums. The data set we have used is obtained from a postsecondary course. The course titled Electronic Health Record and Data Analysis, and was offered in Winter 2010 at University of Alberta. The permission to use the anonymized course data for research purposes was obtained from all the

students registered in the course, at the end of the semester so as not to bias the communications taking place. This data is further anonymized by assigning fake names to students and replacing any occurrence of first, last or user name of the students in the data (including content of the messages in discussion forums) with the assigned fake name. We also removed all email addresses from the data.

In the chosen course, as is also usual in other courses, the instructor initiated different discussion threads. For each thread he posted a question or provided some information and asked students to discuss the issue. Consequently students posted subsequent messages in the thread, responding to the original question or to the response of other students. This course was offered using Moodle which is a widely-used course management system. Moodle like other CSCL tools, enables interaction and collaborative construction of content, mostly using its Forum tool which is a place for students to share their ideas [Moodle]. Only using Moodle, to evaluate student participation the instructor is limited to shallow means such as the number of posts per thread and eventually the apparent size of messages. The instructor would have to manually monitor the content of each interaction to measure the extent of individual participation, which is hard, time consuming and even unrealistic in large classes or forums with large volume, where different participants can be assigned to moderate different discussions and threads.

To assess participation, we build and analyze two kinds of networks from these information: the social network of students and the network of the terms used by them. The instructor of the course denoted the usefulness of the results of these analysis in evaluating the participation of students in the course. Like in [Sundararajan 2010] where the authors noted that using SNA it was easy to identify the workers and the lurkers in the class, in this case study, the instructor reported that using Meerkat-ED it was easy to have an overview of the whole participation and it was possible to identify influential students in each thread as well as identify quiet students or unvoiced opinions, something that would have been impossible with the simple statistics provided by Moodle. More importantly, focusing on the relationships in the graph one can identify the real conduit for information rather than simply basing assessment of participation on message size or frequency of submissions. Learners who place centrally in the network as conduit for the information control and can cause more knowledge exchange which is desirable in an online class. Regardless of the frequency of messages, their size or content, if they do not have influence, their authors remain marginal and sit on the periphery of the network (See Figure 4). This role of conduit of information versus marginal students can change during the course of the semester or from one discussed thread to the other. The systematic analysis of centrality of participants per topic discussed provided by Meerkat-ED allowed a better assessment of the participation of learners at each discussion topic level.

4.1 Interpreting Students Interaction Network

As explained before, first of all we have to extract the students network from the discussion thread. Figure 2 shows the visualized network of students in the course. The size of the nodes corresponds to their degree centrality in the network – the number of incident edges. This means that the bigger a node is, the more messages the student represented by that node sent and received. The thickness of the edges in the network represents the weight of interactions which is based on the number of messages in the interaction of communicating students. Choosing an edge would bring up a pop up window that shows these messages as illustrated in Figure 3. The next step is to analysis the leadership of the students based on their centrality in this network. The nodes' centrality is depicted by the size of the nodes in the visualized network as illustrated in Figure 2. Moreover, students could be ranked more explicitly in a concentric centrality graph in which the more central/powerful the node is, the closer it is to the center, as presented in Figure 4.

4.2 Interpreting Term Network

For this specific course, we extract the term network from the discussion forum. Figure 5 presents the visualization of this term network, where the size of the nodes represents the frequency of their corresponding

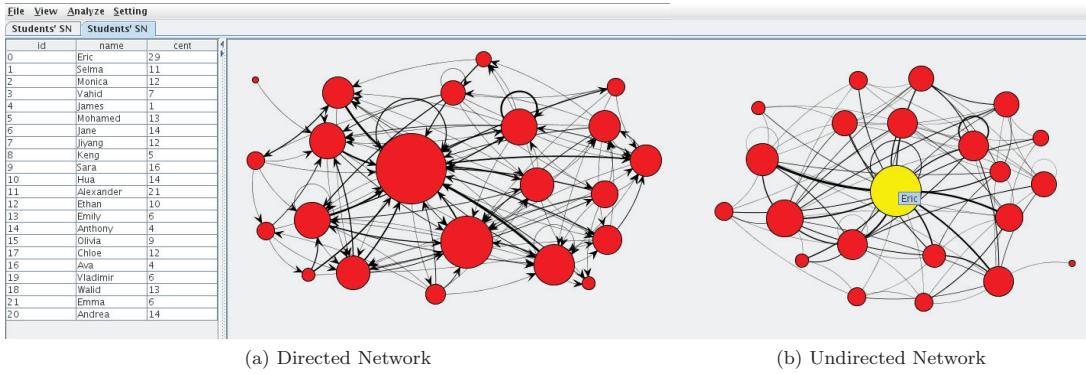


Fig. 2: Visualized Student Network: The left panel lists the students in the course. The right panel shows the social network of interaction of students in the course. The size of nodes corresponds to their centrality/leadership in the discussions. The width of edges represents the weight of communication between incident nodes.

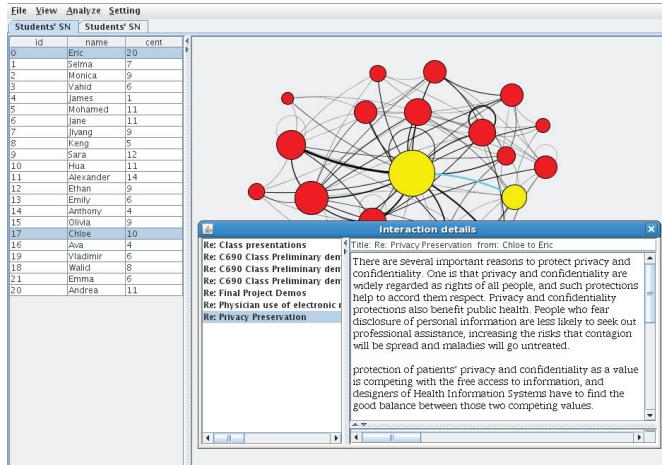


Fig. 3: Visualization of messages in an interaction: the interaction window shows the messages passed between nodes incident to the selected edge: Chloe and Eric. Selecting each message from the left panel would show its title, sender, receiver and content.

terms and the thickness of edges represents the weight of the co-occurrences (i.e. the number of sentences in which incident terms occurred together). Selecting an edge would show these messages as illustrated in Figure 6. In this visualization the instructor would see a list of the discussion threads in the course while selecting any set of those discussions/messages would bring up the corresponding term network, along with the list of terms occurring in them and the list of students that participated in these selected set of discussions/messages. Selecting any of these terms would show the students that used that term. Likewise, selecting any of the students would outline the terms used by that student, as illustrated in Figure 5; which is highlighting the terms discussed by the student named Chloe. The difference between the number of

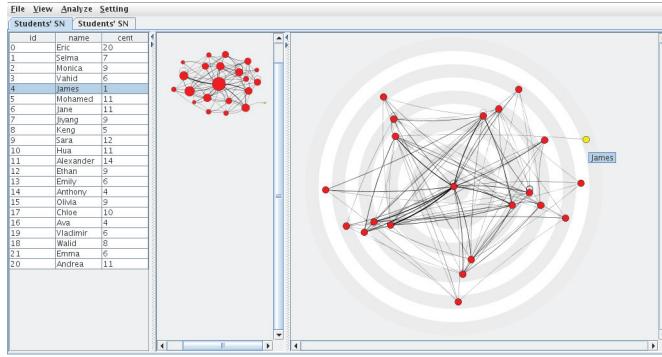


Fig. 4: Comparing centrality of students: the students closer to the center are more central in the student network, i.e., have participated more in the discussions of the course. Likewise, the further from the center, the less the student was active; here James is the least active student in the discussions and is placed on the outer circle.

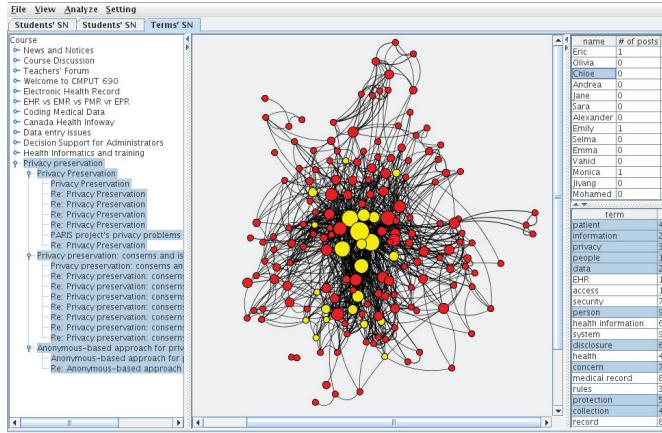


Fig. 5: Visualized Term Network: The left panel lists the discussion threads in the course. The middle panel shows the network of terms in the selected set of discussions. The upper right panel shows list of students participated in the selected discussions, along with some statistics about their participation such as number of posts, replies, etc. The bottom right panel shows the terms used in these discussions. Selecting each student, would outline the terms used by that student.

terms discussed by the students could help the instructor to compare the participations of the students: students who discuss more terms participate more as well. In order to further analyzed the term Network, as explained before, we group the terms co-occurring mostly together. Figure 7a shows the detected topics (term communities) in the network given in Figure 5. The green nodes show the representative nodes of communities. Each representative node, contains 10 most central terms of the terms in the community it represents. The size of the representative nodes corresponds to the number of terms in their communities; while the size of the leaf nodes, terms, is related to their frequency, same as the term network. Similar to the term network, here also one could select a set of terms, usually within a topic, to see who participated in a discussion with that topic and to what extent, as illustrated in Figure 7b.

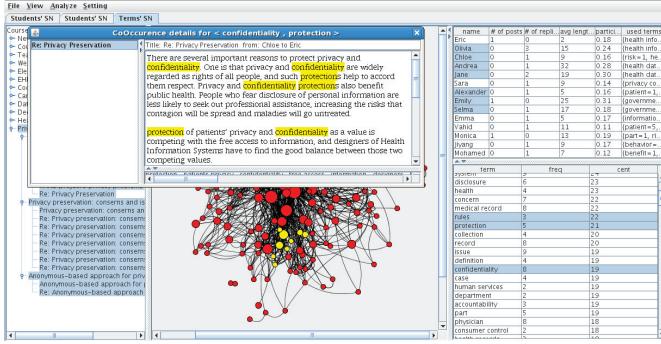


Fig. 6: Co-occurrence of terms: selecting a co-occurrence edge would bring up a pop up window that shows the messages these incident terms co-occurred together in, highlighting the corresponding terms in the content.

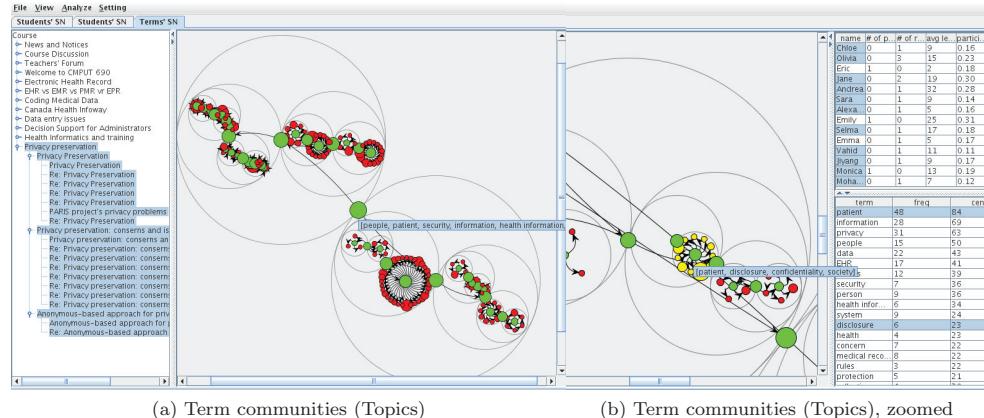


Fig. 7: Term communities (Topics): The gray circles outline the communities boundaries and the green nodes represent the community representatives. Each community representative is accompanied with its top 10 phrases in its community. These could be seen in the tooltip in the figure. Selecting each topic, would outline the students who participated in a discussion with that topic, and the terms in that topic. Here, the topic is roughly about "patient, disclosure, confidentiality and society". Moreover, students who participated in this topic and their contribution could be seen in the upper right panel.

5. CONCLUSIONS

In this paper we elaborated the importance of social network analysis for mining structural data and its applicability in the domain of education. We introduced social network analysis and community mining for studying the structure in relational data. We illustrated the place and need for social network analysis in study of the interaction of users in e-learning environments; then summarized some recent studies in this area. We also proposed Meerkat-ED, a specific and practical toolbox for analyzing students interactions in asynchronous discussion forums. Our toolbox prepares and visualizes overall snapshots of participants in the discussion forums, their interactions, and the leaders/peripheral students. Moreover, it creates a hierarchical summarization of the discussed topics, which gives the instructor a quick view of what is under discussion.

It further illustrates individual student participation in these topics, measured by their centrality in the discussions on that topic, their number of posts, replies, and the portion of terms used by them. We believe exploiting the mining abilities of this toolbox would facilitate fair evaluation of students' participation in online courses.

REFERENCES

- BLACKBOARD. http://en.wikipedia.org/wiki/Blackboard_Learning_System.
- CALVANI, A., FINI, A., MOLINO, M., AND RANIERI, M. 2009. Visualizing and monitoring effective interactions in online collaborative groups. *British Journal of Educational Technology*.
- CHEN, J., ZAÏANE, O. R., AND GOEBEL, R. 2008. An unsupervised approach to cluster web search results based on word sense communities. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*. IEEE Computer Society, Washington, DC, USA, 725–729.
- CHEN, J., ZAÏANE, O. R., AND GOEBEL, R. 2009. Detecting communities in large networks by iterative local expansion. In *CASoN*. 105–112.
- CLAUSET, A., NEWMAN, M. E. J., AND MOORE, C. 2004. Finding community structure in very large networks. *Phys. Rev. E* 70, 066111.
- DARADOUMIS, T., MARTÍNEZ-MONÉS, A., AND XHAFÀ, F. 2006. A layered framework for evaluating on-line collaborative learning interactions. *Int. J. Hum.-Comput. Stud.* 64, 7, 622–635.
- DAVIS, R. H. 1981. Social network analysis - an aid in conspiracy investigations. *FBI Law Enforcement Bulletin* 50, 12, 11–19. The use of social network analysis in the conduct of investigations of conspiracies is described.
- DE LAAT, M., LALLY, V., LIPPONEN, L., AND SIMONS, R.-J. 2007. Investigating patterns of interaction in networked learning and computer-supported collaborative learning: A role for social network analysis. *International Journal of Computer-Supported Collaborative Learning* 2, 1, 87–103.
- ERLIN, YUSOF, N., AND RAHMAN, A. A. 2009. Students' interactions in online asynchronous discussion forum: A social network analysis. In *International Conference on Education Technology and Computer*. IEEE Computer Society, Los Alamitos, CA, USA, 25–29.
- GRUZD, A. AND HAYTHORNTHWAITE, C. A. 2008. The analysis of online communities using interactive content-based social networks. extended abstract. In *Proceedings of the American Society for Information Science and Technology (ASIS&T) Conference*. Columbus, OH, USA, 523–527.
- GRUZD, A. A. 2009. Automated discovery of social networks in online learning communities. Ph.D. thesis, University of Illinois at Urbana-Champaign.
- HRASTINSKI, S. 2008. What is online learner participation? a literature review. *Computers & Education* 51, 4, 1755–1765.
- KEELING, M. J. AND EAMES, K. T. 2005. Networks and epidemic models. *Journal of the Royal Society, Interface / the Royal Society* 2, 4, 295–307.
- LAGHOS AND ZAPHIRIS. 2006. Sociology of student-centred e-learning communities: A network analysis. In *IADIS international conference*. e-Society, Dublin, Ireland.
- MARIN, A. AND WELLMAN, B. forthcoming, 2010. *Handbook of Social Network Analysis*. Sage, Chapter Social Network Analysis: An Introduction.
- MOODLE. <http://en.wikipedia.org/wiki/Moodle>.
- NETMINER. <http://www.netminer.com/NetMiner/>.
- NEWMAN, M. E. J. 2004. Detecting community structure in networks. *Eur. Phys. J.B* 38, 321–330.
- NEWMAN, M. E. J. AND GIRVAN, M. 2004. Finding and evaluating community structure in networks. *Physical Review E* 69.
- NURMELA, K., LEHTINEN, E., AND PALONEN, T. 1999. Evaluating cscl log files by social network analysis. *Computer Support for Collaborative Learning*.
- OPENNLP. <http://opennlp.sourceforge.net/README.html>.
- PALLA, G., DERENYI, I., FARKAS, I., AND VICSEK, T. 2005. Uncovering the overlapping community structure of complex networks in nature and society. *Nature* 435, 814–818.
- SUNDARARAJAN, B. 2010. Emergence of the most knowledgeable other (mko): Social network analysis of chat and bulletin board conversations in a cscl system. *Electronic Journal of e-Learning* 8, 191–208.
- UCINET. <http://www.analytictech.com/ucinet/>.
- WEBCT. <http://en.wikipedia.org/wiki/WebCT>.
- WILLGING, P. A. 2005. Using social network analysis techniques to examine online interactions. *US-China Education Review* 2, 9, 46–56.

A Machine Learning Approach for Automatic Student Model Discovery

Nan Li and Noboru Matsuda and William W. Cohen and Kenneth R. Koedinger, Carnegie Mellon University

Student modeling is one of the key factors that affects automated tutoring systems in making instructional decisions. A student model is a model to predict the probability of a student making errors on given problems. A good student model that matches with student behavior patterns often provides useful information on learning task difficulty and transfer of learning between related problems, and thus often yields better instruction. Manual construction of such models usually requires substantial human effort, and may still miss distinctions in content and learning that have important instructional implications. In this paper, we propose an approach that automatically discovers student models using a state-of-art machine learning agent, SimStudent. We show that the discovered model is of higher quality than human-generated models, and demonstrate how the discovered model can be used to improve a tutoring system's instruction strategy.

1. INTRODUCTION

A student model is a set of *knowledge components (KC)* encoded in intelligent tutors to model how students solve problems. The set of KCs includes the component skills, concepts, or percepts that a student must acquire to be successful on the target tasks. For example, a KC in algebra can be how students should proceed given problems of the form $Nv=N$ (e.g. $3x = 6$). It provides important information to automated tutoring systems in making instructional decisions. Better student models match with real student behavior. They are capable of predicting task difficulty and transfer of learning between related problems, and often yield better instruction. Traditional ways to construct student models include structured interviews, think-aloud protocols, rational analysis, and so on. However, these methods are often time-consuming, and require expert input. More importantly, they are highly subjective. Previous studies [Koedinger and Nathan 2004; Koedinger and McLaughlin 2010] have shown that human engineering of these models often ignores distinctions in content and learning that have important instructional implications. Other methods such as Learning Factor Analysis (LFA) [Cen et al. 2006] apply an automated search technique to discover student models. It has been shown that these automated methods are able to find better student models than human-generated ones. Nevertheless, one key limitation of LFA is that it carries out the search process only within the space of human-provided factors. If a better model exists but requires unknown factors, LFA will not find it.

To address this issue, we propose a method that automatically discovers student models not depending on human-provided factors. The system uses a state-of-art machine learning agent, SimStudent [Matsuda et al. 2009], to acquire skill knowledge. Each skill corresponds to a KC that students need to learn. The model then labels each observation of a real student based on skill application. We evaluated the approach in algebra using real student data. Experiment results show that the discovered model fits with real student data better than human-generated models, and provides useful insights in finding better instructional methods.

In the following sections, we begin with a review of SimStudent. Next, we report experiment results that demonstrate the benefits of the SimStudent model over the human-generated model. After this, we discuss the possible improvements that can be made to a tutoring system suggested by the SimStudent model. In closing, we discuss related work as well as future directions for this work.

Author's address: Nan Li; email: nli1@cs.cmu.edu; Noboru Matsuda; email: Noboru.Matsuda@cs.cmu.edu; William W. Cohen; email: wcohen@cs.cmu.edu; Kenneth R. Koedinger; email: koedinger@cmu.edu; 5000 Forbes Ave, Pittsburgh, PA 15232

2. A REVIEW OF SIMSTUDENT

SimStudent is an intelligent agent that inductively learns skills to solve problems from demonstrated solutions and from problem solving experience. It is a realization of programming by demonstration [Lau and Weld 1998] and employs inductive logic programming [Muggleton and de Raedt 1994] as one of its learning mechanisms. For more details about SimStudent, please refer to Matsuda et al. [2009].

2.1 Input

SimStudent is given a set of *feature predicate symbols* and a set of *operator symbols* as prior knowledge before learning. Each predicate is a boolean function that describes relations among objects in the domain (e.g. (*has-coefficient -3x*)). Operators specify basic manipulations (e.g. (*add 1 2*), (*coefficient -3x*)) that SimStudent can apply to objects in the problem solving interface, like numbers or character strings. Operators are divided into two groups, domain-independent operators and domain-specific operators. Domain-independent operators (e.g. (*add 1 2*)) are basic manipulations that are applicable across multiple domains. Real students usually have knowledge of these simple skills prior to class. Domain-specific operators (e.g. (*add-term 5x-5 5*), (*coefficient -3x*)), on the other hand, are more complicated manipulations that are associated with only one domain. From a learner modeling perspective, beginning students may not know domain-specific operators and thus providing such operators to SimStudent may produce learning behavior that is distinctly different from human students [Matsuda et al. 2009]. Operators in SimStudent (whether domain-independent or domain-specific) have no explicit encoding of preconditions and effects. This matches the intuition that human students often “know how” without “knowing when”.

During the learning process, given the current state of the problem (e.g., $-3x = 6$), SimStudent first tries to find an appropriate *production rule* (skill knowledge acquired by SimStudent) that proposes a plan for the next step (e.g. (*coefficient -3x ?coef*) (*divide ?coef*))). If it finds one, it executes the plan, performs an action in the system interface, and waits for feedback from the human user/author/tutor. If the user provides positive feedback, SimStudent continues to the next step. If not, SimStudent records this negative feedback and may try again. If SimStudent does not find a production rule that generates a correct action, it requests a demonstration of the next step, which the user performs in the interface. SimStudent uses any negative feedback to modify existing productions. It uses the next-step demonstration, if provided, to learn a new production rule.

In the experiments we describe here, the user/author/tutor role is simulated by a hand-engineered algebra tutor [Koedinger et al. 1995], which provides SimStudent with feedback and next-step demonstrations as needed via an API. For each demonstrated step, the user/tutor specifies a tuple of $\langle \text{selection}, \text{action}, \text{input} \rangle$ (SAI tuple) for a skill. SimStudent is given a *skill label* (e.g. “divide”) generated by the cognitive tutor, which corresponds to the type of skill applied. “Selection” in the SAI tuple (e.g. $-3x$ and 6 for $-3x = 6$) is associated with elements in the graphical user interfaces (GUI). It shows where a “focus of attention” is —that is, where to look for relevant information. “Action” (e.g. entering some text) indicates what action to take with the “input” (e.g. (*divide -3*) for problem $-3x = 6$). In this example, the full plan might be to first retrieve coefficient and then to divide by it (e.g. (*coefficient -3x ?coef*) (*divide ?coef*))), but the tutor only demonstrates the final action (e.g., (*divide -3*)) to SimStudent. Taken together, the given information forms one record indexed by the skill label, $R=\langle \text{label}, \langle \text{selection}, \text{action}, \text{input} \rangle \rangle$ (e.g. $R=\langle \text{divide}, \langle (-3x, 6), \text{input text}, (\text{divide } -3) \rangle \rangle$). In learning, SimStudent acquires one production rule for each skill label, based on the set of associated records gathered at that point.

2.2 Production Rules

The output of the learning agent is represented as production rules. Each production rule corresponds to one knowledge component. The left side of Figure 1 shows an example of a production rule learned by SimStudent. A production rule indicates “when” (precondition) to apply a rule to what information found “where” (focus of attention (FoA)) in the interface and “how” (operator sequence) the problem state should be changed. For example, the rule shown in the left side of Figure 1 would be read as “given a left-hand side

- | | |
|--|---|
| <ul style="list-style-type: none"> • Original: • Skill divide (e.g. $-3x = 6$) • FoAs: <ul style="list-style-type: none"> ◦ Left side ($-3x$) ◦ Right side (6) • Precondition: <ul style="list-style-type: none"> ◦ Left side ($-3x$) does not have constant term • Operator sequence: <ul style="list-style-type: none"> ◦ Get coefficient (-3) of left side ($-3x$) ◦ Divide both sides with the coefficient (-3) | <ul style="list-style-type: none"> • Extended: • Skill divide (e.g. $-3x = 6$) • FoAs: <ul style="list-style-type: none"> ◦ Left side (-3, $-3x$) ◦ Right side (6) • Precondition: <ul style="list-style-type: none"> ◦ Left side ($-3x$) does not have constant term • Operator sequence: <ul style="list-style-type: none"> ◦ Get coefficient (-3) of left side (-3x) ◦ Divide both sides with the coefficient (-3) |
|--|---|

Fig. 1. Original and extended production rules for divide in a readable format.

(i.e. $-3x$) and a right-hand side (i.e. 6) of the equation, when the left-hand side does not have a constant term, then get the coefficient of the term on the left-hand side and divide both sides by the coefficient.” The focus of attention of the production represents paths through the task-specific GUI interface that retrieve the items needed by the operator sequence. The precondition of a production rule includes a set of feature tests, representing preconditions for applying the rule. The operator sequence specifies a plan to execute.

2.3 Learning Mechanism

SimStudent uses three different learning components for the three parts of the production rules. The first component (the “where learner”) learns how to focus attention on the relevant aspects of the interface by generalizing paths from the element for the interface as a whole to the specific elements of the interface that have the information needed to execute the operator sequence. The elements in the GUI are organized in a tree structure. In the algebra domain, the root node is a table node that links to columns, and each column has multiple cells as children. The “where learner’s” task is to find the right paths in the tree to reach the nodes in the focus-of-attention (e.g. *Cell 11* and *Cell 21*). A FoA (e.g. *Cell 21*) can be reached either 1) by the path to its exact position (e.g. *Cell 21*) in the tree, 2) by a generalized path (e.g. *Cell 2?*, *Cell ??*) to its position. Therefore, given a set of FoAs from positive records, for each position, the “where learner” searches for one least general path that covers all of the FoAs at that position.

The second part of the learning mechanism is a precondition learner (the “when learner”, which acquires the precondition of the production rule using the given feature predicates. The precondition learner utilizes FOIL [Quinlan 1990], an inductive logic programming system that learns Horn clauses from both positive and negative examples expressed as relations. For each rule, the precondition learner creates a new predicate that corresponds to the precondition of the rule, and sets it as the target relation for FOIL to learn. The arguments of the new predicate are associated with the FoAs. Each training record serves as either a positive or a negative example for FOIL. For example, (*precondition-divide -3x 6*) is a positive example for the new predicate (*precondition-divide ?FoA₁ ?FoA₂*). The precondition learner also computes the truthfulness of all predicates bound with all possible permutations of FoA values, and sends it as input to FOIL. Given these inputs, FOIL will acquire a set of clauses formed by feature predicates describing the precondition predicate.

The last component is the operator sequence learner (the “how learner”). For each positive record, R_i , the learner takes the FoAs, $FoAs_i$, as the initial state, and sets the step, $step_i$, as the goal state. We say an operator sequence explains a FoAs-step pair, $\langle FoAs_i, step_i \rangle$, if the system takes $FoAs_i$ as an initial state and yields $step_i$ after applying the operators. For example, with the FoAs-step pair in the example, $\langle (-3x, 6), (divide -3) \rangle$, the operator sequence (*coefficient -3x ?coef*) (*divide ?coef*) is a possible explanation for this pair. The learner searches for the shortest operator sequence that explains all of the $\langle FoAs, step \rangle$ pairs using iterative-deepening depth-first search.

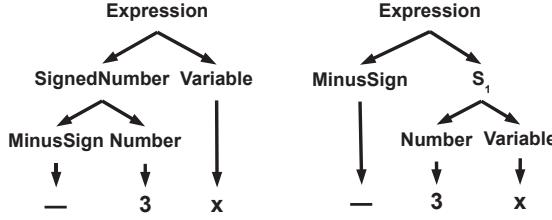


Fig. 2. Correct and incorrect parse trees for $-3x$.

Last, although SimStudent tries to learn one rule for each label, it might fail to do so (e.g., when no operator sequence can explain all records). In that case, SimStudent learns a disjunctive rule just for the last record. This effectively splits the records into two clusters. Later, for each new record, SimStudent tries to acquire a rule for each of the clusters with the new record, and stops whenever it successfully learns a rule with one of the clusters, or creates another new cluster.

2.4 Extending SimStudent to Learn Deep Features

Previous study [Chi et al. 1981] has shown that one of the key differences between experts and novices is that experts view the world in terms of deep features, whereas novices only see shallow features. Recently, we have extended SimStudent to support acquisition of deep features using Li et al. [2010]'s algorithm. They model deep feature learning as a grammar induction problem. In the algebra domain, expressions are modeled with a probabilistic context free grammar (PCFG), and the deep features (e.g., “coefficient”) are intermediate symbols in the grammar rules. Moreover, Li et al. [2010] showed that student errors can be modeled as incorrect parsing, as shown at the right side of Figure 2. Li et al. [2010]'s deep feature learner extends an earlier PCFG learner [Li et al. 2009] to support feature learning and transfer learning.

The input of the system is a set of observation-feature pairs such as $\langle -3x, -3 \rangle$. The output is a PCFG with a designated intermediate symbol in one of the rules set as the target feature. The learning process contains two steps. The system first acquires the grammar using Li et al. [2009]'s algorithm. After that, the feature learner tries to identify an intermediate symbol in one of the rules as the target feature. To do this, the system builds parse trees for all of the observation sequences, and picks the intermediate symbol that corresponds to the most training records as the deep feature. To model transfer learning, Li et al. [2010] further extend the feature learner to acquire PCFGs based on previously acquired knowledge. When the learner is given a new learning task, it first uses the known grammar to build parse trees for each new record in a bottom-up fashion, and stops when there is no rule that could further merge two parse trees into a single tree. The learner then switches to the original learner and acquires new grammar rules as needed. Having acquired the grammar for deep features, when a new problem is given to the system, the learner will extract the deep feature by first building the parse tree of the problem based on the acquired grammar, and then extracting the subsequence associated with the feature symbol from the parse tree as the target feature. However, this model is only capable of learning and extracting deep features without using them to solve problems.

As we have mentioned above, SimStudent is able to acquire production rules in solving complicated problems, but requires a set of operators given as prior knowledge. Some of the operators are domain-specific, and require expert knowledge to build them. On the other hand, the feature learner acquires the deep features that are essential for effective learning, but is limited to information extraction tasks. In order to both reduce the amount of prior knowledge engineering needed for SimStudent and to extend the deep feature learner's capability, we integrated the deep feature learner into SimStudent.

Extending Perceptual Learning. Previously, the FoAs encoded in production rules are always associated with paths to elements in the GUI (such as cells in the algebra example). Intuitively, the deep features discussed above represent perceptual information—however, it is *domain-specific, learned* perceptual infor-

mation. To exploit this information, we extend the perceptual hierarchy for the GUI to further include the most probable parse trees from the learned PCFG in the contents of the leaf nodes. We implement this by appending the parse trees to their associated leaf nodes, marking the appended nodes as type “subcell”. In the algebra example, this extension means that cells representing algebraic expressions (e.g., those corresponding to left-hand sides or right-hand sides of the equation) are linked to parse trees for these expressions. Using $-3x$ as an example, the extended hierarchy includes the parse tree for $-3x$ as shown on the left side of Figure 2 as a subtree connected to the cell node associated with $-3x$. With this extension, the coefficient (-3) of $-3x$ is now explicitly represented in the percept hierarchy. Hence if the extended SimStudent includes this subcell as a FoA in production rules, as shown at the right side of Figure 1, the production rule would no longer need the domain-specific engineered operator “coefficient”.

However, extending the percept hierarchy presents challenges to the original “where learner”. First of all, since the extended subcells are not associated with GUI elements, we can no longer depend on the tutor to specify FoAs for SimStudent. Nor can we simply put all of the subcells in the parse trees as FoAs: if we did, the acquired production rules would contain redundant information that might hurt the generalization capability of the “where learner”. For example, for problem $-3x=6$, among all inserted subcells, only -3 is a relevant FoA in solving the problem. Second, the paths to the relevant FoAs are typically more diverse: for example, for problems $-3x=6$ and $4x=8$, the original where learner would not be able to find one set of generalized paths that explain both training examples, since $-3x$ has eight nodes in its parse tree, while $4x$ has only five. To address these challenges, we extend the original “where learner” to support acquisition of FoAs with redundant and non-fixed length FoA lists.

To do this, SimStudent first includes all of the inserted subcells as candidate FoAs, and calls the operator sequence learner to find a plan that explains all of the training examples. The “where learner” then removes all of the subcells that are not used by the operator sequence from the candidate FoA list. Since all of the training records share the same operator sequence, the number of FoAs remained for each record should be the same. Next, the “where learner” arranges the remained subcell FoAs based on their orderings of being used by the operator sequences. After this process, the “where learner” now has a set of FoA lists that contains fixed number of FoAs ordered in the same fashion. We can then switch to the original “where learner” to find the least general paths for the updated FoA lists. In our example for skill “divide”, as shown at the right side of Figure 1, the FoAs of the production rule would contain three elements, the left-hand side and right-hand side cells which are the same as the original rule, and a coefficient subcell which corresponds to the left child of the variable term. Note that since we removed the redundant subcells, the acquired production rule now works with both $-3x=6$ and $4x=8$.

Extending Precondition Acquisition. In addition to extending the feature learner, we also extend the vocabulary of feature symbols provided to the precondition learner. As implied by its name, the deep feature learner acquires information that reveal essential features of the problem state. It is natural to think that these deep features could also be used in describing desired situations to fire a production rule. Therefore, we construct a set of *grammar features* that are associated with the acquired PCFG. The set of new predicates describe positions of a subcell in the parse tree. For example, we create a new predicate called “is-left-child-of”, which should be true for *(is-left-child-of -3 -3x)* based on the parse tree shown in the left side of Figure 2. Importantly, these new predicates are not domain-specific (although they are specific to the PCFG-based approach to deep feature learning). All of the grammar feature predicates are then included in the set of existing feature predicates for the precondition learner to use later.

3. EXPERIMENT STUDY

3.1 Method

In order to evaluate the effectiveness of the proposed approach, we carried out a study using an algebra dataset. We compared the SimStudent model with a human-generated KC model by first coding the real student steps using the two models, and then testing how well the two model codings fit with real student data.

For the human-generated model, the real student steps were first coded using the “action” label associated with a correct step transaction, where an action corresponds to a mathematical operation(s) to transform an equation into another. As a result, there were nine KCs defined (called the Action KC model) – add, subtract, multiply, divide, distribute, clt (combine like terms), mt (simplify multiplication), and rf (reduce a fraction). Four KCs associated with the basic arithmetic operations (i.e., add, subtract, multiply, and divide) were then further split into two KCs for each, namely a skill to identify an appropriate basic operator and a skill to actually execute the basic operator. The former is called a transformation skill whereas the latter is a typein skill. As a consequence, there were 12 KCs defined (called the Action-Typein KC model). Not all steps in the algebra dataset can be coded with these KC models – some steps are about a transformation that we do not include in the Action KC model (e.g., simplify division). There were 9487 steps that can be coded by both KC models mentioned above. The “default” KC model, which were defined by the productions implemented for the cognitive tutor, has only 6809 steps that can be coded. To make a fair comparison between the “default” and “Action- Typein” KC models, we took the intersection of those 9487 and 6809 steps. As a result, there were 6507 steps that can be coded by both the default and the Action-Typein KC models. We then defined a new KC model, called the Balanced-Action-Typein KC model that has the same set of KCs as the Action-Typein model but is only associated with these 6507 steps, and used this KC model to compare with the SimStudent model.

To generate the SimStudent model, SimStudent was tutored on how to solve linear equations by interacting with a Carnegie Learning Algebra I Tutor like a human student. We selected 40 problems that were used to teach real students as the training set for SimStudent. Given all of the acquired production rules, for each step a real student performed, we assigned the applicable production rule as the KC associated with that step. In cases where there was no applicable production rule, we coded the step using the human-generated KC model (Balanced-Action-Typein). Each time a student encounters a step using some KC is considered as an “opportunity” for that student to show mastery of that KC.

Having finished coding real student steps with both models (the SimStudent model and the human-generated model), we used the Additive Factor Model (AFM) [Cen et al. 2006] to validate the coded steps. AFM is an instance of logistic regression that models student success using each student, each KC, and the KC by opportunity interaction as independent variables,

$$\ln \frac{p_{ij}}{1 - p_{ij}} = \theta_i + \sum_k \beta_k Q_{kj} + \sum_k \beta_k Q_{kj} (\gamma_k N_{ik}) \quad (1)$$

Where:

i. represents a student *i*.

j. represents a step *j*.

k. represents a skill or KC *k*.

p_{ij} . is the probability that student *i* would be correct on step *j*.

θ_i . is the coefficient for proficiency of student *i*.

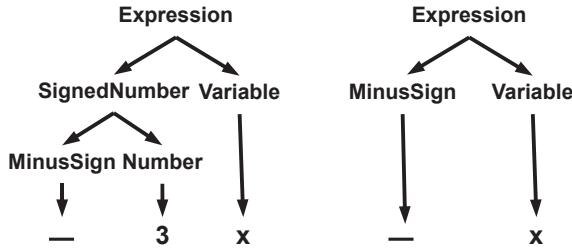
β_k . is coefficient for difficulty of the skill or KC *k*

Q_{kj} . is the Q-matrix cell for step *j* using skill *k*.

γ_k . is the coefficient for the learning rate of skill *k*;

N_{ik} . is the number of practice opportunities student *i* has had on the skill *k*;

We utilized DataShop [Koedinger et al. 2010], a large repository that contains datasets from various educational domains as well as a set of associated visualization and analysis tools, to facilitate the process of evaluation, which includes generating learning curve visualization, AFM parameter estimation, and evaluation statistics including AIC (Akaike Information Criterion) and cross validation.

Fig. 3. Different parse trees for $-3x$ and $-x$.

3.2 Dataset

We analyzed data from 71 students who used an Carnegie Learning Algebra I Tutor unit on equation solving. The students were typical students at a vocational-technical school in a rural/suburban area outside of Pittsburgh, PA. The problems varied in complexity, for example, from simpler problems like $3x=6$ to harder problems like $x/-5+7=2$. A total of 19,683 transactions between the students and the Algebra Tutor were recorded, where each transaction represents an attempt or inquiry made by the student, and the feedback given by the tutor.

3.3 Measurements

To test whether the generated model fits with real student data, we used AIC and a 3-fold cross validation. AIC measures the fit to student data while penalizing over-fitting. We did not use BIC (Bayesian Information Criterion) as the fit metric, because based on past analysis across multiple DataShop datasets, it has been shown that AIC is a better predictor of cross validation than BIC is. The cross validation was performed over three folds with the constraint that each of the three training sets must have data points for each student and KC. We also report the root mean-squared error (RMSE) averaged over three test sets.

3.4 Experiment Result and Implications on Instructional Decision

The SimStudent model contains 21 KCs. Both the AIC (6448) and the cross validation RMSE (0.3997) are lower than the human-generated model (AIC 6529 and cross validation 0.4034). This indicates that the SimStudent model better fits with real student data without over-fitting.

In order to understand whether the differences are significant or not, we carried out two significance tests. The first significance test evaluates whether the SimStudent model is actually able to make better predictions than the human-generated model. During the cross validation process, each student step was used once as the test problem. We took the predicated error rates generated by the two KC models for each step during testing. Then, we compared the KC models' predictions with the real student error rate (0 if the student was correct at the first attempt, and 1 otherwise). After removing ties, among all 6494 student steps, the SimStudent model made a better prediction than the human-generated KC model in 4260 steps. A sign test on this shows that the SimStudent model is significantly ($p < 0.001$) better in predicting real student behavior than the human-generated model. In the second test, due to the random nature of the folding process in cross validation, we evaluated whether the lower RMSE achieved by the SimStudent model was consistent or by chance. To do this, we repeated the cross validation 20 times, and calculated the RMSE for both models. Across the 20 runs, the SimStudent model consistently outperformed the human-generated model. Thus, a paired t-test shows the SimStudent model is significantly ($p < 0.001$) better than the human-generated model. Also note that differences between competitors in the KDD Cup 2010 (<https://pslcdatasshop.web.cmu.edu/KDDCup/Leaderboard>) have also been in this range of thousands

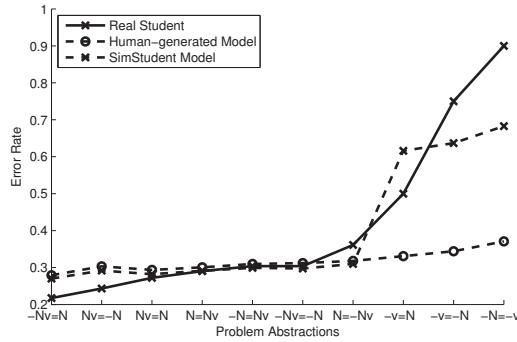


Fig. 4. Error rates for real students and predicted error rates from two student models.

in RMSE. Therefore, we conclude that the SimStudent model is a better student model than the human-generated KC model.

We can inspect the data more closely to get a better qualitative understanding of why the SimStudent model is better and what implications there might be for improved instruction. Among the 21 KCs learned by the SimStudent model, there were 17 transformation KCs and four typein KCs. It is hard to map the SimStudent KC model directly to the expert model. Approximately speaking, the distribute, cht, mt, rf KCs as well as the four typein KCs are similar to the KCs defined in the expert model. The transformation skills associated with the basic arithmetic operators (i.e. add, subtract, multiply and divide) are further split into finer grain sizes based on different problem forms.

One example of such split is that SimStudent created two KCs for division. The first KC (simSt-divide) corresponds to problems of the form $Ax=B$, where both A and B are signed numbers, whereas the second KC (simSt-divide-1) is specifically associated with problems of the form $-x=A$, where A is a signed number. This is caused by the different parse trees for Ax vs $-x$ as shown in Figure 3. To solve $Ax=B$, SimStudent simply needs to divide both sides with the signed number A . On the other hand, since $-x$ does not have -1 represented explicitly in the parse tree, SimStudent needs to see $-x$ as $-1x$, and then to extract -1 as the coefficient. If SimStudent is a good model of human learning, we expect the same to be true for human students. That is, real students should have greater difficulty in making the correct move on steps like $-x=6$ than on steps like $-3x=6$ because of the need to convert (perhaps just mentally) $-x$ to $-1x$. To evaluate this hypothesis, we computed the average error rates for a relevant set of problem types – these are shown with the solid line in Figure 4 with the problem types defined in forms like $-Nv=N$, where the N s are any integer number and the v is a variable (e.g., $-3x=6$ is an instance of $-Nv=N$ and $-6=-x$ is an instance of $-N=-v$). We also calculated the mean of the predicted error rates for each problem type for both the human-generated model and the SimStudent model. Consistent with the hypothesis, as shown in Figure 4, we see that problems of the form $Ax=B$ (average error rate 0.283) are much simpler than problems of the form $-x=A$ (average error rate 0.719). The human-generated model predicts all problem types with similar error rates (average predicted error rate for $Ax=B$ 0.302, average predicted error rate for $-x=A$ 0.334), and thus fails to capture the difficulty difference between the two problem types ($Ax=B$ and $-x=A$). The SimStudent model, on the other hand, fits with the real student error rates much better. It predicts higher error rates (0.633 on average) for problems of the form $-x=A$ than problems of the form $Ax=B$ (0.291 on average).

SimStudent's split of the original division KC into two KCs, simSt-divide and simSt-divide-1, suggests that the tutor should teach real students to solve two types of division problems separately. In other words, when tutoring students with division problems, we should include two subsets of problems, one subset corresponding to simSt-divide problems ($Ax=B$), and one specifically for simSt-divide-1 problems ($-x=A$). We should perhaps also include explicit instruction that highlights for students that $-x$ is the same as $-1x$.

4. RELATED WORK

The objective of this paper is using a machine learning agent, SimStudent, to automatically construct student models. There has been considerable work on comparing the quality of alternative cognitive models. LFA automatically discovers student models, but is limited to the space of the human-provided factors. Other works such as [Pavlik et al. 2009; Villano 1992] are less dependent on human labeling, but may suffer from challenges in interpreting the results. In contrast, the SimStudent approach has the benefit that the acquired production rules have a precise and usually straightforward interpretation. Baffes and Mooney [1996] applies theory refinement to the problem of modeling incorrect student behavior. Other systems [Tatsuoka 1983; Barnes 2005] use Q-matrix to find knowledge structure from student response data. None of the above approaches use simulated students to construct cognitive models.

Other research on creating simulated students [VanLehn et al. 1994; Chan and Chou 1997; Pentti Hietala 1998] also share some resemblance to our work. VanLehn [1990] created a learning system and evaluated whether it was able to learn procedural “bugs” like real students. Biswas et al. [2005]’s system learns causal relations from a conceptual map created by students. None of the above approaches compared the system with learning curve data. To the best of our knowledge, our work is the first combination of the two whereby we use cognitive model evaluation techniques to assess the quality of a simulated learner.

5. CONCLUSION AND FUTURE WORK

In this paper, we introduced an innovative application of a machine-learning agent, SimStudent, for an automatic discovery of student models. An empirical study showed that a SimStudent generated student model was a better predictor of real students learning performance than a human-coded student model. The basic idea is to have SimStudent learn to solve the same problems that human students did and use the productions that SimStudent generated as knowledge components to codify problem-solving steps. We then used these KC coded steps to validate the models prediction. Unlike the human-engineered student model, the SimStudent generated student model has a clear connection between the features of the domain contents and knowledge components. An advantage of the SimStudent approach of student modeling over previous techniques like LFA is that it does not depend heavily on the human-engineered features. SimStudent can automatically discover a need to split a purported KC or skill into more than one skill. During SimStudents learning, a failure of generalization for a particular KC results in learning disjunctive rules. Discovering such disjuncts is equivalent to splitting a KC in LFA, however, whereas human needs to provide potential factors to LFA as the basis for a possible split, SimStudent can learn such factors. The use of the perceptual learning component, implemented using a probabilistic context-free grammar learner, is a key feature of SimStudent for these purposes as we hypothesized that a major part of human expertise, even in academic domains like algebra, is such perceptual learning.

Our evaluation demonstrated that representing the rules SimStudent learns in the student model improves the accuracy of model prediction, and showed how the SimStudent model could provide important instructional implications. Much of human expertise is only tacitly known. For instance, we know the grammar of our first language but do not know what we know. Similarly, most algebra experts have no explicit awareness of subtle transformations they have acquired like the one above (seeing $-x$ as $-1x$). Even though such instructional designers may be experts in a domain they have thus have some blind spots regarding subtle perceptual differences like this one, which may make a real difference for novice learners. A machine learning agent, like SimStudent, can help get past such blind spots by revealing challenges in the learning process that experts may not be aware of.

The current study used a single dataset in a single domain. The generality and validity of the proposed student-modeling technique could be extended by training SimStudent with one dataset and applying a discovered KC model to another dataset. For instance, the experiment dataset was from one high school. An interesting future study would be to examine data from other schools or grade levels, and evaluate the generality of the proposal technique. We should also apply this approach in other domains such as stoichiometry, fraction addition and so on. The Pittsburgh of Science of Learning Centers DataShop contains

over 200 datasets in algebra and other domains that could be used for such cross-dataset or cross-domain validation.

6. ACKNOWLEDGEMENTS

The research reported here was supported by National Science Foundation Award No. DRL-0910176 and the Institute of Education Sciences, U.S. Department of Education, through Grant R305A090519 to Carnegie Mellon University. The opinions expressed are those of the authors and do not represent views of the Institute or the U.S. Department of Education. This work is also supported in part by the Pittsburgh Science of Learning Center, which is funded by the National Science Foundation Award No. SBE-0836012.

REFERENCES

- BAFFES, P. T. AND MOONEY, R. J. 1996. A novel application of theory refinement to student modeling. In *Proceedings of the thirteenth national conference on Artificial intelligence*. AAAI Press, 403–408.
- BARNES, T. 2005. The Q-matrix method: Mining student response data for knowledge. In *Proceedings AAAI Workshop Educational Data Mining*. Pittsburgh, PA, 1–8.
- BISWAS, G., SCHWARTZ, D., LEELAWONG, K., AND VYE, N. 2005. Learning by teaching: A new agent paradigm for educational software. *Applied Artificial Intelligence* 19, 363–392.
- CEN, H., KOEDINGER, K., AND JUNKER, B. 2006. Learning factors analysis - a general method for cognitive model evaluation and improvement. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*. 164–175.
- CHAN, T.-W. AND CHOU, C.-Y. 1997. Exploring the design of computer supports for reciprocal tutoring. *International Journal of Artificial Intelligence in Education* 8, 1–29.
- CHI, M. T. H., FELTOVICH, P. J., AND GLASER, R. 1981. Categorization and representation of physics problems by experts and novices. *Cognitive Science* 5, 2, 121–152.
- KOEDINGER, K. R., ANDERSON, J. R., HADLEY, W. H., AND MARK, M. A. 1995. Intelligent Tutoring Goes to School in the Big City. In *Proceedings of the 7th International Conference on Artificial Intelligence in education*.
- KOEDINGER, K. R., BAKER, R. S., CUNNINGHAM, K., SKOGSHOLM, A., LEBER, B., AND STAMPER, J. 2010. A data repository for the EDM community: The PSLC DataShop.
- KOEDINGER, K. R. AND McLAUGHLIN, E. A. 2010. Seeing language learning inside the math: Cognitive analysis yields transfer. In *Proceedings of the 32nd Annual Conference of the Cognitive Science Society*. Austin, TX, 471–476.
- KOEDINGER, K. R. AND NATHAN, M. J. 2004. The real story behind story problems: Effects of representations on quantitative reasoning. *The Journal of Learning Sciences* 13, 2, 129–164.
- LAU, T. AND WELD, D. S. 1998. Programming by demonstration: An inductive learning formulation. In *Proceedings of the 1999 International Conference on Intelligence User Interfaces*. 145–152.
- LI, N., COHEN, W. W., AND KOEDINGER, K. R. 2010. A computational model of accelerated future learning through feature recognition. In *Proceedings of 10th International Conference on Intelligent Tutoring Systems*. 368–370.
- LI, N., KAMBHAMPATI, S., AND YOON, S. 2009. Learning probabilistic hierarchical task networks to capture user preferences. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*. Pasadena, CA.
- MATSUDA, N., LEE, A., COHEN, W. W., AND KOEDINGER, K. R. 2009. A computational model of how learner errors arise from weak prior knowledge. In *Proceedings of Conference of the Cognitive Science Society*.
- MUGGLETON, S. AND DE RAEDT, L. 1994. Inductive logic programming: Theory and methods. *Journal of Logic Programming* 19, 629–679.
- PAVLIK, P. I., CEN, H., AND KOEDINGER, K. R. 2009. Learning Factors Transfer Analysis: Using Learning Curve Analysis to Automatically Generate Domain Models. In *Proceedings of 2nd International Conference on Educational Data Mining*. 121–130.
- PENTTI HIETALA, T. N. 1998. The competence of learning companion agents. *International Journal of Artificial Intelligence in Education* 9, 178–192.
- QUINLAN, J. R. 1990. Learning logical definitions from relations. *Machine Learning* 5, 3, 239–266.
- TATSUOKA, K. K. 1983. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement*, 345–354.
- VANLEHN, K. 1990. *Mind Bugs: The Origins of Procedural Misconceptions*. MIT Press, Cambridge, MA, USA.
- VANLEHN, K., OHLSSON, S., AND NASON, R. 1994. Applications of simulated students: an exploration. *Journal of Artificial Intelligence in Education* 5, 135–175.
- VILLANO, M. 1992. Probabilistic student models: Bayesian belief networks and knowledge space theory. In *Proceedings of the 2nd International Conference on Intelligent Tutoring Systems*. Heidelberg, 491–498.

Conditions for effectively deriving a Q-Matrix from data with Non-negative Matrix Factorization

MICHEL C. DESMARAIS, Polytechnique Montréal

The process of deciding which skills are involved in a given task is tedious and challenging. Means to automate it are highly desirable, even if only partial automation that provides supportive tools can be achieved. A recent technique based on Non-negative Matrix Factorization (NMF) was shown to offer valuable results, especially due to the fact that the resulting factorization allows a straightforward interpretation in terms of a Q-matrix. We investigate the factors and assumptions under which NMF can effectively derive the underlying high level skills behind assessment results. We demonstrate the use of different techniques to analyse and interpret the output of NMF. We propose a simple model to generate simulated data and to provide lower and upper bounds for quantifying skill effect. Using the simulated data, we show that, under the assumption of independent skills, the NMF technique is highly effective in deriving the Q-matrix. However, the NMF performance degrades under different ratios of variance between subject performance, item difficulty, and skill mastery. The results corroborates conclusions from previous work in that high level skills, corresponding to general topics like World History and Biology, seem to have no substantial effect on test performance, whereas other topics like Mathematics and French do. The analysis and visualization techniques of the NMF output, along with the simulation approach presented in this paper, should be useful for future investigations using NMF for Q-matrix induction from data.

1. INTRODUCTION

The construction of a Q-matrix from data is a highly desirable goal for tutoring systems. Not only would it waive the expertise and labour intensive task of assigning which skills are involved in which task, but it would also offer a more objective and replicable means of getting the correct skill-to-task mapping. Furthermore, it might also allow a more effective means to build Q-matrices, as machine learning methods often outperform humans over a range of complex tasks.

However, the success in achieving this goal remains limited. Nowadays, we find no reliable method to automate the mapping of skills to tasks from data, but some progress has been made.

Working with log data from tutoring systems, data which is characterized by the fact that the knowledge state of the student dynamically changes in the data as the student learns, Cen et al. [2006; 2005] have used a technique known as Learning Factor Analysis (LFA) in order to bring improvements over an initially hand built Q-matrix (also termed a *transfer model*). This technique was shown useful for bringing improvements to the Q-matrix composed of fine-grained skills which are deemed necessary to complete certain exercises.

Inspired from the work of Tatsuoka [1983], Barnes [2006] developed a method of mapping skills to items based on a measure of the fit of a potential Q-matrix to the data. This method and the other methods described below rely on static student knowledge states, as opposed to the dynamically changing knowledge states of the LFA technique. Barnes method is fully automated and it was shown to perform at least as well as Principal Component Analysis for skill clustering analysis. However, it involves an algorithm that does not scale well to a Q-matrix that comprises 20 or more items.

Winters et al. [2005] investigated how a number of standard clustering techniques can effectively match skills to test items. They applied these techniques to a wide array of test outcomes, from SAT topics such as Mathematics, Biology and French, to computer science exams, and to different trivia topics. Their findings show that for skills associated to topics within a single course, for example, the techniques were essentially no better at classifying test items than random clustering. The same conclusion applies for topics like World

history and Biology. However, the techniques were relatively successful at separating items that belongs to totally different topics, such as Mathematics and French.

In this paper, we replicate parts of the study by Winters et al. [2005] and focus on one of the cluster algorithms they used, Non-negative Matrix Factorization (NMF). We use visualization techniques to analyze in greater details the results of the factorization. We propose a model to simulate student data and show that the NMF technique is indeed effective under certain assumptions. We use the simulation data model parameters as a means to quantify and estimate the effect of skills over the observed examinee performance in some of the real data of Winters et al. original study. First, let us give some details about NMF.

2. NON-NEGATIVE MATRIX FACTORIZATION AND Q-MATRIX INTERPRETATION

Non-negative matrix factorization (NMF) decomposes a matrix into two smaller matrices. It is used for dimensionality reduction, akin to Principal Component Analysis and Factor analysis. NMF decomposes a matrix of $n \times m$ positive numbers, \mathbf{V} , as the product of two matrices:

$$\mathbf{V} \approx \mathbf{WH} \quad (1)$$

The matrices \mathbf{W} and \mathbf{H} are respectively $n \times r$ and $r \times m$, where r is called the rank of the factorization. For our purpose, matrix \mathbf{V} represents the observed test outcome data for n question items and m respondents. Therefore, the product of \mathbf{W} and \mathbf{H} reproduces the observed patterns of success/failures of the m examinee to the n items. The matrix \mathbf{W} can be considered as a Q-matrix, whereas \mathbf{H} can be considered as the skills mastery for each m examinee. In the case of a Q-matrix, r represents the number of skills, which can take any value but should normally conform to: $r < nm/(n + m)$ [Lee and Seung 1999].

Let us take an example to better explain NMF in our context. Assume the following Q-matrix, \mathbf{W} , composed of 3 skills and 4 items, and the following skills mastery matrix, \mathbf{H} , for 5 examinees:

$$\mathbf{W} = \begin{matrix} \text{skills} \\ \text{items} \end{matrix} \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \mathbf{H} = \begin{matrix} \text{examinees} \\ \text{skills} \end{matrix} \begin{pmatrix} 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

Given this Q-matrix and the skills mastered by the 5 examinees, the expected results are:

$$\mathbf{V} = \mathbf{WH} = \begin{matrix} \text{examinees} \\ \text{items} \end{matrix} \begin{pmatrix} 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

For example, taking the first item and the first examinee, we have, from \mathbf{W} , that item 1 requires skill 2, but, from \mathbf{H} , we see that examinee 1 only masters skill 1, therefore item 1 is failed by examinee 1. In fact, examinee 1's only success is over item 3 since all other items require either skills 2 or 3.

It is important to emphasize that there are many solutions to $\mathbf{V} = \mathbf{WH}$. For example, the same results as those above can be obtained with different Q-matrix and skills matrix:

$$\begin{matrix} \text{examinees} \\ \text{items} \end{matrix} \begin{pmatrix} 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \end{pmatrix} = \begin{matrix} \text{skills} \\ \text{items} \end{matrix} \begin{pmatrix} 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & 1 \\ \frac{1}{2} & 0 & 0 \end{pmatrix} \begin{matrix} \text{examinees} \\ \text{skills} \end{matrix} \begin{pmatrix} 0 & 2 & 0 & 2 & 2 \\ 0 & 0 & 2 & 0 & 2 \\ 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Notice that the weights are changed as well as the ordering of rows and columns compared to the first solution. Nevertheless, it remains a valid factorization of \mathbf{V} that could be derived by some NMF algorithm.

Indeed, there are many NMF algorithms that were developed since its introduction by Lee and Seung [1999] and they can yield different solutions. We refer the reader to Berry et al. [2007] for a more thorough and recent review of this technique which has gained strong adoption in many different fields.

Whereas the other matrix factorization techniques often impose constraints of orthogonality among factors, NMF imposes the constraint that the two matrices, \mathbf{W} and \mathbf{H} , be non-negative. This constraint makes the interpretation much more intuitive in the context of using this technique for building a Q-matrix. It implies that the skills (latent factors) are additive “causes” that contribute to the success of items, and that they can only increase the probability of success and not decrease it, which makes good sense for skill factors. Note that negative values in \mathbf{W} can be interpreted as misconceptions and would lower the expected score to items, but allowing negative values in the factorization also opens up the space of possible solutions and raises the issue of convergence and of the multiplicity of solutions, making the interpretation of \mathbf{W} much more speculative.

The non-negative constraint and the additive property of the skills bring a specific interpretation of the Q-matrix. For example, if an item requires skills a and b with the same weight each, then each skill will contribute equally to the success of the item. This corresponds to the notion of a *compensatory* or *additive* model of skills.

In our study, we focus on high level skills, which we term *topic skills*. However, if an item requires two specific lower level skills, such as mastery of the rules $a/b + c/b = (a+b)/c$ and $a/b \cdot b = a$, a *conjunctive* model would be necessary, indicating that a failure is expected if any skill is not mastered. The standard interpretation of the Q-matrix corresponds to the conjunctive model, and the \mathbf{W} matrix of NMF does not correspond to this interpretation, unless and as mentioned, we assume that each item belongs to a single skill and for which case the two interpretations are indiscernible.

A last remark on NMF: as mentioned above, the factorization can produce multiple solutions, even with a single algorithm, which raises the issue of stability of the results. However, Schachtner et al. [2010] discuss this issue and suggest that for binary data the problem may not appear at all. Nevertheless, we will assess the extent to which the multiple solution issue impacts the validity and usefulness of the approach by running multiple folds simulations.

3. Q-MATRIX EXTRACTION FROM SIMULATED DATA

Let us start with an assessment of the validity of the NMF technique to extract the Q-matrix from simulated data and ascertain under which assumptions its effectiveness can be shown.

For the sake maintaining the similarity with real data analyzed later in this paper, let us use a 4 skill Q-matrix. Under the assumption that the topic (skill) is the only factor that affects performance and that each item depends on a single topic, the simulated data for 40 items and 100 examinees can be generated from a matrix 40×100 , \mathbf{P} , where each column contains 40 probabilities, one probability per item, structured as a sequence of 10×4 probabilities:

$$(p_{1,1}, p_{1,2}, \dots, p_{1,10}, p_{2,1}, \dots, p_{2,10}, p_{3,1}, \dots, p_{3,10}, p_{4,1}, \dots, p_{4,10})$$

where $p_{1,1}$ to $p_{1,10}$ are all equal, $p_{2,1}$ to $p_{2,10}$ are all equal, and so on. Each column contains therefore only 4 distinct and independent probabilities, one for each skill. These probabilities are generated from a random variable, z , taken from a normal standard distribution and transformed into a probability by computing the *cumulative distribution function* (the area $[-\infty, z]$).

Given the probability matrix \mathbf{P} , a data matrix having the same dimensions as \mathbf{P} is generated, \mathbf{D} , by sampling a success or failure, $\{0, 1\}$ using $P_{i,j}$ as the probability of success and $1 - P_{i,j}$ for failure. The matrix \mathbf{D} corresponds to \mathbf{V} in equation (1). A sample of this data is provided in figure 1(a). By grouping

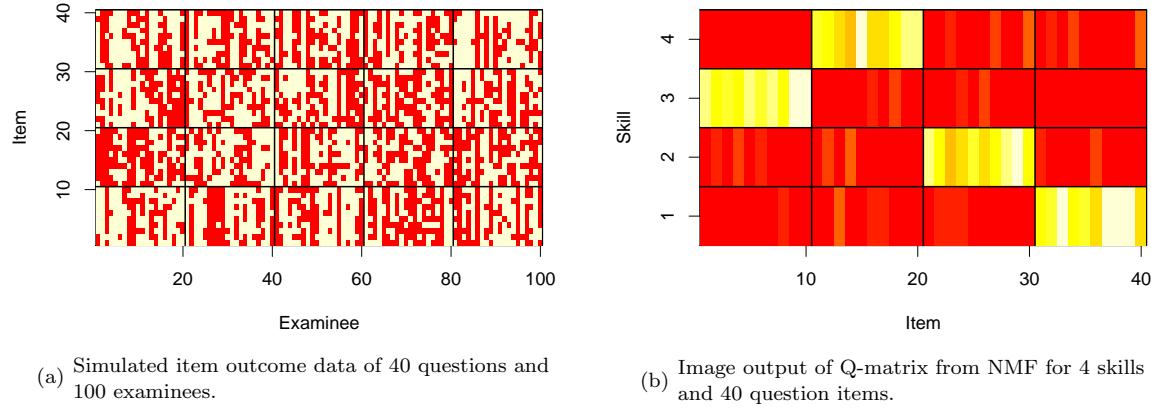


Fig. 1. Simulated data (a) and the corresponding Q-matrix (b) under the assumption that topic is the only factor that affects success. Skill mastery follows a standard normal distribution. A perfect match from items to skills is obtained with this Q-matrix.

items in 4 contiguous groups of 10, the effect of the different levels of skill is apparent: a high probability of mastery will appear as a vertical pattern (single examinee) consisting mostly of pale square dots between vertical stretches of 10 contiguous items, whereas a low probability appears as a pattern composed of mostly dark red dots. No horizontal pattern is apparent since we do not define an item difficulty factor in this data. Similarly, no vertical pattern is apparent *across* the groups of 10 contiguous items because no general ability factor is attributed to examinee (however, vertical patterns are apparent *within* the 10 contiguous item arrangement).

When NMF is applied to \mathbf{D} the resulting \mathbf{W} matrix can be considered as the Q-matrix. For simulated data generated according to the procedure described above, the NMF algorithm is perfectly accurate in assigning the contiguous items in the same group, as can be seen in figure 1(b) where we find 4 bright squares representing the clusters. The figure's image represent the values of the 40×4 \mathbf{W} matrix in NMF (transposed in this image) that directly represents what can be considered as a Q-matrix. Values are mapped to color gradients ranging from pale yellow to dark red.

Items 1 to 10 can readily be assigned to skill 3, items 10 to 20 to skill 4, and so on. The pattern is very obvious to the eye. A simple algorithm, that takes the maximum value for each item in the Q-matrix of figure 1(b) as the main skill, can systematically and correctly classify all question items in the correct skill cluster. These results are, for all practical purposes, deterministic, even though some variance could theoretically occur (we report variance when it becomes substantial later).

The visual results of the Q-matrix leaves little doubt that, under the assumption that topic skill is the only factor that affects performance, the NMF technique is highly effective. We now turn to real data and replicate some experiments by Winters et al. [2005] to verify how the results come out under realistic conditions.

4. Q-MATRIX EXTRACTION FROM REAL DATA

Winters et al. [2005] experimented with NMF over SAT Subject Test data (see CollegeBoard [2011])¹. The data is broken down in 4 topics: (1) Mathematics, (2) Biology, (3) World History, and (4) French. These topics are sufficiently far apart that we can expect that they have strong intra-topic correlation and are therefore discernible for clustering. The data is composed of a total of 297 respondents who completed the 40 question items tests over the Internet. The profile of the respondents is unknown but they are probably from the university student community.

This data has the same structure as the simulated data of section 3: 40 question items broken down into 4 topics of 10 items each. The results of the NMF algorithm over this data is reported in figure 2. Variation in the difficulty of each topic is apparent in figure 2(a), where items 1 to 10 show a higher success rate than items 10 to 20. Individual item difficulty is also apparent by the horizontal patterns, as can be expected. Although we can discern some vertical patterns across item groups, it is far less apparent (except intra-topic vertical patterns), suggesting that examinee ability does not span very much across topics.

Figure 2(b) shows the Q-matrix obtained from the SAT data. It is consistent with the results from Winters et al. [2005]. Clustering of the Mathematics (items 1 to 10) and the French items (31 to 40) is relatively well defined, but not so with the Biology (21 to 30) and World History (31 to 40).

As mentioned, clustering is based on the simple algorithm which assigns each item to one of the 4 clusters based on the maximum column value in matrix \mathbf{W} . Given that we know the actual category of each item, the accuracy of the clustering can be computed. This is obtained by a two step process. First, a contingency table is compiled from the clustering algorithm. Next, the lines are reordered so that the sum of the diagonal is maximized. The ratio of this sum over the total represents the accuracy of the assignment. An example of the contingency table obtained is given below for the SAT data along with its reordering:

		Cluster						Cluster				
		1	2	3	4			1	2	3	4	
Category	1	5	5	0	0	Reordering	4	10	0	0	0	
	2	0	0	10	0			1	5	5	0	0
	3	1	0	1	8			2	0	0	10	0
	4	10	0	0	0			3	1	0	1	8

Note that the category and the cluster labels are irrelevant for measuring accuracy, but it is interesting to note that in this example the values of 10 are the Mathematics and French categories/clusters. As mentioned, the sum of the diagonal over the sum of all values represents the accuracy of this assignment: $33/40 = 0.825$.

Let us now turn to another data set from Winters et al. [2005] for which the task of deriving a Q-matrix from data was shown very challenging. They used questions published from the Trivial Pursuit game and assembled a test that mimics the 4 topic structure of the SAT with 10 questions on each of: (1) Arts and entertainment, (2) Sports and leisure, (3) Science and nature, and (4) Geography. The results of our replication of this experiment are reported in figure 3.

Winters et al. [2005] results over the Trivia data concurs with our experiment and show that the NMF is no better than chance at correctly clustering items and building a Q-matrix. The most troubling findings from their experiments is that the Trivia results are similar to the results they obtain over a number of test outcome from different computer science courses: “Nearly every course behaves the same as the trivia data. Only our smallest data set, the Algorithms course data, showed any significant hint of topic structure.” This

¹The data sets from [Winters et al. 2005] were made available from <http://alumni.cs.ucr.edu/~titus/>. The simulation scripts of this paper are available from <http://www.professeurs.polymtl.ca/michel.desmarais/Papers/EDM2011/scripts.html>. They are based on the NMF package from the statistical software R.

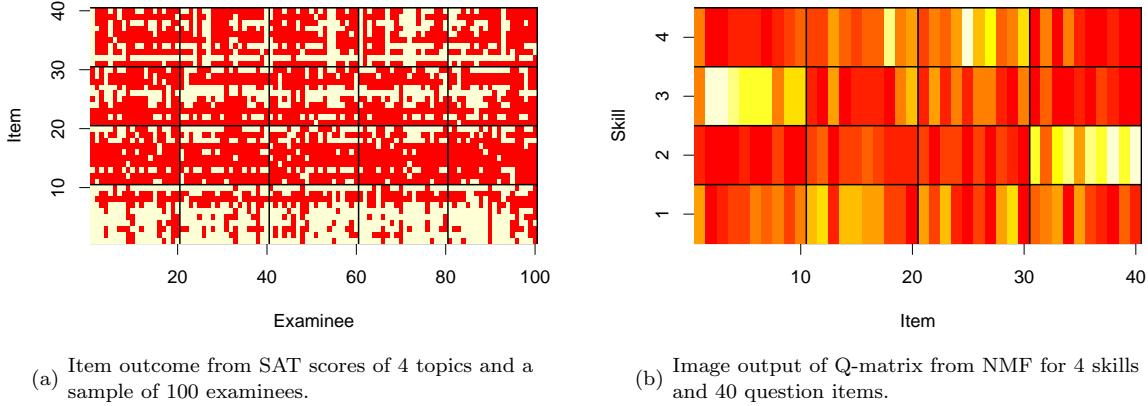


Fig. 2. NMF results over SAT data.

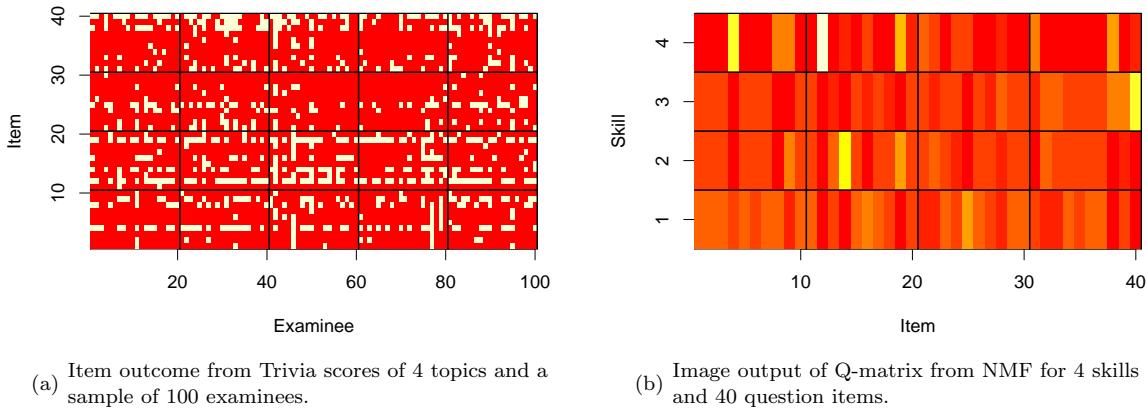


Fig. 3. NMF results over Trivia data.

conclusion casts a gloomy picture for high level transfer models, where we aim to assess the mastery of topic specific skills from similar topic skills.

However, statistical characteristics of the test data may also influence what can be extracted from this data. For example, skewness of the scores towards 0% or 100% will result in sparsity of success/failure that can negatively affect the ability to extract a valid Q-matrix from the data. The Trivia data shows such skewness towards low success rate and we can question whether this is not the source of the low accuracy.

In the next section, we investigate the influence of the success rates and item and examinee variance over the Q-matrix validity.

5. INVESTIGATING THE PARAMETER SPACE OF SIMULATED STUDENT DATA

We turn back to simulated data to assess how the validity of the Q-matrix degrades under relaxed assumptions and under different ratios variance ratios between the skill, item difficulty, and examinee ability factors. This will allow us to better quantify the effect of the skill factor on examinee performance with respect to item difficulty and examinee ability.

First, we verify if NMF can extract the Q-matrix if we drop the unrealistic assumption set in section 3 and assume that item difficulty and person ability each contribute to the probability of success of an item by the same amount that topic skill can influence the probability of success.

Recall that the matrix \mathbf{P} , as defined in section 3, contains independent normally distributed probabilities, each probability representing the chances of success to items of a single topic and for a single examinee. To account for the fact that item difficulty also affects item success, the probability of each item is modulated by a random quantity that is normally distributed with the same mean and variance (0,1) as the topic skill probability. Akin to item difficulty, examinee ability is accounted for by a similar quantity added on an examinee basis. Therefore, the probability of success by an examinee, m , to an individual item, n , belonging to topic, q , is defined as:

$$P(X_{mnq}) = \Phi(\beta_m + \beta_n + \beta_q) \quad (2)$$

where $\Phi(x)$ is the *cumulative distribution function* of the standard normal distribution, and where β_m , β_n and β_q are random Gaussian variables where the mean and standard deviation of β_m and β_n are:

$$\begin{aligned} \beta_m &\sim \mathcal{N}(\bar{X}, s_m) \\ \beta_n &\sim \mathcal{N}(\bar{X}, s_n) \end{aligned}$$

The variable \bar{X} is constrained to be the mean of the whole data (matrix \mathbf{D}). Variables s_m and s_n are respectively the individual examinee and item specific standard deviations. In the case of β_q , the mean can vary across each skill and is therefore defined as:

$$\beta_q \sim \mathcal{N}(\bar{X}_q, s_q)$$

The parameter \bar{X}_q is the specific mean of a skill and the different values must be congruent with \bar{X} (the weighted sum of the mean for each skill times the number of items belonging to that skill must be equal to \bar{X}). s_q is the inter-skill standard deviation, measured by averaging the standard deviations of cluster means on an examinee basis.

Equation (2) can be considered as a simple model of examinee performance as a function of topic skill mastery, item difficulty, and examinee general ability (which spans across topics). In spite of its simplicity compared to other means of generating simulated data (for eg., see [Desmarais and Pelczer 2010]), it remains realistic for our context where we assume that topic skills are relatively independent, or at least this is an assumption we want to investigate and therefore it makes sense that our model follows that same assumption.

Figure 4(b) displays the Q-matrix (\mathbf{W}) obtained from applying NMF over the data generated according to equation (2) with values of 0 for the mean and of 1 for the standard deviation for all β parameters. The raw data is displayed in figure 4(a).

Although we can visually appreciate that the clustering in the Q-matrix is not quite as sharp as in figure 1, these results still yield a perfect match of item to skills using the simple algorithm outlined in section 4.

Figure 4 shows that, when the mean and variance of the different β parameters in equation (2) are all equal (standard normal), the Q-matrix from NMF perfectly matches the underlying Q-matrix. Of course, as the effect of the topic skill parameter, β_q , becomes weaker compared to the other two parameters, the

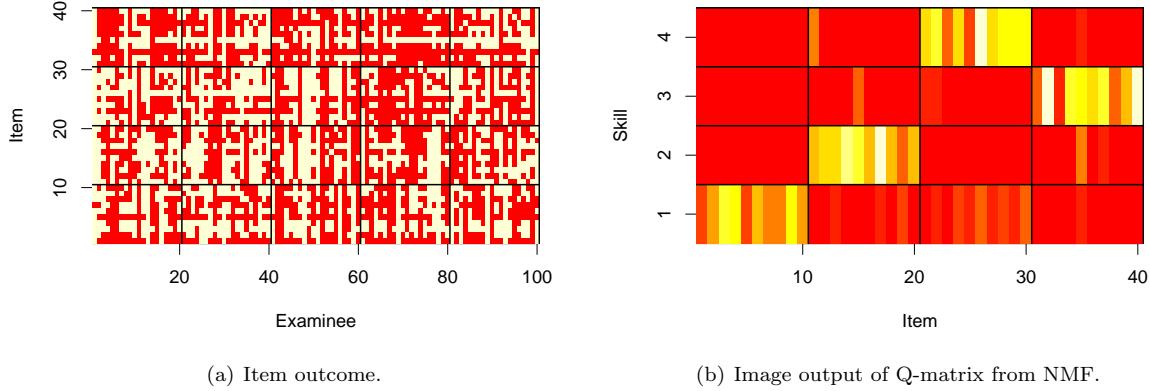


Fig. 4. NMF results over random data from randomly distributed data according to equation (2), reflecting equal effect of topic, item difficulty, and examinee ability over the probability of success.

accuracy of the item-skill match will become lower. This can be observed in table I where the link between accuracy and parameter ratios is quantified.

Table I reports the accuracy results of 14 N-folds simulation experiments conducted with different parameters. For simplicity, we consider a single mean of 0 for β_q . We also restrict the standard deviations to 1 for β_m and β_n given that they have the same effect according to equation (2) and that we are interested in the values of the parameters respective to one another, therefore we can keep them fixed and vary s_q only. Note also that positive and negative values for the means β_n and β_m have symmetric effect such that only positive values are reported.

The first experiment reports an accuracy of 0.36 when no topic skill is defined². As the variance increases (“S.d.”: standard deviation column in the table), the accuracy over a 20-fold simulation gradually reaches 1 as its variance approaches that of the other two parameters. This trend is expected, but it quantifies, in terms of relative variance, the relation between the effect of the topic skill and the item and examinee effect. When the variance of the topic factor is comparable to that of item and examinee factors, the method yields very high accuracy.

Experiments 6 to 9 show the results of variations over the means of β_n and β_m . Experiment 7 shows that when both means of β_n and β_m are increased to 1 (in z score of the standard normal distribution), the accuracy starts to drop slightly to 0.98. Only for means of 1.5 and 2.0 does the performance decrease noticeably to 0.90 and 0.81 respectively.

In experiment 10, the simulation parameters replicate those of the Trivia data set, whereas experiment 12 is done with parameters from the SAT data set. Experiments 11 and 13 report the accuracies of NMF over the real data, corresponding respectively to figures 3 and 2.

For the Trivia data, the accuracy is comparable to the random, no topic skill condition. This results concurs with the conclusion of Winters et al. [2005], namely that topic subject is not a determining factor that affects

²If we had a very large number of items, this number, 0.36, would be close to 0.25, the theoretical accuracy of a random match in a 4×4 contingency table. However, the 40 items distribution in this table create an opportunity of over fit for the algorithm that decides which cluster is assigned to which skill. The difference of 0.11 ($0.36 - 0.25$) can be attributed to this over-fitting.

Table I.
Experiments over the parameter space of skills, items, and examinee
(respectively β_q , β_n , and β_m in equation (2)).

	Parameter space						N folds	Accuracy		
	Topic skill (β_q)		Item (β_n)		Examinee (β_m)			Mean acc.	S.d. acc.	
	Mean	S.d.	Mean	S.d.	Mean	S.d.				
1*	0	0	0	1	0	1	20	0.36	0.05	
2	0	0.10	0	1	0	1	20	0.48	0.07	
3	0	0.25	0	1	0	1	20	0.60	0.11	
4	0	0.50	0	1	0	1	20	0.93	0.08	
5	0	1	0	1	0	1	20	1	0	
6	0	1	0.50	1	0.50	1	20	1.00	0.01	
7	0	1	1	1	1	1	20	0.98	0.07	
8	0	1	1.50	1	1.50	1	20	0.90	0.12	
9	0	1	2	1	2	1	20	0.81	0.16	
<i>Trivia data parameters</i>										
10	0	0.12	-1.05	0.73	-1.05	0.45	20	0.75	0.12	
11**	n.a.	0.12	-1.05	0.73	-1.05	0.45	20	0.35	0.03	
<i>SAT data parameters</i>										
12	0	0.24	-0.33	0.86	-0.33	0.50	20	0.98	0.05	
13***	n.a.	0.24	-0.33	0.86	-0.33	0.50	20	0.72	0.02	
14***	n.a.	0.24	-0.33	0.86	-0.33	0.50	20	0.96	0.05	

* No topic skill effect conditions

** Real data

*** Real data and scoring for the Mathematics and French topics only

test performance. Considering that they obtained similar results for topics from academic computer science courses, these results are disconcerting.

However, we conjectured earlier that the low success rate of the Trivia data could explain the low accuracy results obtained. This is only partly the case. When the simulations parameters are set to the same values as the Trivia data, the accuracy obtained is 0.75 (experiment 10³) whereas the real data results are 0.35 (experiment 11). Therefore, results of experiment 10 suggest that the gap between 0.75 and 0.35 is attributable to the lack of skill effect in this data.

Comparing the results to the accuracy reported on experiments 11 and 13 for real data, we observe that for SAT data, the accuracy is lower than experiment 12 and somewhere between experiments 3 and 4, which corresponds to a standard deviation of topic skill between 0.25 and 0.5 when β_n and β_m have a (0,1) standard distribution. In other words, the skill effect is a little less than half the item and examinee effects.

If we look only at the clustering for Mathematics and French (experiment 14) which are the most separable topics, then the accuracy goes up to 0.96, which is much closer to experiment 12. In terms of relative effect, the skill effect between Mathematics and French is close to the 0.93 accuracy obtained in 4, for which the standard deviation of skill effect is 0.50 of the item and examinee parameters.

In summary, the Trivia data shows negligible effect of topic skill, whereas the SAT data shows an effect that is essentially attributable to the Mathematics and French topics that can be clearly distinguished in the Q-matrix derived with NMF. The topic skill effect can be quantified as somewhere between 1/4 to 1/2 of the

³Experiment 10 has a relative skill-item s.e. of $0.12/0.73 = 0.16$, standing between experiments 2 and 3, and a relative skill-examinee s.d. of $0.12/0.45 = 0.27$, standing close to experiment 3. If the performance followed some additive function of each of these ratios, we would expect the performance to be no better than that of experiment 3, 0.60. Given that it stands higher at 0.75, we have to conclude that the effect of s.d. ratios over the performance is more complex, possibly a ratio of s.d. such as topic/(item \times examinee).

item and examinee effect as measured by the standard deviation, and over 1/2 if we only take Mathematics and French effects alone.

6. DISCUSSION

In undertaking this exploratory work, we were hoping to show that the failure to find an effective Q-matrix from some data sets, such as the Trivia data set, was due to highly skewed tests scores: either the scores are too high or too low, and the raw data becomes too sparse of successes or failures to allow the NMF algorithm to derive a reliable Q-matrix. Results from our experiments suggest that, in fact, this is only partly the case. It still leaves open the suggestion that the topic skill factor has sometimes a negligible effect on performance, or at least a much lower effect than we are generally inclined to believe. From Winters et al.'s [2005] previous results, we can expect this to be the case for many courses that divide their content according to sub-topics.

Our results further indicate that for well delineated topic skills like Mathematics and French, the effect is relatively strong, in a range around half that item difficulty and examinee ability according to the results in table I, at least for highly separable topics like Mathematics and French. In this case, the accuracy of matching items to skills with NMF is well in the range of 90%, which confirms the effectiveness of this technique under these conditions.

This study was conducted under the assumption that we know the number of skills for the clustering and for building the Q-matrix. This is not the case in general. However, the visualization technique used throughout this paper shows that for well delineated topic skills, clustering with NMF is easily perceived through the human eye.

REFERENCES

- BARNES, T. 2006. Evaluation of the q-matrix method in understanding student logic proofs. In *Proceedings of the Nineteenth International Florida Artificial Intelligence Research Society Conference, Melbourne Beach, Florida, USA, May 11-13, 2006*, G. Sutcliffe and R. Goebel, Eds. AAAI Press, 491–496.
- BERRY, M. W., BROWNE, M., LANGVILLE, A. N., PAUCA, V. P., AND PLEMMONS, R. J. 2007. Algorithms and applications for approximate nonnegative matrix factorization. *Computational Statistics & Data Analysis* 52, 1, 155 – 173.
- CEN, H., KOEDINGER, K. R., AND JUNKER, B. 2005. Automating cognitive model improvement by A* search and logistic regression. In *Educational Data Mining: Papers from the 2005 AAAI Workshop.*, J. Beck, Ed. Technical Report WS-05-02. Menlo Park, California: AAAI Press, 47–53.
- CEN, H., KOEDINGER, K. R., AND JUNKER, B. 2006. Learning factors analysis — A general method for cognitive model evaluation and improvement. In *Intelligent Tutoring Systems, 8th International Conference, ITS 2006, Jhongli, Taiwan, June 26-30, 2006, Proceedings*. 164–175.
- COLLEGEBOARD. 2011. Sat subject tests practice questions. <http://sat.collegeboard.com/practice/sat-subject-test-preparation> (consulted on April 2, 2011).
- DESMARAIIS, M. C. AND PELCZER, I. 2010. On the faithfulness of simulated student performance data. In *3rd International Conference on Educational Data Mining EDM2010*, R. S. J. de Baker, A. Merceron, and P. I. Pavlik, Eds. www.educationaldatamining.org, 21–30.
- LEE, D. D. AND SEUNG, H. S. 1999. Learning the parts of objects by non-negative matrix factorization. *Nature* 401, 6755, 788–791.
- SCHACHTNER, R., POPPEL, G., AND LANG, E. 2010. A nonnegative blind source separation model for binary test data. *Circuits and Systems I: Regular Papers, IEEE Transactions on* 57, 7, 1439 –1448.
- TATSUOKA, K. K. 1983. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement* 20, 345–354.
- WINTERS, T., SHELTON, C., PAYNE, T., AND MEI, G. 2005. Topic extraction from item level grades. In *American Association for Artificial Intelligence 2005 Workshop on Educational Datamining*.

Student Translations of Natural Language into Logic: The Grade Grinder Corpus Release 1.0

Dave Barker-Plummer, Richard Cox and Robert Dale

Students find logic hard. In particular, they seem to find it hard to translate natural language sentences into their corresponding representations in logic. As an enabling step towards determining why this is the case, this paper presents the public release of a corpus of over 4.5 million translations of natural language (NL) sentences into first-order logic (FOL), provided by 55,000 students from almost 50 countries over a period of 10 years. The translations, provided by the students as FOL renderings of a collection of 275 NL sentences, were automatically graded by an online assessment tool, the Grade Grinder. More than 604,000 are in error, exemplifying a wide range of misunderstandings and confusions that students struggle with. The corpus thus provides a rich source of data for discovering how students learn logical concepts and for correlating error patterns with linguistic features. We describe the structure and content of the corpus in some detail, and discuss a range of potentially fruitful lines of enquiry. Our hope is that educational data mining of the corpus will lead to improved logic curricula and teaching practice.

1. INTRODUCTION

From a student's perspective, logic is generally considered a difficult subject. And yet it is an extremely valuable and important subject: the ability to reason logically underpins the Science, Technology, Engineering and Mathematics (STEM) fields which are seen as central in advanced societies. We believe it is in society's interests to make logic accessible to more students; but to do this, we need to have an understanding of precisely what it is about logic that is hard, and we need to develop techniques that make it easier for students to grasp the subject.

One key component skill in the understanding of logic is a facility for manipulating formal symbol systems. But such a skill is abstract and of little value if one does not also have the ability to translate everyday descriptions into formal representations, so that the formal skills can be put to use in real-world situations. Unfortunately, translating from natural language into logic is an area where students often face problems.

It seems obvious that the difficulties students face in this translation task will, at least in part, be due to characteristics of the natural language statements themselves. For example, we would expect it to be relatively easy to translate a natural language sentence when the mapping from natural language into logical connectives is transparent, as in the case of the mapping from *and* to ' \wedge ', but harder when the natural language surface form is markedly different from the corresponding logical form, as in the translation of sentences of the form *A provided that B*. However, evidence for this hypothesis is essentially anecdotal, and we have no quantitative evidence of *which* linguistic phenomena are more problematic than others.

It is against this background that we present in this paper the release of a publicly-available anonymised corpus of more than 4.5 million translations of natural language (NL) sentences into first-order logic (FOL) sentences, of which more than 604,000 (approximately 13%) are categorized by an automatic assessment tool as being in error. For each item in the corpus, we know what NL sentence was being translated, and we have both the FOL translation the student provided, and a 'gold-standard' answer representing the class of correct

Author's addresses:

Dave Barker-Plummer, CSLI, Cordura Hall, Stanford University Stanford, CA, 94305, USA; email: dbp@stanford.edu;
Richard Cox, Department of Informatics, University of Edinburgh, Edinburgh, EH8 9AB, UK; email: rcox@inf.ed.ac.uk;
Robert Dale, Center for Language Technology, Department of Computing, Macquarie University, Sydney, NSW, 2109, Australia; email: Robert.Dale@mq.edu.au.

answers.¹ Students are identified by unique anonymised IDs, so the corpus allows us to determine how many previous attempts the student has made at the same exercise and the time intervals between attempts, and also to correlate any given student's performance across exercises. The data thus makes possible a broad range of analyses of student behaviors and performance. We are making the corpus available to the wider community in the hope that this will encourage research that leads to improvements in the teaching of logic.²

Section 2 explains the wider context in which this data has been collected, which has allowed us to gather a very large corpus of data regarding student performance at various tasks in logic learning. Section 3 then describes the focus of this paper—what we call the *Translations Subcorpus*—in more detail. Section 4 describes the format of the data as it appears in the corpus. Section 5 provides summary statistics over the errors in the corpus, and makes some observations about the nature of these errors. Section 6 concludes with some illustrative analyses and suggestions for ways in which this corpus can be exploited.

2. BACKGROUND

The data described here consists of student-generated solutions to exercises in *Language, Proof and Logic* (LPL; [Barwise et al. 1999]), a courseware package consisting of a textbook together with desktop applications which students use to complete exercises.³ The LPL textbook is divided into three parts covering, respectively, Propositional Logic, First-Order Logic and Advanced Topics. The first two parts cover material typical of introductory courses in logic. Students completing these two parts of the textbook will have been exposed to notions of syntax and semantics of first-order logic and a natural deduction-style system for constructing formal proofs. Each of these areas of the course are supported by a number of software applications which provide environments where students can explore the concepts being taught.

The LPL textbook contains 748 exercises, which fall into two categories: 269 exercises which require that students submit their answers on paper to their instructors, and 489 for which students may submit answers to the Grade Grinder, a robust online automated assessment system that has assessed approximately 2.75 million submitted exercises by more than 55,000 individual students in the period 2001–2010. This student population is drawn from approximately a hundred institutions in almost fifty countries. Figure 1 provides statistics on how this data breaks down across the 10 years that the corpus represents.⁴

Student users of the system interact with the Grade Grinder by constructing computer files that contain their answers to particular exercises that appear in the LPL textbook. These exercises are highly varied, and make use of the software applications packaged with the book. Some focus on the building of truth tables using an application called Boole; some involve building blocks world scenarios using a multimodal tool called Tarski's World, in which the student can write FOL sentences and simultaneously build a graphical depiction which can be checked against the sentences; and some require the construction of formal proofs using an application called Fitch. The Grade Grinder provides us with significant collections of data in all these areas. The exercises of interest here are what we call **translation exercises**; they form the basis of the corpus whose release this paper describes, and we discuss them in detail in Section 3 below.

The Grade Grinder corpus is similar to some of the corpora in the PSLC Datasshop repository [Koedinger et al. 2010]. It shares with these the characteristics of being extensive (millions of data points) and longitu-

¹Since the same information can be expressed by many different FOL sentences, any answer that is provably equivalent to this gold-standard answer is considered correct.

²A website is under development; in the interim, the corpus may be obtained by contacting the authors. A longer version of this paper which describes the corpus in more detail is available as a technical report [Barker-Plummer et al. 2011].

³See <http://lpl.stanford.edu>.

⁴The 'Domains' column shows the number of different internet country domains found in the email addresses of the student population for the year in question; definitively correlating these with countries is difficult since a student may use an email address in a domain other than that of their home country, the international use of .com mail hosts being the most obvious instance.

Year	Submissions	Students	Instructors	Domains
2001	190,653	4,097	142	23
2002	237,942	5,219	152	26
2003	238,104	5,106	168	33
2004	251,898	5,473	196	28
2005	255,974	5,295	182	27
2006	266,208	5,295	207	31
2007	304,719	6,444	224	33
2008	322,273	7,174	243	31
2009	331,746	6,489	212	33
2010	352,262	7,404	217	23

Fig. 1. Grade Grinder Usage Statistics: 2001–2010

dinal (repeat submissions by students over a semester or longer). However, it is not as fine-grained as many DataShop datasets.⁵ For example, the DataShop Geometry tutor dataset contains data on students' actions and system responses at the level of knowledge components (skills or concepts). In contrast, a Grade Grinder submission represents the end-point of a student's work on an exercise. The corpus described here also differs from many DataShop corpora in that is not derived from an intelligent tutoring system or cognitive tutor, but from a blended learning *package* consisting of courseware, several desktop computer applications, and an online grading system.

3. NATURAL LANGUAGE TO LOGIC TRANSLATIONS

As noted above, the exercises in LPL cover a range of different types of logic exercises, and so the Grade Grinder's collection of assessments is very large and varied. Over time, we aim to make the various components of this corpus available; as a first step, we are making available what we believe may be the most useful component of the corpus, this being the part that is concerned with students' translations of natural language sentences into logic.

Translation exercises ask the student to translate a number of what we will call **translatable sentences**, writing their answers in a single file, which is then submitted to the Grade Grinder. We will refer to each submission of a translated sentence as a **translation act**. Figure 2 shows an example exercise that calls for the student to translate twenty English sentences into the language of FOL. The student's response to such an exercise is considered correct if it contains a translation act for every translatable sentence in the exercise, and every translation act corresponds to a correct translation. The LPL textbook contains 33 translation exercises, involving a total of 275 distinct translatable NL sentences.

The Grade Grinder examines each submitted file, making a note of errors that are found within the student's answers. The files are saved to the corpus, the errors are noted, and an email message is sent to the submitter summarizing these errors. Currently, the Grade Grinder offers only *flag feedback* [Corbett and Anderson 1989], indicating only whether a submitted solution is correct. The software makes no attempt to diagnose the error that has been made, apart from reporting the difference between a well-formed expression of logic that is incorrect, and an ill-formed expression which is meaningless. Figures 3 and 4 respectively give examples of the feedback for the submission of correct and incorrect solutions to the exercise shown in Figure 2. The feedback report in Figure 4 indicates that the student has submitted an incorrect answer to the second sentence, and an ill-formed expression in answer to the sixth sentence. The solution for sentence eighteen is also reported as ill-formed, since there is no text in this slot of the solution.

Each student may submit solutions to the same exercise as many times as desired. Once a student is satisfied with their work, they may submit the work again, this time requesting that a copy of the system's

⁵However, note the File Timestamps information discussed in Section 4.

* **Exercise 7.12** (Translation) Translate the following English sentences into FOL. Your translations will use all of the propositional connectives.

- (1) If a is a tetrahedron then it is in front of d .
- (2) a is to the left of or right of d only if it's a cube.
- (3) c is between either a and e or a and d .
- (4) c is to the right of a , provided it (i.e., c) is small.
- (5) c is to the right of d only if b is to the right of c and left of e .
- (6) If e is a tetrahedron, then it's to the right of b if and only if it is also in front of b .
- (7) If b is a dodecahedron, then if it isn't in front of d then it isn't in back of d either.
- (8) c is in back of a but in front of e .
- (9) e is in front of d unless it (i.e., e) is a large tetrahedron.
- (10) At least one of a , c , and e is a cube.
- (11) a is a tetrahedron only if it is in front of b .
- (12) b is larger than both a and e .
- (13) a and e are both larger than c , but neither is large.
- (14) d is the same shape as b only if they are the same size.
- (15) a is large if and only if it's a cube.
- (16) b is a cube unless c is a tetrahedron.
- (17) If e isn't a cube, either b or d is large.
- (18) b or d is a cube if either a or c is a tetrahedron.
- (19) a is large just in case d is small.
- (20) a is large just in case e is.

Fig. 2. An example exercise (7.12) from LPL

```
Grade report for Oedipus Maas (oedipa@oyodyne-industries.com)
Submission ID: 11.076.18.28.21.L00-000222
Submission received at: Thu Mar 17 18:28:21 GMT 2011
Submission graded at: Thu Mar 17 18:28:33 GMT 2011
Submission graded by: gradegrinder.stanford.edu
```

```
#### No instructor name was given. The report was sent only to the student.
```

```
The following files were submitted:
Sentences 7.12
```

EXERCISE 7.12

```
Sentences 7.12 (Student file: "Sentences 7.12")
Your sentences are all correct. Hurrah!
```

Fig. 3. Example feedback from the Grade Grinder: A translation exercise without errors

email response be sent to a named instructor. The effect of this pattern of interaction with the Grade Grinder is that the corpus contains a trace of each student's progression from their initial submission to their final answer.

We can categorize the translation exercises along three dimensions as follows, and as summarized in Figure 5.

Logical Language. The LPL textbook introduces the language of first-order logic in stages, starting with atomic formulae in Chapter 1, then the Boolean connectives (\wedge , \vee and \neg) in Chapter 3, followed by conditional connectives (\rightarrow and \leftrightarrow) in Chapter 7. These connectives together define the propositional fragment of first-order logic. Finally, the universal and existential quantifiers (\forall, \exists) are introduced in Chapter 9 to complete the language of first-order logic. Exercises have correspondingly complex languages according to the position in which they appear.

```

Grade report for Tyrone Slothrop (tyrone@yoodyne-industries.com)
Submission ID: 11.076.18.30.56.L00-0002222
Submission received at: Thu Mar 17 18:30:56 GMT 2011
Submission graded at: Thu Mar 17 18:31:02 GMT 2011
Submission graded by: gradegrinder.stanford.edu

```

No instructor name was given. The report was sent only to the student.

The following files were submitted:
Sentences 7.12

EXERCISE 7.12

```

Sentences 7.12 (Student file: "Sentences 7.12")
We found problems in your sentences:
*** Your second answer, "SameCol(a d)->Cube(a)", isn't well formed.
*** Your sixth sentence, "Tet(e)->(RightOf(e, b)->FrontOf(e, b))", is not
equivalent to any of the expected translations.
*** Your fifteenth sentence, "Large(a)->Cube(a)", is not equivalent to any
of the expected translations.
*** Your eighteenth answer, "", isn't well formed.
*** Your nineteenth sentence, "Large(a)->Small(d)", is not equivalent to
any of the expected translations.
*** Your twentieth sentence, "Large(a)->Large(e)", is not equivalent to
any of the expected translations.

```

Fig. 4. Example feedback from the Grade Grinder: A translation exercise with errors

Domain Language. While the majority of the exercises in LPL use the language of the blocks world used in Figure 2, eight translation exercises use one of two other languages. In particular we have a language involving the care and feeding of various pets by their associated people. In this language, it is possible to give a translation for sentences like *Max fed Pris at 2:00*. This language is used in six of the translation exercises. The third language is used in only two exercises and is used to make claims about numbers, such as *There is a number which is both even and prime*.

Supporting and Additional Tasks. Each of the exercises in the pet and number languages require only the translation of sentences from NL into FOL. However, the use of the Tarski's World application provides scope for variety in the blocks language tasks. For example, some exercises call for students to complete their translations while looking at a world in which the English sentences are true; some call for them to verify the plausibility of their answers by examining a range of worlds in which the sentences have different truth values; and yet others call for the students to build a world making all of the English sentences true *de novo*. These alternatives represent a range of exercises in which the *agency* of the student varies. The act of constructing, from scratch, a blocks world that is consistent with a list of sentences (such as Example 7.15) requires more engagement and ‘deeper’ processing than one in which the student checks the truth of a sentence against a pre-fabricated diagram (such as Example 7.1). The effect of this variety in agency is one of many possible analyses that could be carried out using this corpus.

Figure 5 lists the different translation exercises and their characteristics. The ‘Language’ column indicates the target language, which is full FOL unless otherwise noted. In the exercises involving the blocks world language, the different kinds of agency that the students have are indicated. **Looking at world** indicates that students are instructed to look at a world in which the sentences are true as they translate the sentences, while **with world check** means that students are instructed to check their translations in specific worlds after the exercise is completed. **With world construction** indicates that students are required to construct

Exercise	Sentences	Language	Supporting Tasks
1.4	12	blocks (atoms)	
3.20	10	blocks (Boolean)	indirect + looking at world
3.21	12	blocks (Boolean)	with world check in next exercise
7.11	10	blocks (Propositional)	indirect + looking at world
7.12	20	blocks (Propositional)	with world check in next exercise
7.15	12	blocks (Propositional)	with world construction
9.12, 11.4, 14.4	10, 8, 7	blocks	indirect + looking at world
9.16	16	blocks	one existential + with world check
9.17	15	blocks	one universal + with world check
9.18, 11.14, 11.40, 14.28	5, 2, 11, 5	blocks	looking at world, with world check
11.16	10	blocks	skeleton translation given + with world check
11.17, 11.18, 11.19, 14.3	10, 5, 5, 5	blocks	with world check
11.20, 11.39	12, 6	blocks	looking at world
14.6	11	blocks	incomplete information
14.8	2	blocks	
14.27	2	blocks	with world construction
1.9	6	pet (atoms)	
3.23	6	pet (Boolean)	
7.18	5	pet (Propositional)	
9.19, 11.21, 11.41	10, 10, 5	pet	
9.13, 9.25	5, 5	number	

Fig. 5. Exercises involving English sentences (N=33)

(and submit) a world in which their sentences are true. **Incomplete information** means that not all relevant aspects of the world that they are looking at can be seen (e.g., a block may be obscured by a larger one).

The remaining annotations reflect other information given to the student. **Indirect** indicates that translations are given in the form ‘Notice that all the cubes are universal. Translate this’. In the exercises marked with **one existential/universal** students are told that their translations have the specified form, while **skeleton translation given** indicates that students are given a partial translation that they must complete.

4. THE DATA IN THE TRANSLATIONS SUBCORPUS

The Translations Subcorpus represents all of the solutions to translation exercises submitted in the period 2001–2010. Translation exercises have in common that some number of sentences must be translated from NL into FOL. As noted above, we refer to the submission of a single answer to the translation of a sentence as a **translation act**; the corpus records a row of data for each translation act consisting of:

Unique ID. The unique identifier of this translation act (an integer).

Submission ID. The unique identifier of the submission in which this act occurs (an integer).

Subject ID. The unique identifier of the subject performing this act (an integer).

Instructor ID. The unique identifier of the instructor to whom this submission was copied (an integer). This field can be empty if the submission was not copied to an instructor.

Task. An indication of the task to which this is a response (for example, ‘Exercise 1.4, Sentence 7’).

Status. One of the values **correct**, **incorrect**, **ill-formed**, **not-a-sentence**, **undetermined**, **missing** (explained further below).

Answer. The text of the subject’s answer (a string).

Canonical. The canonicalized text of the subject’s answer (a string), where canonicalization simply involves removing whitespace from the answer, so that we can recognize answers which differ only in the use of whitespace.

Field	A Correct Act	An Incorrect Act
ID	7982509	7982763
Submission ID	3808583	4172630
Subject ID	68114	68114
Instructor ID	NULL	NULL
Task	7.12.1	7.12.15
Status	correct	incorrect
answer	Tet(a) → FrontOf(a,d)	Cube(a) → Large(a)
canonical	Tet(a) → FrontOf(a,d)	Cube(a) → Large(a)
Timestamp	2009-05-02 14:01:24	2009-05-02 14:49:32
File Timestamps	C1241297735665D1241298049184	suppressed—see text

Fig. 6. Example data for two translation acts from the corpus

	Correct	Incorrect	Missing	Ill-formed	Non-sentence	Undetermined	Total
First Submission	3,260,979	604,965	481,851	233,605	19,378	45,085	4,645,863
All Submissions	17,254,818	1,805,268	481,851	843,183	58,532	245,055	20,688,707
	83.40%	8.73%	2.33%	4.08%	0.28%	1.18%	

Fig. 7. Total submitted translation acts, classified by status

Timestamp. The time at which the submission was made.

File Timestamps. An indication of timing data concerning the file in which this act appears (explained further below).

Corpus data for two translation acts are shown in Figure 6. Each is an answer to one task within Exercise 7.12 (see Figure 2); the first data column shows a correct answer for Sentence 7.12.1, and the second represents an incorrect answer for Sentence 7.12.15.

The different **Status** values indicate different conditions that can occur when the student's submitted sentence is judged against the gold-standard answer. In addition to **correct** and **incorrect**, a solution may be **ill-formed**, indicating that the solution is not syntactically correct; **not-a-sentence**, indicating a well-formed FOL expression which does not express a claim (the closest analog in NL is a sentence with an unresolved anaphor); or **undetermined**, indicating that the Grade Grinder could not determine whether the submitted answer was correct. Finally, a solution can be **missing**. Because translations are packaged together into submissions of solutions for an exercise which contains multiple translation tasks, we code a solution as **missing** if the subject submitted translations for some, but not all of, the sentences in the exercise. A status of **missing** therefore represents a missed opportunity to submit a solution to accompany others that were submitted.

File Timestamps are an integral part of the Grade Grinder system, and record the times of save and read operations on the submissions file being constructed on the user's desktop. Each time a student opens or saves a file, a timestamp for this operation is added to a collection which is stored in the file. The collection of timestamps serves as a 'fingerprint' for the file, which allows the Grade Grinder to detect the sharing of files between students. Since these timestamps are accurate to the millisecond, it is extremely unlikely that files constructed independently will share any timestamps, and so two students submitting files whose timestamps are the same have likely shared the file. This fingerprinting mechanism is similar to the more familiar checksum algorithms which are often used to fingerprint files; the difference here is that the timestamp fingerprints are not dependent on the content of the file. This is important since some LPL exercises have a unique solution: consequently, arrival at the same content should not be considered evidence of sharing of a file.

Note that this timestamp data can be used to measure the amount of time that subjects spent considering their answers at a more fine-grained level than is indicated by the time between submissions. In the case of the first answer in Figure 6, the timestamp indicates that this file was opened (the segment beginning with C) and then saved (the segment beginning with D) about five minutes later (313,519ms being the precise difference between the two numbers). The timestamp data for the second answer contains fifteen segments, and so has been suppressed here because it is too large to display.

5. SOME SUMMARY DATA

The corpus contains a total of 4,645,563 initial submissions of translation acts by students, with 604,965 (13%) considered to be in error by the Grade Grinder. The breakdown of these initial submissions as provided by the Grade Grinder is shown in the upper half of Figure 7.

In fact, however, these numbers form a lower bound on the number of translation acts in the corpus. As noted earlier, a typical interaction with the Grade Grinder consists in a sequence of submissions, each of which may contain many translation acts. Initially, some of the translations in the submission will be correct and others incorrect. In each subsequent submission, some of the incorrect sentences will be corrected, while the correct sentences will be resubmitted; finally, the student may verify that all sentences are correct, and the student will likely then resubmit the complete set copied to their instructor. We therefore store multiple instances of the same translation acts.

The same phenomenon impacts on incorrect translation acts. If a student has made a mistake in both Sentence n and Sentence $n + 1$, a common behavior is to repeat the submission first with a correction for Sentence n , but leaving the incorrect translation of Sentence $n + 1$ unmodified from the previous submission, only returning to this once a correct answer for Sentence n has been achieved. This results in multiple instances of the same incorrect translation act. However, it is important to observe that in some cases these resubmitted incorrect answers may reflect deliberate acts, and so the real number of intended translation acts in the distributed corpus may in fact be larger than our initial counts suggest. We provide all translation acts in the distributed corpus, with the corresponding counts shown in the lower half of Figure 7. The distributed corpus thus contains a total of 20,688,707 translation acts; this opens the door to additional analyses that would not be possible if only first submissions were available.

Note that we count as errors only those translations that are assessed by the Grade Grinder as definitely incorrect. Expressions which are offered as translations but which are not well-formed expressions of FOL, and those which are well-formed but not sentences, are counted separately. Of course, these expressions are really different kind of errors, and may serve to shed light on student behavior in other ways.

Among the translation exercises, the sentences most commonly mistranslated on the student's first attempt are shown in Figure 8. In this figure, the column headed **N** represents the total number of translation acts concerning this sentence, while the column headed **error/N** is the proportion of these acts that are marked as incorrect. The column headed **Count** applies to the distinct incorrect sentences, and indicates the number of translation acts that result in this answer.

6. POTENTIAL ANALYSES OF THE CORPUS

We conclude by outlining a number of ways in which the Translations Subcorpus can be analysed.

Sentence Features. What features of sentences are particularly difficult for all students (in the aggregate) to translate? We report on work of this type in [Barker-Plummer et al. 2011]. We categorized the sentences according to whether they contained shape, size and spatial predicates, and then examined the error rates for eight resulting types of sentences. Sentences that mix shape and spatial predicates, and size and spatial predicates are each harder to translate than sentences that contain all three kinds of predicates.

Task	Answer	N	Error/N	Count
11.39.4	Every small cube is in back of a particular large cube	3520	69.0%	
11.39.4	Correct $\exists x (\text{Large}(x) \wedge \text{Cube}(x) \wedge \forall y ((\text{Small}(y) \wedge \text{Cube}(y)) \rightarrow \text{BackOf}(y, x)))$			
11.39.4	Incorrect $\forall x ((\text{Cube}(x) \wedge \text{Small}(x)) \rightarrow \exists y (\text{Cube}(y) \wedge \text{Large}(y) \wedge \text{BackOf}(x, y)))$			818
11.39.4	Incorrect $\forall x ((\text{Small}(x) \wedge \text{Cube}(x)) \rightarrow \exists y (\text{Large}(y) \wedge \text{Cube}(y) \wedge \text{BackOf}(x, y)))$			420
11.39.4	Incorrect $\forall x ((\text{Cube}(x) \wedge \text{Small}(x)) \rightarrow \exists y (\text{Large}(y) \wedge \text{Cube}(y) \wedge \text{BackOf}(x, y)))$			281
11.39.4	Incorrect $\forall x \exists y ((\text{Cube}(x) \wedge \text{Small}(x)) \rightarrow (\text{Cube}(y) \wedge \text{Large}(y) \wedge \text{BackOf}(x, y)))$			207
11.39.4	Incorrect $\forall x ((\text{Small}(x) \wedge \text{Cube}(x)) \rightarrow \exists y (\text{Cube}(y) \wedge \text{Large}(y) \wedge \text{BackOf}(x, y)))$			164
11.20.1	Nothing to the left of a is larger than anything to the left of b	9101	54.9%	
11.20.1	Correct $\neg \exists x (\text{LeftOf}(x, a) \wedge \forall y (\text{LeftOf}(y, b) \rightarrow \text{Larger}(x, y)))$			
11.20.1	Incorrect $\forall x \forall y ((\text{LeftOf}(x, a) \wedge \text{LeftOf}(y, b)) \rightarrow \neg \text{Larger}(x, y))$			941
11.20.1	Incorrect $\forall x (\text{LeftOf}(x, a) \rightarrow \forall y (\text{LeftOf}(y, b) \rightarrow \neg \text{Larger}(x, y)))$			913
11.20.1	Incorrect $\neg \exists x (\text{LeftOf}(x, a) \wedge \forall y (\text{LeftOf}(y, b) \wedge \text{Larger}(x, y)))$			582
11.20.1	Incorrect $\forall x \forall y ((\text{LeftOf}(x, a) \wedge \text{LeftOf}(y, b)) \rightarrow \neg \text{Larger}(x, y))$			406
11.20.1	Incorrect $\forall x (\text{LeftOf}(x, b) \rightarrow \neg \exists y (\text{LeftOf}(y, a) \wedge \text{Larger}(y, x)))$			307
3.21.5	Neither e nor a is to the right of c and to the left of b	34608	54.4%	
3.21.5	Correct $\neg(\text{RightOf}(e, c) \wedge \text{LeftOf}(e, b)) \wedge \neg(\text{RightOf}(a, c) \wedge \text{LeftOf}(a, b))$			
3.21.5	Incorrect $\neg(\text{RightOf}(e, c) \wedge \text{RightOf}(a, c)) \wedge \neg(\text{LeftOf}(e, b) \wedge \text{LeftOf}(a, b))$			4681
3.21.5	Incorrect $\neg \text{RightOf}(e, c) \wedge \neg \text{RightOf}(a, c) \wedge \neg \text{LeftOf}(e, b) \wedge \neg \text{LeftOf}(a, b)$			1777
3.21.5	Incorrect $\neg(\text{RightOf}(e, c) \wedge \text{LeftOf}(e, b)) \vee \neg(\text{RightOf}(a, c) \wedge \text{LeftOf}(a, b))$			1678
3.21.5	Incorrect $\neg(\text{RightOf}(e, c) \vee \text{RightOf}(a, c)) \wedge \neg(\text{LeftOf}(e, b) \vee \text{LeftOf}(a, b))$			1569
3.21.5	Incorrect $\neg(\text{RightOf}(e, c) \wedge \text{RightOf}(a, c) \wedge \text{LeftOf}(e, b) \wedge \text{LeftOf}(a, b))$			1345
3.23.5	2:00pm is between 1:55pm and 2:05pm	14747	50.4%	
3.23.5	Correct $1 : 55 < 2 : 00 \wedge 2 : 00 < 2 : 05$			
3.23.5	Incorrect $\text{Between}(2 : 00, 1 : 55, 2 : 05)$			14546
3.23.5	Incorrect $\text{Between}(1 : 55, 2 : 00, 2 : 05)$			319
3.23.5	Incorrect $\text{Between}(2 : 00, 2 : 05, 1 : 55)$			178
3.23.5	Incorrect $\text{Between}(2, 1 : 55, 2 : 05)$			133
3.23.5	Incorrect $2 : 00 < 2 : 05$			91
11.40.3	There is a dodecahedron unless there are at least two large objects	3887	48.7%	
11.40.3	Correct $\neg \exists x \exists y (x \neq y \wedge \text{Large}(x) \wedge \text{Large}(y)) \rightarrow \exists z \text{Dodec}(z)$			
11.40.3	Incorrect $\exists x \exists y (\text{Large}(x) \wedge \text{Large}(y) \wedge x \neq y) \rightarrow \neg \exists z \text{Dodec}(z)$			84
11.40.3	Incorrect $\exists x \exists y ((\text{Large}(x) \wedge \text{Large}(y) \wedge x \neq y) \rightarrow \neg \exists z \text{Dodec}(z))$			67
11.40.3	Incorrect $\exists x \exists y \exists z (\text{Dodec}(x) \rightarrow \neg(\text{Large}(y) \wedge \text{Large}(z) \wedge y \neq z))$			54
11.40.3	Incorrect $\exists x \exists y ((\text{Large}(x) \wedge \text{Large}(y) \wedge x \neq y) \rightarrow \exists z (\text{Dodec}(z) \wedge z \neq x \wedge z \neq y))$			48
11.40.3	Incorrect $\forall x \forall y ((\text{Large}(x) \wedge \text{Large}(y) \wedge x \neq y) \rightarrow \neg \exists z \text{Dodec}(z))$			46

Fig. 8. The top five erroneous answers to the each of the five most error-prone tasks

Error Typology. Can the errors that students make in their translations be categorized according to type? In [Barker-Plummer et al. 2008] we examined the most frequent errors in the solution of Exercise 7.12, and discovered that the failure to distinguish between the conditional and biconditional was a significant source of error. Another significant source of error appears to be an expectation that names will appear in contiguous alphabetical order in a sentence (we call these ‘gappy’ sentences); so, a sentence like ‘**a** is between **b** and **d**’ is frequently mistranslated with **c** in place of **d**.

Response to Errors. How do subjects go about finding solutions when their initial attempt is incorrect? We can ask whether the difficulty of repair correlates with the subject, the sentence or with the particular error that was initially made. We have carried out preliminary work [Barker-Plummer et al. 2009] investigating the differences between, on the one hand, translation tasks which are difficult to get right initially but which

are easy to recover from, and on the other hand, those which are perhaps less error-prone, but hard to repair. We think both aspects of the task contribute to the ‘difficulty’ of a task.

Exercise-Level Strategies. There is potential in the corpus for examining strategies that the students adopt when they make multiple errors. Some students appear to attempt to fix all of their incorrect sentences, and others proceed one at a time. These strategies might correlate with success. We can detect differences between these strategies by looking at the sequence of submissions that occurs after the initial submission. In some cases only one sentence will be modified in each subsequent submission; in others many may be altered.

Modality Heterogeneity of Task. Exercises differ in the extent to which they are linguistically and graphically heterogeneous. Some require translation from NL sentences to FOL, whereas others require translation followed by blocks world diagram building. In [Cox et al. 2008], we compared students’ constructed diagrammatic representations of information expressed in NL sentences to their FOL translations, and determined that the error patterns differed in their graphical versus their FOL translations.

Agency in the Task. As discussed in Section 3, translation tasks vary in the degree of agency they require on the part of the student. Using the corpus it would be possible to analyze variability in student performance with agency, to see if these adjunct tasks have an effect on translation accuracy.

Time Course. The timestamp information in the corpus makes it possible to ask how much time students spend (re)considering their answers: does the bulk of time go to particular tasks, or is it evenly distributed?

7. CONCLUSION

With the first release of this corpus, we invite colleagues to exploit its potential for educational data mining. Our hope is that further analyses will provide additional insights into student cognition in the difficult domain of logic, and that findings will inform improved educational practice in logic teaching. In our own work, we aim to (1) enrich the feedback that Grade Grinder provides to students, (2) investigate task agency effects upon learning outcomes, and (3) identify evidence-based improvements to the logic curriculum.

REFERENCES

- BARKER-PLUMMER, D., COX, R., AND DALE, R. 2009. Dimensions of difficulty in translating natural language into first order logic. In *Second International Conference on Educational Data Mining*. Cordoba Spain.
- BARKER-PLUMMER, D., COX, R., AND DALE, R. 2011. Student translations of natural language into logic: The Grade Grinder corpus release 1.0. Technical Report OP-TR-01. Available from openproof.stanford.edu.
- BARKER-PLUMMER, D., COX, R., DALE, R., AND ETCHEMENDY, J. 2008. An empirical study of errors in translating natural language into logic. In *Proceedings of the 30th Annual Cognitive Science Society Conference*, V. Sloutsky, B. Love, and K. McRae, Eds. Lawrence Erlbaum Associates.
- BARKER-PLUMMER, D., DALE, R., AND COX, R. 2011. Impediment effects of visual and spatial content upon language-to-logic translation accuracy. In *Proceedings of the 32nd Annual Cognitive Science Society Conference*, C. Hoelscher, T. F. Shipley, and L. Carlson, Eds. Lawrence Erlbaum Associates.
- BARWISE, J., ETCHEMENDY, J., ALLWEIN, G., BARKER-PLUMMER, D., AND LIU, A. 1999. *Language, Proof and Logic*. CSLI Publications and University of Chicago Press.
- CORBETT, A. T. AND ANDERSON, J. R. 1989. Feedback timing and student control in the lisp intelligent tutoring system. In *Proceedings of the Fourth International Conference on Artificial Intelligence and Education*, D. Bierman, J. Brueker, and J. Sandberg, Eds. IOS Press Amsterdam Netherlands.
- COX, R., DALE, R., ETCHEMENDY, J., AND BARKER-PLUMMER, D. 2008. Graphical revelations: Comparing students’ translation errors in graphics and logic. In *Proceedings of the Fifth International Conference on the Theory and Application of Diagrams*, G. Stapleton, J. Howse, and J. Lee, Eds. Lecture Notes in Computer Science LNAI 5223, Berlin: Springer Verlag.
- KOEDINGER, K., BAKER, R., CUNNINGHAM, K., SKOGSHOLM, A., LEBER, B., AND STAMPER, J. 2010. A data repository for the EDM community: The PSLC datashop. In *Handbook of Educational Data Mining*, C. Romero, S. Ventura, M. Pechenizky, and R. Baker, Eds. CRC Press Boca Raton Florida.

Instructional Factors Analysis: A Cognitive Model For Multiple Instructional Interventions

Min Chi, Stanford University, CA USA
Kenneth Koedinger, Carnegie Mellon University, PA USA
Geoff Gordon, Carnegie Mellon University, PA USA
Pamela Jordan, University of Pittsburgh, PA USA
Kurt VanLehn, Arizona State University, AZ USA

In this paper, we proposed a new cognitive modeling approach: Instructional Factors Analysis Model (IFM). It belongs to a class of Knowledge-Component-based cognitive models. More specifically, IFM is targeted for modeling student's performance when multiple types of instructional interventions are involved and some of them may not generate a direct observation of students' performance. We compared IFM to two other pre-existing cognitive models: Additive Factor Models (AFMs) and Performance Factor Models (PFMs). The three methods differ mainly on how a student's previous experience on a Knowledge Component is counted into multiple categories. Among the three models, instructional interventions without immediate direct observations can be easily incorporate into the AFM and IFM models. Therefore, they are further compared on two important tasks—unseen student prediction and unseen step prediction—and to determine whether the extra flexibility afforded by additional parameters leads to better models, or just to over fitting. Our results suggested that, for datasets involving multiple types of learning interventions, dividing student learning opportunities into multiple categories is beneficial in that IFM out-performed both AFM and PFM models on various tasks. However, the relative performance of the IFM models depends on the specific prediction task; so, experimenters facing a novel task should engage in some measure of model selection.

1. INTRODUCTION

For many existing Intelligent Tutoring Systems (ITSs), the system-student interactions can be viewed as a sequence of steps [VanLehn 2006]. Most ITSs are student-driven. That is, at each time point the system elicits the next step from students, sometimes with a prompt, but often without any prompting (e.g., in a free form equation entry window where each equation is a step). When a student enters an attempt on a step, the ITS records whether it is a success or failure *without the tutor's assistance* and may give feedbacks and/or hints based on the entry. Students' first attempt records on each step are then collected for student modeling. Often times in ITSs, completion of a single step requires students to apply multiple Knowledge Components. A *Knowledge Component (KC)* is: “a generalization of everyday terms like concept, principle, fact, or skill, and cognitive science terms like schema, production rule, misconception, or facet” [VanLehn et al. 2007]. They are the atomic units of knowledge. Generally speaking, students' modeling on conjunctive-KC steps are more difficult than that on steps that require a single KC.

The three most common student modeling methods are: **Knowledge Tracing (KT)** [Corbett and Anderson 1995], **Additive Factor Models (AFM)** [Cen et al. 2006; 2008], and **Performance Factor Models (PFM)** [Pavlik et al. 2009]. When performing student modeling we seek to construct a cognitive model based upon these observed behaviors and to apply the model to make predictions. Generally speaking, we are interested in three types of predictions: type 1 is about how *unseen students* will perform on the observed steps same as those in the observed dataset; type 2 is about how the same students seen in the observed data will perform on *unseen steps*; and type 3 is about how unseen students will perform on unseen steps, that is, *both*. For the present purposes we classify students or steps that appear in the observed training data

as seen and those that appear only in the unobserved test data as unseen. In this paper we will examine prediction types 1 and 2 and leave type 3 for future work.

Previously KT and PFM have been directly compared both on datasets involved single-KC steps [Pavlik et al. 2009] and those involved conjunctive-KC steps[Gong et al. 2010]. Results have shown that PFM is as good or better than KT for prediction tasks under Bayesian Information Criteria (BIC) [Schwarz 1978] in [Pavlik et al. 2009] or using Mean Squared Error (MSE) as criteria in [Gong et al. 2010]. For both BIC and MSE, the lower the value, the better.

While PFM and KT have been compared on datasets involved conjunctive-KC step, prior applications of AFM and PFM have mainly been with single-KC steps and indicated no clear winner. More specifically, while AFM is marginally superior to PFM in that the former has lower BIC and cross-validation Mean Absolute Deviance (MAD) scores in [Pavlik et al. 2009], PFM performed better than AFM under MAD scores in [Pavlik et al. 2011]. For MAD, same as MSE, the lower the value, the better. On the other hand, previous research have shown that AFM can, at least in some cases, do a fine job in modeling conjunctive KCs [Cen et al. 2008]. Therefore, in this paper we will compare AFM and PFM directly on a dataset involving many conjunctive-KC steps.

Moreover, most prior research on cognitive modelings was conducted on datasets collected from classical student-driven ITSs. Some ITSs, however, are not always student-driven in that they may involve other instructional interventions that do not generate direct observations on student's performance. The dataset used in this paper, for example, was collected from a tutor that, at each step chose to elicit the next step information from students or to tell them the next step. In our view these tell steps should also be counted as a type of Learning Opportunity (LO) as they do provide some guidance to students. Yet on the other hand, these steps do not allow us to directly observe students' performance. KT model is designed mainly for student-driven ITSs in that its parameters are directly learned from the sequences of student's performance (right or wrong) on each step. When there are multiple instructional interventions and some of them do not generate direct observations, it is not very clear how to incorporate these interventions directly into conventional KT models. Therefore, in this paper we are mainly interested in comparing AFM and PFM.

Our dataset was collected from an ITS that can either *elicit* the next step from the student or *tell* them directly. Incorporating tell steps into AFM model is relatively easy in that tells can be directly added to total LO counts. The PFM, however, uses student's prior performance counts, the success or failure, in the equation. Since tells do not generate any observed performance, it is hard to include them in the PFM. Therefore, we elected to add a new feature to represent instructional interventions such as tells. As shown later, the new model can be easily modified for modeling datasets with multiple instructional interventions and thus it is named as Instructional Factors Analysis Model (IFM).

To summarize, in this paper we will compare three models, AFM, PFM and IFM, on a dataset involving many conjunctive-KC steps and multiple instructional interventions. Previous research has typically focused on how well the models fit the observed data. In the following, we also investigated how well they perform at making the predictions of unseen students' performance on seen steps (type 1) and seen students' performance on unseen steps (type 2). Before describing our general methods in details we will first describe the three models.

2. THREE MODELS: AFM, PFM, AND IFM

All three models, AFM, PFM, and IFM, use a *Q-matrix* to represent the relationship between individual steps and KCs. Q-matrices are typically encoded as a binary 2-dimensional matrix with rows representing KCs and columns representing Steps. If a given cell $Q_{kj} = 1$, then step j is an application of KC k . Previous researchers have focused on the task of generating or tuning Q-matrices based upon a dataset [Barnes 2005; Tatsuoka 1983]. For the present work we employed a static Q-matrix for all our experiments. Equations 1,

2, and 3 present the core of each model. Below the equations are the detailed descriptions of each term used in the three equations.

The central idea of AFM was originally proposed by [Draney et al. 1995] and introduced into ITS field by [Cen et al. 2006; 2008]. Equation 1 shows that AFM defines the log-odds of a student i completing a step j correctly to be a linear function of several covariates. Here p_{ij} is a student i 's probability of completing a step j correctly, N_{ik} is the prior LO counts. AFM models contain three types of parameters: student parameters θ_i , KC (or skill) parameters β_k , and learning rates γ_k . While AFM is sensitive to the frequency of prior practice, it assumes that all students accumulate knowledge in the same manner and ignores the correctness of their individual responses.

PFM, by contrast, was proposed by [Pavlik et al. 2009] by taking the correctness of individual responses into account. It can be seen as a combination of learning decomposition [Beck and Mostow 2008] and AFM. Equation 2 expresses a student i 's log-odds of completing a step j correctly based upon performance features such as S_{ik} (the number of times student i has previously practiced successfully relevant KC k) and F_{ik} (the number of times student i has previously practiced unsuccessfully relevant KC k). PFM may also include student parameters such as θ_i and skill parameters, such as β_k . Additionally, PFM employs parameters to represent the benefit of students' prior successful applications of the skill μ_k and the benefit of prior previous failures ρ_k .

While PFM was originally proposed without a θ_i , it is possible to include or exclude these student parameters from either PFM or AFM. In prior work, Corbett et al. noted that models which tracked learning variability on a per-subject basis, such as with θ outperform models that do not [Corbett and Anderson 1995]. Pavlik [Pavlik et al. 2009] further noted that the full AFM model seemed to outperform PFM without θ which in turn outperformed AFM without θ . Pavlik et al. also hypothesized that PFM with θ would outperform the other models and they investigated it in their recent work. In this study, our analysis showed that prediction is better with student parameters, especially for AFM models, thus we include θ_i in our versions of both AFM and PFM.

From PFM, IFM can be seen as adding a new feature to represent the tells together with the success or failure counts, shown in Equation 3. Equation 3 expresses a student i 's log-odds of completing a step j correctly based upon performance features including S_{ik} , F_{ik} , T_{ik} (the number of times student i has previously got told on relevant KC k). IFM also includes student parameters θ_i , skill parameters β_k , μ_k , ρ_k , and the benefit of prior previous tells ν_k .

$$\text{AFM: } \ln \frac{p_{ij}}{1 - p_{ij}} = \theta_i + \sum_k \beta_k Q_{kj} + \sum_k Q_{kj} (\gamma_k N_{ik}) \quad (1)$$

$$\text{PFM: } \ln \frac{p_{ij}}{1 - p_{ij}} = \theta_i + \sum_k \beta_k Q_{kj} + \sum_k Q_{kj} (\mu_k S_{ik} + \rho_k F_{ik}) \quad (2)$$

$$\text{IFM: } \ln \frac{p_{ij}}{1 - p_{ij}} = \theta_i + \sum_k \beta_k Q_{kj} + \sum_k Q_{kj} (\mu_k S_{ik} + \rho_k F_{ik} + \nu_k T_{ik}) \quad (3)$$

Where:

- i. represents a student i .
- j. represents a step j .
- k. represents a skill or KC k .
- p_{ij} . is the probability that student i would be correct on step j .
- θ_i . is the coefficient for proficiency of student i .
- β_j . is coefficient for difficulty of the skill or KC k .

- Q_{kj} . is the Q-matrix cell for step j using skill k .
- γ_k . is the coefficient for the learning rate of skill k (AFM only);
- N_{ik} . is the number of practice opportunities student i has had on the skill k (AFM only);
- μ_k . is the coefficient for the benefit of previous successes on skill k (PFM & IFM);
- S_{ik} . is the number of prior successes student i has had on the skill k (PFM & IFM);
- ρ_k . is the coefficient for the benefit of previous failures on skill k (PFM & IFM);
- F_{ik} . is the number of prior failures student i has had on the skill k (PFM & IFM);
- ν_k . the coefficient for the benefit of previous tells on skill k (IFM only);
- T_{ik} . the number of prior Tells student i has had on the skill k (IFM only);

3. TRAINING DATASET AND EIGHT LEARNING OPPORTUNITY MODES

The original dataset was collected by training 64 students on a natural-language physics tutoring system named Cordillera [VanLehn et al. 2007; Jordan et al. 2007] over a period of four months in 2007. The physics domain contains **eight** primary KCs including the weight law (KC_1), Definition of Kinetic Energy (KC_{20}), Gravitational Potential Energy (KC_{21}), and so on. All participants began with a standard pretest followed by training 7 physics problems on Cordillera and then a post-test. The pre- and post-tests are identical in that they both have the same 33 test items. The tests were given online and consisted of both multiple-choice and open-ended questions. Open-ended questions required the students to derive an answer by applying one or multiple KCs.

In this study, our training dataset comprises 19301 data points resulted from 64 students solving 7 training problems on Cordillera. Each student completed around 300 training problem steps. Note that the training dataset does not include the pre- or posttest. In other words, a data point in our training dataset is either the first attempt by a student on an elicit step or a system tell during his/her training on Cordillera only.

There are two types of steps in Cordillera. The *primary steps* are necessary problem-solving and conceptual discussion steps. The *justification steps*, on the other hand, are optional steps that occur when students are asked to justify the primary step they have just completed. The primary steps are designed to move the solution process forward while the justification steps are designed to help the students engage with the domain knowledge in a deeper way. When collecting our dataset the Cordillera system decided whether to elicit or tell each step randomly. Thus, we have two types of LOs: *elicit* and *tell* for the primary steps; and *self-explain* or *explain* for the justifications.

Figure 1 shows a pair of sample dialogues taken from the cordillera system for the same series of primary steps with the same domain content. In dialogue (1.a) the system elects to elicit the students' answer (steps 2- 3), while in dialogue (1.b) the system chooses to tell the student the answer (steps 2). Similarly in Figure 2 we present a similar comparison between a pair of self-explain and explain dialogues. As before both dialogues cover the same domain content. In dialogue (2.a) the system asks the student to self-explain their answer to the question in qualitative terms (steps 3-4). In dialogue (2.b) they are provided with a short qualitative explanation (step 3).

For the primary steps, the average number of decisions ranges from 2.5 for KC_1 to 31 for KC_{20} for elicit steps and ranges from 4 for KC_1 to 47 for KC_{20} for the tell steps. Compared with primary steps, justification steps are significantly less frequent in that the average number ranges from 0 for KC_1 to 3.5 for KC_{21} for *self-explain* steps and ranges from 0 for KC_1 to 2.5 for KC_{20} for explain steps.

In most of the prior research that we have surveyed the authors mainly used datasets involving quantitative problem-solving steps similar to the primary steps in Cordillera. So when counting LOs in this study, we first need to decide whether or not to include justification steps. Additionally, most previous research did not include students' performance on the pretest when counting LOs. While this makes some sense as students

(1.a) Elicit Version
(1) T: So let's start with determining the value of v1.
(2) T: Which principle will help you calculate the rock's instantaneous magnitude of velocity at T1? {ELICIT}
(3) S: definition of kinetic energy
(4) T: Let me just write the equation for you: KE1 = 0.5*m*v1^2.
(1.b) Tell Version
(1) T: So let's start with determining the value of v1.
(2) T: To calculate the rock's instantaneous magnitude of velocity at T1, we will apply the definition of kinetic energy. {TELL}
(3) T: Let me just write the equation for you: KE1 = 0.5*m*v1^2.

Fig. 1. Learning Opportunities on Problem-Solving Steps: Elicit vs. Tell

(2.a) Self-Explain
(1) T: Can we infer the direction of the velocity of the rock at T1 from the rock's kinetic energy at T1? {ELICIT}
(2) S: nope.
(3) T: Excellent! Please explain why. {Self-Explain}
(4) S: Only the magnitude of the velocity and not the direction of it is part of the definition of kinetic energy.
(5) T: Excellent! Now that we know v1, ...
(2.b) Explain
(1) T: Can we infer the direction of the velocity of the rock at T1 from the rock's kinetic energy at T1? {ELICIT}
(2) S: nope.
(3) T: Excellent! This is because the kinetic energy only depends on mass and the magnitude of velocity, not the direction of velocity. {Explain}
(4) T: Now that we know v1, ...

Fig. 2. SelfExplain vs. Explain

receive no feedback indicating their successes or failures during the test, it is still the case that they do practice their skills. Secondly we need to decide whether or not to include student's pretest performance in the LO counts.

In order to explore how different choices of LOs would impact different cognitive models, we defined four ways to count the LOs. In the *primary mode* we count only the primary steps within the ITS. In *pretest-primary* we count the primary mode steps plus the pretest (each test item is treated as one step for training). *Primary-Justify* mode counts the primary and justification steps within the ITS alone. And finally the *overall* mode counts all steps in both the pretest and ITS training.

Note that using different modes of LOs neither changes the size of the training dataset which is generated along students' logs when training on Cordillera nor changes the number of parameters to be fit. Using pretest in the LO count means that various LOs do not start with 0 for the *pretest-primary* and *overall*

modes but are based on the frequency of KC appearances (and, in the case of PFM, the accuracy) in the pretest. For example, if a KC_{20} is tested 20 times in the pretest and a student was correct 5 times and wrong 15 times, then the student's LOs on KC_{20} for pretest-primary and *overall* mode would start with $LO = 20$, $Success = 5$, $Fail = 15$, $Tell = 0$. For *Primary* and *Primary-Justify* modes, all LOs start with 0.

Coupled with this variation we can also count LOs additively or logarithmically. Using logarithmic count is inspired by the power law relationship between measures of performance (reaction time or error rate) and the amount of practice [Newell and Rosenbloom 1981]. But others [Heathcote et al. 2000] have argued that the relationship is an exponential, which corresponds to additive counting. To summarize, we have $\{\text{Primary}, \text{Pretest-Primary}, \text{Primary-Justify}, \text{Overall}\} \times \{\text{count}, \ln(\text{count})\}$, a total of eight LO modes.

4. RESULTS

Two measures of quality, the Bayesian Information Criteria (BIC) and the cross-validation Root Mean Squared Error (RMSE), are used to evaluate how well various instantiated models perform. For both BIC and cross-validation RMSE, the lower the value, the better. BIC [Schwarz 1978] is a criterion for model selection among a class of parametric models with different numbers of parameters. In prior research on the evaluation and comparison of different cognitive models [Cen et al. 2006; Pavlik et al. 2009; Gong et al. 2010] the authors used BIC as a measure of success. In machine learning, however, it is conventional to use the cross-validation RMSE, which is a more interpretable metric and, we believe, a more robust measure. For the purposes of this paper, we will report both BIC and RMSE.

4.1 AFM, PFM, vs. IFM.

First, we will investigate whether considering Tell and Explains into the LOs is beneficial. In traditional cognitive modeling the focus is solely on steps where the student's performance is observed. In the context of Cordillera that means counting only the elicit and self-explain steps as both require students to apply their knowledge without support and their performance can be directly evaluated. For AFM models, we thus compared the AFM algorithms shown in equation 1 by either including Tells and Explains into N_{ik} or by excluding them out of N_{ik} . The two resulted models are referred as AFM-Tell and AFM+Tell respectively. Therefore, in this section we compared four models: AFM-Tell, AFM+Tell, PFM and IFM across eight LO modes.

For each of the four models, its corresponding **count** LOs on corresponding $\{\text{Primary}, \text{Pretest-Primary}, \text{Primary-Justify}, \text{Overall}\}$ modes are defined in Table I. For example, the IFM has three LO counts: prior success S_{ik} , prior failures F_{ik} , and prior tells T_{ik} . Under the Primary-Justify mode (shown in the left bottom of the table), $S_{ik} = \text{Success in (Elicit + Self-Explain)} \text{ on the KC } k$, $F_{ik} = \text{prior failure in (Elicit + Self-Explain)} \text{ on the KC } k$, and $T_{ik} = \text{prior tells and explains on the KC } k$. Once the count mode is defined, the corresponding $\ln(\text{Count})$ mode is simply taking each count logarithmically. For example, under $\{\text{Primary-Justify}, \ln(\text{Count})\}$ mode, we have $S_{ik} = \ln[\text{Success in (Elicit + Self-Explain)} \text{ on KC } k]$, $F_{ik} = \ln[\text{prior failure in (Elicit + Self-Explain)} \text{ on KC } k]$, and $T_{ik} = \ln[\text{prior tells and explains on the KC } k]$.

For each model on each mode, we carried out a 10-fold cross-validation. Such procedure resulted in 8 (modes) \times 4 (models) = 32 BIC values and CV RMSE values. Table II shows the comparisons among the four models when using $\{\text{Primary-Justify}, \text{Count}\}$ and $\{\text{Primary-Justify}, \ln(\text{Count})\}$ LO modes respectively. It shows that across both modes, the IFM is more accurate (both lower BIC and RMSE) than the PFM; similarly, the latter is more accurate than AFM+Tell and AFM-Tell. However, it is harder to compare AFM-Tell and AFM+Tell. For example, on $\{\text{Primary-Justify}, \text{Count}\}$ mode, although AFM-Tell has lower BIC than AFM+Tell 9037 vs. 9058, the latter has lower RMSE than the former: 4.456E-01 vs. 4.459E-01. So on both $\{\text{Primary-Justify}, \text{Count}\}$ and $\{\text{Primary-Justify}, \ln(\text{Count})\}$ modes, we have IFM > PFM > AFM+Tell, AFM-Tell. Such pattern is consistence across all eight modes.

Table I. {Primary, Pretest-Primary, Primary-Justify, Overall} Learning Opportunity Modes

		Primary	Pretest-Primary
AFM-Tell	N_{ik}	Elicit	Pretest+Elicit
AFM+Tell	N_{ik}	Elicit+Tell	Pretest+Elicit+Tell
PFM	S_{ik}	Success(Elicit)	Success in (Pretest + Elicit)
	F_{ik}	Failure(Elicit)	Failure in (Pretest + Elicit)
IFM	S_{ik}	Success(Elicit)	Success in (Pretest + Elicit)
	F_{ik}	Failure(Elicit)	Failure in (Pretest + Elicit)
	T_{ik}	Tell	Tell
		Primary-Justify	Overall
AFM-Tell	N_{ik}	Elicit + SelfExplain	Pretest+ Elicit+SelfExplain
AFM+Tell	N_{ik}	Elicit+Tell + SelfExplain +Explain	Pretest+ Elicit+Tell + SelfExplain+Explain
PFM	S_{ik}	Success in (Elicit + Self-Explain)	Success in (Pretest+ Elicit + Self-Explain)
	F_{ik}	Failure in (Elicit + Self-Explain)	Failure in (Pretest+ Elicit + Self-Explain)
IFM	S_{ik}	Success in (Elicit + Self-Explain)	Success in (Pretest+ Elicit + Self-Explain)
	F_{ik}	Failure in (Elicit + Self-Explain)	Failure in (Pretest+ Elicit + Self-Explain)
	T_{ik}	Tell+ Explain	Tell+ Explain

Table II. Compare AFM-Tell, AFM+Tell, PFM and IFM on {Primary-Justify, Count} and {Primary-Justify, Ln(Count)} mode

Model	{Primary-Justify, Count}		{Primary-Justify, Ln(Count)}	
	BIC	10-fold RMSE	BIC	10-fold RMSE
AFM-Tell	9037	4.460E-01	9037	4.459E-01
AFM+Tell	9117	4.470E-01	9058	4.456E-01
PFM	8474	4.235E-01	8461	4.236E-01
IFM	8347	4.217E-01	8321	4.211E-01

In order to compare the performance among four models, Wilcoxon Signed Ranks Tests were conducted on resulted BICs and RMSEs. Results showed that IFM significantly outperformed the PFM across eight modes: $Z = -2.52$, $p = 0.012$ for both BIC and cross-validation RMSE. Similarly, it was shown that across all eight modes IFM beat corresponding AFM-Tell across eight modes significantly on both BIC and RMSE: $Z = -2.52$, $p = 0.012$. Similar results were found between IFM and AFM+Tell in that the former out-performed the latter across eight modes significantly on both BIC and RMSE: $Z = -2.52$, $p = 0.012$.

Comparisons between PFM and AFM-Tell and AFM+Tell showed that PFM beats corresponding AFM-Tell across eight modes significantly on both BIC and RMSE: $Z = -2.52$, $p = 0.012$; and PFM also beat AFM+Tell significantly on both BIC and RMSE: $Z = -2.52$, $p = 0.012$. Finally, comparisons between AFM-Tell and AFM+Tell showed that adding Tells and Explains into LOs did not statistically significantly improve the BIC and RMSE of the corresponding AFM model: $Z = -0.28$, $p = 0.78$ for BIC and $Z = -1.35$, $p = 0.18$ for RMSE respectively. Therefore, our overall results suggested: IFM > PFM > AFM-Tell, AFM+Tell.

Next, we investigated which way of counting LOs is better, using logarithmic or additive tabulation? Wilcoxon Signed Ranks Tests were conducted on comparing the BIC and RMSE of the performances when using Count versus using Ln(Count) on the same model and mode. Results showed using Ln(Count) performed significantly better than using Count: $Z = -2.27$, $p = 0.008$ for BIC and $Z = -2.33$, $p = 0.02$ for RMSE respectively. This analysis is interesting in relation to a long-standing debate about whether the learning curve is exponential (like additive tabulation) or a power law (logarithmic tabulation) [Heathcote et al. 2000]. Our results appear to favor the power law.

Next, we investigated the impact of four LO modes. The BICs and RMSEs were compared among the {Primary, Pretest-Primary, Primary-Justify, Overall} modes regardless of Count and Ln(Count). A pairwise comparisons on Wilcoxon Signed Ranks Tests showed that the {Primary-Justify} modes generated signif-

cantly better models than using {Primary} modes $Z = -2.1, p = 0.036$; the {Primary} modes generated better models than using {Pretest-Primary} and {Overall} $Z = -2.27, p = 0.018$ and $Z = -2.521, p = 0.012$ respectively. While no significant difference was found between {Pretest-Primary} and {Overall} modes. Similar results was found on RMSE. Therefore, it suggested that adding justification steps into LOs is beneficial in that Primary-Justify mode beats Primary; however, adding pretest into the LOs did not produce better models and it may even have resulted worse models: the benefit of adding justification steps into LOs was seemingly washed out by including pretest in the LOs in that {Overall} modes generate worse models than {Primary-Justify} and {Primary}.

To summarize, for modeling the training data, applying IFM model and using {Primary-Justify, Ln(Count)} as LOs generated the best fitting model. Additionally, comparisons among the IFM, PFM, AFM-Tell, and AFM+Tell showed that IFM > PFM > AFM-Tell, AFM+Tell. In this paper, our goal is to compare cognitive models on datasets involving multiple types of instructional interventions. As shown above, for AFM the tell steps can be directly added into existing opportunity count N_{ik} ; For the PFM model, however, there is no direct way how tells should be incorporated. Therefore, in the following we will mainly compare IFM and AFM+Tell. For the convenient reasons, we will refer to AFM+Tell as AFM.

4.2 IFM vs. AFM for Unseen Student Prediction (Type 1)

Next we compared the AFM and IFM models on the task of unseen student prediction. In order to predict unseen student's performance, Student ID was treated as a random factor in both AFM and IFM models. Here we conducted Leave-one-student-out cross-validation. In other words, 64 students resulted in a 64-fold cross validation. Thus, we have 8 (modes) $\times 2$ (AFM vs. IFM) BIC values and Cross-Validation RMSE values.

Table III shows the correpsonding BIC and RMSE values of AFM and IFM models using {Primary-Justify, Ln(Count)} mode. Table III shows that IFM generates better prediction models (both lower BIC and RMSE) than AFM and the difference is large. Such pattern is consistence across all eight modes.

Table III. AFM vs. IFM On Unseen Students with Random Effect Student Parameters		
Model	BIC	64-fold Cross-Validation RMSE
AFM	8724	4.6144E-01
IFM	7952	4.1661E-01

To compare IFM and AFM across eight modes, Wilcoxon Signed Ranks Tests were conducted on both BICs and cross-validation RMSEs. Consistent with the patterns shown in Table III, results showed that IFM is significant better than AFM across eight modes: $Z = -2.52, p = 0.012$ for both BIC and cross-validation RMSE. To summarize, IFM with random student parameter is a better model for predicting unseens students' performances on seen steps than AFM model with random student parameter. The best performance was generated IFM model using {Primary-Justify, Ln(Count)} as LOs.

4.3 AFM vs. IFM for Unseen Step prediction (Type 2).

Finally we compared AFM and IFM models on the task of unseen step prediction. Here we used training dataset and tested each models' prediction using students' post-test performance. For each model on each mode, we carried out a 10-fold cross-validation. Such procedure again resulted in 8×2 BIC values and CV RMSE values.

Table IV shows the results on comparisons for the AFM and IFM models on both {Primary-Justify, Ln(Count)} and {Overall, Ln(Count)} modes. Across the eight LO modes, the performance of AFM reaches its best when using {Primary-Justify, Ln(Count)} mode and IFM reaches its best when using {Overall, Ln(Count)} mode. Table III shows that when using {Primary-Justify, Ln(Count)} mode, the AFM is even

more accurate (both lower BIC and RMSE) than the corresponding IFM model; while when using {Overall, Ln(Count)} LO mode, the IFM is more accurate (both lower BIC and RMSE) than the corresponding AFM. Moreover, the best IFM model, using {Overall, Ln(Count)} LO mode, is still better than the best AFM which using {Primary-Justify, Ln(Count)} LO mode. Thus, cross 8 modes on both AFM and IFM, the best prediction model is still generated by IFM but using {Overall, Ln(Count)} LO mode.

Table IV. AFM vs. IFM On Predicting Post-test Performance by {Primary-Justify, Ln(Count)} and {Overall, Ln(Count)} modes

Mode	Model	BIC	10-fold RMSE
{Primary-Justify, Ln(Count)}	AFM	2414	4.6632E-01
	IFM	2428	4.6791
{Overall, Ln(Count)}	AFM	2443	4.7027E-01
	IFM	2252	4.4529E-01

In order to compare AFM and IFM across eight modes, Wilcoxon Signed Ranks Tests were again conducted on resulted 8×2 BIC and RMSE results. Result showed that IFM is marginally significant better than AFM across eight modes: $Z = -1.68$, $p = 0.093$ for BIC and $Z = -1.82$, $p = 0.069$ for 10-fold CV RMSE respectively. Previously, the best model for fitting the training dataset and type 1 predictions are generated by IFM using {Primary-Justify, Ln(Count)} LOs; on the task of predicting students' posttest performance (type 2), however, the best model is still IFM but using {Overall, Ln(Count)} LO counts. To summarize, the best performance of IFM is better than the best AFM and across the eight LO modes and IFM is marginally better than AFM model on type 2 prediction.

5. CONCLUSION

In this paper we investigated student modeling on a dataset involving multiple instructional interventions. We proposed a cognitive model named IFM. We compared IFM with AFM and PFM on the training dataset. We determined that including non-standard LOs such as tells and explains as a separated parameter is effective in that the IFM models' out-performance PFM, AFM-Tell, and AFM+Tell across all modes; but for AFM modes, simply adding tells into AFM LO counts did not seemingly significantly improved the AFM model's performance. This is probably because AFM gives a same learning rate for different instructional interventions. For example, under the {Primary, Count} mode, the N_{ik} in AFM+Tell model is *Elicit + Tell*. On one KC, KC_{20} , the AFM had: the learning rate $\gamma_k = 0.011462$. By contrast, the corresponding IFM has three parameters: μ_k for benefit of previous successes on skill k; ρ_k is the coefficient for the benefit of previous failures, and ν_k the coefficient for the benefit of previous tells on skill k. For the same KC, the IFM resulted $\mu_k = 0.083397$; $\rho_k = -0.213746$, $\nu_k = 0.031982$. The values of the three parameters are quite different from each other, which suggested the the benefit of tells is in the middle of the benefit of success and failure. Such patterns on learned parameters between AFM and IFM showed throughout our analysis. It suggested that rather than using one learning rate parameters for different instructional interventions, it is better to break them into categories and learn seperated parameters.

In order to fully exploring the effectiveness of three models, we further compared them on two prediction tasks – unseen student prediction (type 1) and unseen step prediction (type 2). Our results indicate that the IFM model is significantly better than the AFM model on predicting unseen student's performance on seen steps (type 1) and marginal significant better on predicting seen students' performance on posttest (type 2).

Additionally, we examined the impact of including pretest performance in the LOs as well as qualitative justification steps in the LOs. We found that the Primary-Justify mode seems to be most effective. Generally speaking, models trained with logarithmic tabulation outperformed those trained with additive tabulation

probably because the number of prior LOs counts in this study can be relatively large. For example, the average number of primary steps (including both elicits and tells) in the training data varies from 6 for KC_1 to 83 for KC_{20} .

Even though IFM model performed the best on modeling the training data on both type 1 and type 2 predictions, its performance is heavily dependent upon the specific prediction task being performed and the way in which the specific LOs were counted. For modeling the training data and type 1 prediction, it is the best to use (Primary-Justify,Ln(Count)) mode; but for type 2 predictions, it was best to include the pretest data as well and thus use (Overall,Ln(Count)) mode for LO counts. Thus we conclude that, for datasets involving multiple learning interventions, IFM is a more robust choice for student and cognitive modeling. However the performance of IFM is heavily dependent upon the specific prediction task being performed and the way in which the specific LOs were counted. Experimenters facing a novel task should engage in some measure of parameter-fitting to determine the best fit.

ACKNOWLEDGMENTS

NSF (#SBE-0836012) and NSF (#0325054) supported this work.

REFERENCES

- BARNES, T. 2005. The q-matrix method: Mining student response data for knowledge.
- BECK, J. E. AND MOSTOW, J. 2008. How who should practice: Using learning decomposition to evaluate the efficacy of different types of practice for different types of students. See Woolf et al. [2008], 353–362.
- CEN, H., KOEDINGER, K. R., AND JUNKER, B. 2006. Learning factors analysis - a general method for cognitive model evaluation and improvement. In *Intelligent Tutoring Systems*, M. Ikeda, K. D. Ashley, and T.-W. Chan, Eds. Springer, 164–175.
- CEN, H., KOEDINGER, K. R., AND JUNKER, B. 2008. Comparing two irt models for conjunctive skills. See Woolf et al. [2008], 796–798.
- CORBETT, A. T. AND ANDERSON, J. R. 1995. Knowledge tracing: Modelling the acquisition of procedural knowledge. *User Model. User-Adapt. Interact.* 4, 4, 253–278.
- DRANEY, K., PIROLI, P., AND WILSON, M. 1995. *A Measurement Model for a Complex Cognitive Skill*. Erlbaum, Hillsdale, NJ.
- GONG, Y., BECK, J., AND HEFFERNAN, N. 2010. Comparing knowledge tracing and performance factor analysis by using multiple model fitting procedures. In *Intelligent Tutoring Systems*, V. Aleven, J. Kay, and J. Mostow, Eds. Lecture Notes in Computer Science Series, vol. 6094. Springer Berlin / Heidelberg, 35–44. 10.1007/978-3-642-13388-6_8.
- HEATHCOTE, A., BROWN, S., AND D.J.K., M. 2000. The power law repealed: The case for an exponential law of practice. *Psychonomic Bulletin and Review* 7, 2, 185207.
- JORDAN, P. W., HALL, B., RINGENBERG, M., CUE, Y., AND ROSÉ, C. 2007. Tools for authoring a dialogue agent that participates in learning studies. In *AIED*, R. Luckin, K. R. Koedinger, and J. E. Greer, Eds. Frontiers in Artificial Intelligence and Applications Series, vol. 158. IOS Press, Los Angeles, California, USA, 43–50.
- NEWELL, A. AND ROSENBLOOM, P. 1981. *Mechanisms of Skill Acquisition and the Law of Practice*. Erlbaum Hillsdale NJ.
- PAVLIK, P. I., CEN, H., AND KOEDINGER, K. R. 2009. Performance factors analysis – a new alternative to knowledge tracing. In *Proceeding of the 2009 conference on Artificial Intelligence in Education*. IOS Press, 531–538.
- PAVLIK, P. I., YUDELSON, M., AND KOEDINGER, K. 2011. Using contextual factors analysis to explain transfer of least common multiple skills.
- SCHWARZ, G. E. 1978. Estimating the dimension of a model. *Annals of Statistics*. 6, 2, 461464.
- TATSUOKA, K. 1983. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement*. 20, 4, 345–354.
- VANLEHN, K. 2006. The behavior of tutoring systems. *International Journal Artificial Intelligence in Education* 16, 3, 227–265.
- VANLEHN, K., JORDAN, P., AND LITMAN, D. 2007. Developing pedagogically effective tutorial dialogue tactics: Experiments and a testbed. In *Proceedings of SLATE Workshop on Speech and Language Technology in Education ISCA Tutorial and Research Workshop*. 17–20.
- WOOLF, B. P., AÏMEUR, E., NKAMBOU, R., AND LAJOIE, S. P., Eds. 2008. *Intelligent Tutoring Systems, 9th International Conference, ITS 2008, Montreal, Canada, June 23-27, 2008, Proceedings*. Lecture Notes in Computer Science Series, vol. 5091. Springer.

The Simple Location Heuristic is Better at Predicting Students' Changes in Error Rate Over Time Compared to the Simple Temporal Heuristic

A.F. NWAIGWE

AMERICAN UNIVERSITY OF NIGERIA, NIGERIA

AND

K.R. KOEDINGER

CARNEGIE MELLON UNIVERSITY, U.S.A

In a previous study on a physics dataset from the Andes tutor, we found that the simple location heuristic was better at making error attribution than the simple temporal heuristic when evaluated on the learning curve standard. In this study, we investigated the generality of performance of the simple location heuristic and the simple temporal heuristic in the math domain to see if previous results generalized to other Intelligent Tutoring System domains. In support of past results, we found that the simple location heuristic provided a better goodness of fit to the learning curve standard, that is, it was better at performing error attribution than the simple temporal heuristic. One observation is that for tutors where the knowledge components can be determined by the interface location in which an action appears, using the simple location heuristic is likely to show better results than the simple temporal heuristic. It is possible that the simple temporal heuristic is better in situations where the different problem subgoals can be associated with a single location. However, our prior results with a physics data set indicated that even in such situations the simple location heuristic may be better. Further research should explore this issue.

Key Words and Phrases: Error attribution methods, Intelligent Tutoring Systems, learning curves, mathematics

1. INTRODUCTION

Increasingly, learning curves have become a standard tool for evaluation of Intelligent Tutoring Systems (ITS) [Anderson, Bellezza & Boyle, 1993; Corbett, Anderson, & O'Brien, 1995; Koedinger & Mathan, 2004; Martin, Mitrovic, Mathan, & Koedinger, 2005; Mathan & Koedinger, 2005; Mitrovic & Ohlsson, 1990] and measurement of students' learning [Anderson, Bellezza & Boyle, 1993; Heathcote, Brown, & Mewhort, 2002]. The slope of learning curves show the rate at which a student learns over time, and reveals how well the tutor's cognitive model fits what the student is learning. However, these learning curves require a method for attributing error to the "knowledge components" (skills or concepts) in the student model that the student is missing. Knowledge components, concepts and skills will be used interchangeably in this paper. In a previous study using data from the Andes Intelligent tutor [VanLehn et al., 2005], four

Authors' addresses: A.F. Nwaigwe, School of Information Technology and Communication, American University of Nigeria, Nigeria; E-mail : adaeze.nwaigwe@aun.edu.ng; K.R. Koedinger, Human Computer Interaction Institute, Carnegie Mellon University, U.S.A.; E-mail : koedinger@cmu.edu

alternative heuristics were evaluated - simple location heuristic (LH), simple temporal heuristic (TH), model-based location heuristic (MLH) and model-based temporal heuristic (MTH) [Nwaigwe et al., 2007]. When evaluated on the learning curve standard, the two location heuristics LH and MLH, outperformed the temporal heuristics, TH and MTH. However, the generality of performance of these heuristics in other ITS subject domains needs to be tested.

In this study conducted in the mathematics domain, we investigated whether the previous performance of the LH and TH generalized to other ITS domains. We specifically asked if the LH was better than the TH at predicting student changes in error rate over time. We used log data from a Cognitive Tutor on a Scatterplot lesson and implemented the learning curves standard using the statistical component of Learning Factors Analysis [Cen, Koedinger & Junker, 2005; Pirolli & Wilson, 1998].

Our intuition is that the LH may be the better choice for error attribution when knowledge components (KCs) can be determined by the interface location where an action occurs. To justify this, imagine that a worker has homes, H_a and H_b in which to perform tasks A and B respectively. The worker goes to home H_a and attempts task A but fails. The worker abandons the failed task A and goes to home H_b , where he/she succeeds at task B. The assumption is that tasks A and B are associated with different KCs. The worker later returns to location H_a , and this time, is successful at task A. The LH will more rationally attribute the initial failed attempt at H_a to the KC associated with task A since its rule is to attribute error to the first successfully implemented KC at the initial error location. The TH will however, wrongfully put blame on the KC associated with task B since its method of error attribution is to blame the KC associated with the first correctly implemented task.

Sometimes, TH might be a better choice for making error attribution. We believe this to be the case when it is necessary to perform a set of tasks in a prescribed sequence. To elaborate, imagine that homeschooler Bella is required to perform two tasks and in the given sequence – eat breakfast (EB), and do schoolwork (DS) and in any of two locations, L1 and L2 on the dining table of the family's apartment. We again assume that tasks EB and DS are associated with different KCs. Bella decides that she did not like what Mom served for breakfast that morning and goes straight to her schoolwork, DS, at location L1, skipping task EB. However, Bella fails at task DS due to hunger associated distractions. Later, she abandons task DS and revisits and succeeds at task EB at location L2. Bella then goes back location to L1 and completes task DS. In attributing blame, TH will rationally blame the KC associated with task EB. However the LH will wrongfully blame the KC associated with task, DS. These examples imply that it may be better to apply heuristics in making error attribution.

Although an immediate purpose for error attribution is to drive learning curve generation, the assignment of blame problem is more general and affects many aspects of student modeling.

2. ERROR ATTRIBUTION HEURISTICS

A basic assumption of many cognitive models is that knowledge can be decomposed into components, that each component is learned independently of the others and that implementation of a step in the solution of a problem is an attempt to apply one or more knowledge components (KCs). When correct solution steps are generated, either by an expert system or a human expert, the step is often annotated with the KCs that must be applied in order to generate the step. Thus, when a student enters that step, the system can infer that the student is probably (but not necessarily) applying those KCs.

An ITS system can be designed to anticipate and generate some incorrect steps and associated goals, however, it is rare for expert systems or expert authors to anticipate and generate a large number of incorrect steps and corresponding goals. Hence, when the student enters an incorrect step, it is often not clear what KC(s) should have been applied, so the system cannot determine which KC(s) the student is weak on. If the system simply ignores incorrect steps, then it only “sees” successful applications of KCs. It cannot “see” failures of a KC. It may see lots of incorrect steps, but it cannot determine and record what KC(s) to blame for each error [VanLehn et al, 2005] and so, learning curves cannot be generated. This suggests using heuristics.

The tutoring system usually has two clues available: the location of the incorrect step on the user interface and the subsequent steps entered by the student. For instance, if a student makes an error on a step at time 1 and at location A, the student will often attempt to correct it immediately, perhaps with help from the tutor. So if the first correct step, at time 2 is also at location A, and say, that the step is annotated with KC x, then it is likely that the incorrect step at time 1 was a failed attempt to apply KC x. This heuristic allows the system to attribute errors to KCs whenever the system sees a correct step immediately following the target incorrect step, and both steps are in the same location on the user interface.

However, it is not clear how to generalize this heuristic. What if the next correct step is not in the same location? What if there are intervening incorrect steps in different locations? In previous work using data from the Andes Physics Tutor, four automated heuristics for making error attribution (LH, TH, MLH, MTH) were proposed and evaluated guided by whether the heuristic was driven by location or by the temporal order of events [Nwaigwe et al, 2007].

For every error transaction, LH attributes blame to the KC mapped to a subsequent correct entry at the widget location where the error occurred [Anderson, Bellezza & Boyle, 1993; Koedinger & Mathan, 2004; Martin, Mitrovic, Mathan, & Koedinger, 2005] while the TH ascribes blame to the KC that labels the first correct entry in time. When there is no subsequent correct entry with a label of the error location, LH blames the KC with the first correct entry in time, that is, it implements the behavior of TH. When the tutor provides a choice of some KC to blame for an error, the MLH goes with the tutor’s choice otherwise, it simply implements the LH. For an error transaction, MTH also goes with the domain model’s choice if one exists, otherwise it implements the TH.

In this work, we examine the performance of the LH and TH in the mathematics domain. Table I shows sample log transaction from the cognitive tutor for the scatterplot lesson. The table illustrates how the LH and the TH can help resolve the error attribution ambiguity. Columns in table 1 are described thus: “location” column indicates the place on the interface (the interface widget) in which the student made an input; “Outcome” indicates if an input is correct or not, while “Student Model KC” lists the system’s choice of KC which the student should implement.

In row 1, the student makes an error at the location labeled, “var-Oval-1”. The system however does not indicate the KC the student ought to be practicing. To resolve this ambiguity, the LH uses the KC that labels a subsequent correct entry in the same location – see row 5. That is, it chooses “choose variable”. On the other hand, the TH chooses the KC that labels the first correct entry in time, irrespective of interface location. Its choice is “label x-axis”. In row #2, the domain model blames the KC “choose variable” for the student’s error. LH chooses “choose variable” since it is the first correctly implemented KC at the location “var-Oval-1”. TH blames the KC “label x-axis” in this case.

In the prior study, the cognitive model generated by the LH was found to outperform that of the TH and also, the tutor’s original model according to the learning curve

standard. In other words, the LH was better at making error attributions than the other two cognitive models. Compared to the TH, we also found that the error attribution method of the LH was more like that made by human coders. In this work, we conduct our analysis in the math domain and compare the performance of the LH to that of the TH based on the learning curve standard. Our goal is to see if the previous performance of the LH and TH can be generalized to other intelligent tutoring system domains.

Table I. Table illustrating different error attributions made by the 2 methods

#	Location	Outcome	Student KC	Model	KC Error Attributions methods	
					LH	TH
1	var-0val-1	incorrect			choose variable	label x-axis
2	var-0val-1	incorrect	choose variable	choose variable	label x-axis	
3	var-1val-1	correct	label x-axis	label x-axis	label x-axis	
4	var-0val-1	incorrect		choose variable	choose variable	
5	var-0val-1	correct	choose variable	choose variable	choose variable	

3. LEARNING CURVES

Learning curves plot the performance of students with respect to some measure of their ability over time [Anderson, Bellezza & Boyle, 1993; Corbett, Anderson, O'Brien, 1995; Koedinger & Mathan, 2004; Martin, Mitrovic, Mathan, & Koedinger, 2005; Mathan & Koedinger, 2005; Mitrovic & Ohlsson, 1990]. For ITSs, the standard approach is to measure the proportion of knowledge components in the domain model that have been “incorrectly” applied by the student. This is also known as the “error rate”. Other alternatives exist, such as the number of attempts taken to correct a particular type of error. Time is generally represented as the number of opportunities to practice a KC or skill. This in turn may be determined in different ways: for instance, it may represent each new step a student attempts that is relevant to the skill, on the basis that repeated attempts at the KC are benefiting from the student having been given feedback and as-needed instruction about that particular skill and hence may improve from one attempt to the next. If the student is learning the KC or skill being measured, the learning curve will follow a so-called “power law of practice” [Mathan & Koedinger, 2005]. If such a curve exists, it presents evidence that the student is learning the skill being measured or conversely, that the skill represents what the student is learning.

3.1. THE LEARNING CURVES STANDARD

The power law applies to individual skills and does not take into account student effects. The statistical component of Learning Factors Analysis (LFA) extends the power law to a logistic regression model which accommodates student effects for a cognitive model incorporating multiple knowledge components and multiple students [Cen, Koedinger, & Junker, 2005], see equation 1. The following are the assumptions on which equation 1 is based:

1. Different students may know more or less initially. An intercept parameter of this model reflects each student's initial knowledge.

2. Students learn at the same rate. Thus, slope parameters do not depend on the student. Slope parameters reflect the learning rate of each KC which the student model encompasses and are independent of student effect. This assumption made so as to reduce the number of parameters in equation 1 and is further justified since equation 1 is focused on refining a cognitive model rather than on evaluating students' knowledge growth [Draney, Pirolli & Wilson, 1995].
3. Some KCs are more likely known than others. An intercept parameter for each KC captures initial difficulty of the skill.
4. Since some KCs are easier to learn than other, the model of equation 1 uses a slope parameter to reflect this for each skill. Larger values for initial difficulty reflect tougher skills.

$$\ln[p/(1-p)] = \sum \alpha_i X_i + \sum \beta_j Y_j + \sum \gamma_j Y_j T_j. \quad (1)$$

where p – the probability of success at a step performed by student i that requires knowledge component j ; X_i and Y_j – the dummy variable vectors for students and knowledge components respectively; T_j – the number of practice opportunities student i has had on knowledge component j ; α_i – the coefficient that models student i 's initial knowledge; β_j – the coefficient that reflects the initial difficulty of knowledge component j where larger values of initial difficulty reflect tougher skills; γ_j – the coefficient that reflects the learning rate of knowledge component j , given its practice opportunity.

In this paper, the model of equation 1 is used to apply the learning curve standard. Bayesian Information Criterion (BIC) [Wasserman, 2004] is used to estimate prediction risk in the model while loglikelihood is used to measure model fit. Lower BIC scores, mean a better balance between model fit and complexity.

4. DATA SOURCE

The data used for this research was collected as part of a study conducted in a set of 5 middle-school classrooms at 2 schools in the suburbs of a medium-sized city in the Northeastern United States. Student ages ranged approximately from 12 to 14 years. The classrooms studied were taking part in the development of a new 3-year cognitive tutor curriculum for middle school mathematics [Baker, 2005; Baker., Corbett, Koedinger & Wagner, 2004]. Data collected was from the study on these classrooms during the course of a short (2 class periods) cognitive tutor unit on scatterplot generation and interpretation. Scatterplots depict the relationship between two quantitative variables in a Cartesian plane, using a point to represent paired values of each variable.

The scatterplot lesson consisted of a set of problems and for each problem, a student was given a data set to generate a graph. The student then had to choose from a list, the variables that were appropriate for use in the scatterplot (see figure 1); those that were quantitative or categorical; and subsequently whether a chosen variable was appropriate for a bar chart.

Next the student was required to label the X and Y-axis (see figure 2), and to choose each axis bound and scale. The student was then required to plot points on the graph by clicking on the desired position on the graph. Finally, the student was required to answer a set of interpretation questions to reason about the graph's trend, outliers, monotonicity, and extrapolation and in comparison with other graphs. In our dataset, students solved a maximum of six problems and a minimum of two in the scatterplot lesson.

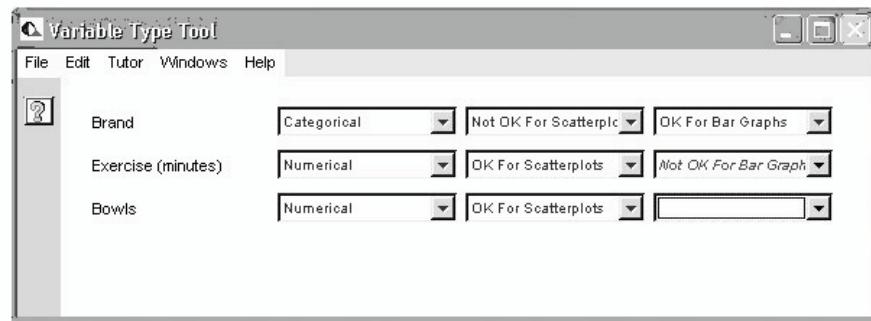


Figure 1 Scatterplot lesson interface for choosing variable type [Baker, 2005]

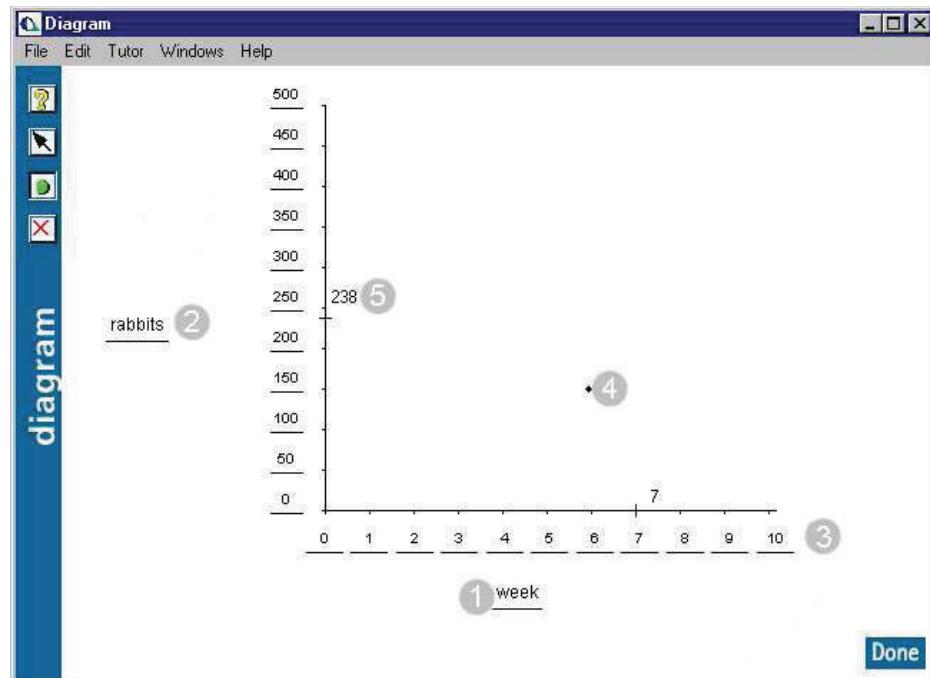


Figure 2 Interface for graph creation in the scatterplot lesson [Baker, 2005]

5. METHODOLOGY

The algorithms for the LH and TH used in this research was implemented in pure java 1.6 and designed to process student log data in MS Excel format. Both algorithms used sequential search. Log data from the cognitive tutor unit on scatterplot generation and interpretation served as input to the programs. The output from each program was the choice of KC codes made by the heuristic being implanted as explained in section 2.

To analyze the cognitive model of each heuristic according to the learning curve standard, the data output from each program was then fit to equation 1 to derive learning behavior. The coefficients of equation 1, initial KC difficulty (β_j), initial student difficulty (α_i) and KC learning rate (γ_j) were used to describe learning behavior for each heuristic. If the intercept of a KC was higher, then, its initial difficulty was lower. Further, if the slope of each KC was higher, then, the faster students learned that skill. For the model of each heuristic, BIC score was used to estimate prediction risk while loglikelihood was used to measure model fit.

6. RESULTS AND DISCUSSION

Table II summarizes the results of the learning curve standard for the student models for both the LH and TH. The results show that the simple location heuristic, LH (BIC score: 7,510.12) shows better fit to the learning curve standard compared to the simple temporal heuristic, TH (BIC scores: 7,703.58). This means that the model of the LH is more reliable and so, a prediction error is more likely to occur if one used the TH model. Loglikelihood score was also better for the LH (-3,370.37) than for the TH (-3,464.93), indicating that the LH model was a better fit to the data than the competing TH model. This shows how the different error attribution methods affect the result.

Table II. Results of the Learning Curve Standard

		TH	LH
logLikelihood		-3,464.93	-3,370.37
BIC		7,703.58	7,510.12
Learning Rate (γ_j)	Mean (Std)	0.09 (0.09)	0.133 (0.11)
Initial KC Difficulty (β_j)	Mean (Std)	-1.81 (0.94)	0.08 (1.10)
Initial Student Difficulty (α_i)	Mean (Std)	2.03 (0.61)	-0.00 (0.63)
# of KCs		17	17
# of transactions across entire scatterplot lesson		16,291	16,291
# of students		52	52

Table III. Knowledge Component Details for the two Heuristics

Knowledge Component (KC)	Simple Temporal Heuristic (TH)			Simple Location Heuristic (LH)		
	Ave Opp	β_j (Initial difficulty)	γ_j (learning rate)	Ave Opp	β_j (Initial difficulty)	γ_j (learning rate)
CHOOSE-VAR-TYPE-CAT	6.6	-1.449	0.076	6.6	0.048	0.244
MMS-VALUING-DETERMINE-SET-MAX	6.9	-0.793	0	6.2	1.275	0
MMS-VALUING-DETERMINE-SET-MIN	6.3	-0.361	0.031	6.2	1.587	0.063
QUANTITATIVE-VALUING-FIRST-BIN	6.1	-2.642	0.159	5.5	-0.565	0.163
QUANTITATIVE-VALUING-SECOND-BIN	5.6	-0.947	0	5.4	0.879	0.052
MMS-VALUING-LABELSUSED	6.9	-2.625	0.049	5.8	-0.942	0.219
CHOOSE-VAR-TYPE-NUM	18.2	-1.044	0.038	16.2	0.799	0.044
MMS-VALUING-DETERMINE-SCALE	53.1	-0.069	0.007	50.2	2.364	0
MMS-VALUING-LABELSUSED-PLUS2	5.9	-1.99	0.063	5.8	-0.805	0.187
TEST-SLOPE	3.3	-2.213	0.131	3.3	0.238	0
CHOOSE-OVERALL-REL	5.0	-3.175	0.257	5.2	-0.865	0.149
EXTRAPOLATE	1.7	-2.093	0	1.5	0.206	0
CHOOSE-OK-BG	11.5	-1.708	0.198	11.4	0.263	0.215
CHOOSE-X-AXIS-QUANTITATIVE	4.3	-2.572	0.274	3.2	-0.719	0.314
CHOOSE-Y-AXIS-QUANTITATIVE	3.7	-2.618	0.018	3.2	-1.884	0.276
MMS-VALUING-DETERMINE-MIN	6.3	-3.053	0.119	5.8	-1.106	0.139
MMS-VALUING-DETERMINE-RANGE	6.3	-1.357	0.147	6.1	0.584	0.196

Generally, we observed that, the LH performed better than the TH when the student failed to successfully complete an attempted step and subsequently attempted and succeeded at a different step. As shown in table I, the student unsuccessfully attempted a step at location “var-0val-1” (trn # 1 & 2). The student subsequently went to location “var-1val-1”, attempted and succeeded at the new step. While the TH incorrectly blamed “label x-axis” which is the KC associated with the new step at location “var-1val-1”, the LH more rationally blamed “choose variable” which is the KC that should be associated with the step at location “var-0val-1”. Because the LH uses location for error attribution, it correctly assigns blame to the KC associated with the error. TH however, wrongfully blames the first subsequent KC that the student correctly attempts. Of the 16,291

transactions in our dataset, error transactions recorded were 5,733. Of the latter, the LH and TH differed on 1,583 (36%) transactions with respect to error attribution choices.

We also found that both the LH and the TH had the tendency to yield the same result when the student succeeded at a step, even after multiple attempts, prior to attempting and succeeding at a new step. This was the case 64% of the time.

In table III, average practice opportunity, initial KC difficulties and learning rates are given for KCs and used to describe learning behavior for each heuristic. For example, for the KC “CHOOSE-VAR-TYPE-CAT”, the learning rate (γ_j) for the LH was more than 3 times that of the TH. Judging by KC initial difficulty (β_j), “CHOOSE-VAR-TYPE-CAT” appeared more difficult for the model of the TH (-1.449) than for the model of the LH (0.244). The average practice opportunity measured for that skill (6.6), was the same for each heuristic. The latter means that on the average, each student had approximately 7 opportunities to practice the KC “CHOOSE-VAR-TYPE-CAT”.

From table III, for the most part, KC learning rate was higher for the skills in the cognitive model of the LH compared to that for the TH. The trend for initial KC difficulty was in the opposite direction as seen for KCs such as “MMS-VALUING-DETERMINE-SET-MIN”, “QUANTITATIVE-VALUING-SECOND-BIN”, etc. Generally, KCs in the cognitive model for TH appeared more difficult to students initially, when compared to similar KCs in the cognitive model of the LH.

From table II, the mean learning rate for the LH was 0.133(± 0.11) which evaluated higher than that of the TH, 0.09(± 0.09). The mean initial KC difficulty for the LH and TH were 0.08(± 1.1) and -1.84(± 0.94) respectively. The reason for the latter seems to be due to more errors being attributed to later opportunities in the TH than the LH. These results thus illustrate the effects of error attribution.

7. CONCLUSION

In this paper, we investigated the generality of performance of two alternative methods for making error attribution in intelligent tutoring systems - the simple location heuristic and the simple temporal heuristic. Our study was carried out in the mathematics domain using data from a cognitive tutor unit on scatterplot generation and interpretation. In support of previous results obtained in the physics domain, we found that the simple location heuristic was better at predicting students’ changes in error rate over time compared to the simple temporal heuristic. This work shows that simpler, easier-to-implement methods can be effective in the process of making error attribution.

One observation is that for tutors where the KCs can be determined by the interface location (or widget) in which an action appears it is likely that the LH will show better results than the TH. This feature is mostly true of the scatterplot tutor. It is possible that the TH is better in situations where the different problem subgoals can be associated with a single location. However, our prior results with a physics data set indicated that even in such situations the LH may be better. Further research should explore this issue.

We also intend to investigate whether the use of the simple location-based heuristic may improve on-line student modeling and associated future task selection. The availability of datasets from the Pittsburgh Science of Learning Center’s ‘DataShop’ (see <http://learnlab.org>) will facilitate the process of getting appropriate data.

ACKNOWLEDGEMENTS

We would like to thank Hao Cen for his help with the LFA tool.

REFERENCES

- ANDERSON, J. R., BELLEZZA & BOYLE, C. F., 1993. The Geometry Tutor and Skill Acquisition. In Anderson, J. R. (Ed.) Rules of the Mind, Chapter 8. Hillsdale, NJ: Erlbaum.
- BAKER R. S., 2005. Designing Intelligent Tutors that Adapt to when Students game the system. Doctoral Thesis. Human Computer Interaction. School of Computer Science. Carnegie Mellon University.
- BAKER, R.S., CORBETT, A.T., KOEDINGER, K. R., WAGNER A. Z., 2004. Off-Task Behavior in the Cognitive Tutor Classroom: When Students “Game the System”. Proceedings of ACM CHI: Computer-Human Interaction, pps. 383-390.
- CEN, H., KOEDINGER, K. & JUNKER, B., 2005. Automating Cognitive Model Improvement by A* Search and Logistic Regression. In: Proceedings of AAAI 2005 Educational Data Mining Workshop.
- CORBETT A.T., ANDERSON, J.R., O'BRIEN A.T., 1995. Student Modelling in the ACT Programming Tutor. In: Cognitively Diagnostic Assessment. Hillsdale, NJ: Erlbaum.
- DRANEY, K., PIROLLI, P. & WILSON, M. 1995. A Measurement Model for a Complex Cognitive Skill. In Cognitively Diagnostic Assessment. Erlbaum, Hillsdale, NJ
- HEATHCOTE, A., BROWN, S., & MEWHORT, D. J. K., 2002. The Power Law repealed: The Case for an Exponential Law of Practice. Psychonomic Bulletin & Review, vol. 7, pps. 185-207.
- KOEDINGER, K.R. AND MATHAN, S., 2004. Distinguishing qualitatively different kinds of learning using log files and learning curves. In: ITS 2004 Log Analysis Workshop, Maceio, Brazil. pps. 39-46.
- MARTIN, B., MITROVIC, T., MATHAN, S., & KOEDINGER, K.R., 2005. On Using Learning Curves to Evaluate ITS. Automatic and Semi-Automatic Skill Coding With a View Towards Supporting On-Line Assessment. In: Proceedings of the 12th International Conference on Artificial Intelligence in Education. Amsterdam, IOS Press.
- MATHAN S. & KOEDINGER K., 2005. Fostering the Intelligent Novice: Learning From Errors With Metacognitive Tutoring Educational Psychologist. 40(4), pps. 257–265.
- MITROVIC A. OHLSSON S., 1990. Evaluation of a Constraint-Based Tutor for a Database Language. International Journal of Artificial Intelligence in Education. vol. 10, pps. 238-256.
- NEWELL, A. & ROSENBLUM, P., 1981. Mechanisms of Skill Acquisition and the Law of Practice. In: Anderson J., (ed.): Cognitive Skills and their Acquisition, Erlbaum Hillsdale NJ.
- NWAIGWE, A., KOEDINGER, K. ET AL. , 2007. Evaluating Alternative Methods for Making Error Attribution in Intelligent Tutoring Systems. In: Proceedings of the 13th International Conference on Artificial Intelligence and Education. Los Angeles, CA.
- PIROLLI P. & WILSON M. A., 1998. Theory of Measurement of Knowledge Content, Access and Learning, Psychological Review. vol. 105, 1, pps. 58-82.
- VANLEHN, K., LYNCH, C., SCHULTZ, K., SHAPIRO, J. A., SHELBY, R. H., TAYLOR, L., ET AL., 2005: The Andes physics tutoring system: Lessons learned. International Journal of Artificial Intelligence and Education. 15(3), pps.147-204.
- WASSERMAN, L., 2004. All of Statistics: A Concise Course in Statistical Inference. Springer.

Items, skills, and transfer models: which really matters for student modeling?

Y. GONG AND J. E. BECK

Worcester Polytechnic Institute, U.S.A.

Student modeling is broadly used in educational data mining and intelligent tutoring systems for making scientific discoveries and for guiding instruction. For both of these goals, having high model accuracy is important, and researchers have incorporated a variety of features into student models. However, since different techniques use various features, when evaluating those approaches, we could not easily figure out what is key for a high predictive accuracy: the model or the features. In this paper, to establish such knowledge, we performed empirical studies varying which features the models considered such as items, skills, and transfer models. We found that item difficulty is a better predictor than skill difficulty or student proficiencies on the transfer model. Moreover, we evaluated two versions of the PFA model; the one with item difficulty resulted in slightly higher predictive accuracy than the one with skill difficulty. In addition, prior work has shown that considering student overall proficiencies, not just those thought to be important by the transfer model, works substantially better on ASSISTments data. However, in this study, we failed to find consistency of this phenomenon on the data collected from the Cognitive Tutor.

Key Words and Phrases: Performance factors analysis, item difficulty, student performance, predictive accuracy

1. INTRODUCTION

Student modeling has been broadly used in educational data mining and applications of intelligent tutoring systems (ITS) for discovering scientific truth about student knowledge, performance, behaviors and motivations, with the goal of leading to a better understanding of students. A wide array of research has been conducted based on student modeling, such as research related to “Gaming the system” [2, 10], the impacts of student non-academic strengths on learning [1, 11], and the effect of item order on student learning [16]. Furthermore, a good student model is also indispensable for a successful ITS. Given the effectiveness of ITS [9, 15], findings such as one-to-one tutoring is better than classroom tutoring [3], and that a step-based computer tutor was not outperformed by human tutors [7], give us a sense that a reason for an ITS’s success is its ability to provide individualized tutoring (one-to-one tutoring). Such tutoring relies on the support of an accurate student model in order to understand students.

Our research interest in this paper lies in student modeling. We simply wish to study what makes a good student model. There is more than one criterion for judging the goodness of a student model [21]. In this study, we focus on the student model’s predictive accuracy. Although student models are frequently evaluated, it can be difficult to know what aspect is responsible for a success or failure. As a result, knowledge as to what makes an accurate student model is insufficient. Our goal in this study is to use the same student modeling framework for different evaluations, to construct guidance about what features (student model components) are important for designing an accurate student model.

There are many potential features that can inform a student model. In this study, items, skills and transfer models were chosen for evaluation, as those are the most commonly used components across different student modeling techniques. In addition, it is also meaningful to examine complete student models constructed with those features, as knowledge about whether and how much multiple features can contribute higher accuracy is also significant. Therefore, we evaluated a series of student models.

1.1. STUDENT MODELING FRAMEWORK

Performance Factors Analysis (PFA) is a student modeling approach proposed by Pavlik, et al. in 2009 [19]. It takes the form of logistic regression with student performance as the dependent variable. We chose PFA as our framework as, relative to Bayesian networks, logistic regression is more flexible to incorporate more (or different) predictors.

It is particularly important to note that there are two student models, both of which were named as Performance Factors Analysis. Both models were designed based on the reconfigurations of Learning Factors Analysis [4] by dropping student variable and considering a student's prior correct and incorrect performances. The two models vary in their independent variables. The model presented in [20] estimates item difficulty (i.e. one parameter per question); the other [19] estimates skill difficulty (i.e. one parameter per skill). Note that in the original paper [19], the authors used the term "knowledge components (KC)" while we use the term "skills"). In this paper, we refer to the first model as the PFA-item model; the other is represented as the PFA-skill model.

$$m(i, j \in \text{required_skills}, q \in \text{questions}, s, f) = \beta_q + \sum_{j \in \text{required_skills}} (\gamma_j s_{i,j} + \rho_j f_{i,j}) \quad (1) \text{ PFA-item}$$

$$m(i, j \in \text{required_skills}, s, f) = \sum_{j \in \text{required_skills}} (\beta_j + \gamma_j s_{i,j} + \rho_j f_{i,j}) \quad (2) \text{ PFA-skill}$$

The m s in Equation 1 and 2 are logits (i.e., are transformed by $e^x/(1+e^x)$ to generate a probability). They represent the likelihood of student i generating a correct response to an item. In the equations, $s_{i,j}$ and $f_{i,j}$ are two observed variables, representing the numbers of the prior successful and failed practices done by student i on skill j . The corresponding two coefficients (γ_j and ρ_j) are estimated to reflect the effects of a prior correct response and a prior incorrect response of skill j . Rather than considering all of the skills in the domain, the PFA model focuses on just those skills required to solve the problem.

The PFA-item model estimates a parameter (β_q) for each question representing its difficulty. In the PFA-skill model, as seen in Equation 2, the β parameter has a subscript of j , indicating that it captures the difficulty of a skill. Also, it is moved to the inside of the summation part to incorporate multiple skills, i.e., in PFA-skill an item's difficulty is the sum of its skills' difficulties.

1.2. EXPERIMENTS

The data used in this study are a small portion of the algebra-2005-2006 development data set for the KDD cup competition 2010 from the Cognitive Algebra Tutor. Since the original data set is very large, to form our working data set, we randomly selected 74 students and their performance records, 94,585 steps completed by the students. We don't have access to the transfer model used in this data set. Thus for determining which skills are required in a question, we directly used the skill labels given in the data. There are a number of questions that do not specify which skills are required to solve them. For those questions, we removed them from the data set. Therefore, in the remaining data set, there are 117 algebra skills, including: Addition/Subtraction, Remove constant, Using simple numbers, Using small numbers, etc.

In this study, we did 4-fold crossvalidation at the level of students, and tested the models on held-out students. We chose to hold out at the student level since that results in a more independent test set. We focused on a student model's accuracy in predicting those held-out students' performances. Predictive accuracy is the measure of how well the instantiated model fits the test data. We used two metrics to examine the model's predictive performance on the test data set: Efron's R^2 and AUC of ROC curve (Area under the curve of Receiver Operating Characteristic). Efron's R^2 is a measure of how

much error the model makes in predicting each data point, compared to a model that uses the mean of the those data to predict. A 0 indicates the model does no better than simply predicting the mean; a 1 indicates perfect prediction. A negative value of Efron's R^2 indicates that the model has more error than a model that just simply guesses the mean for every prediction. AUC of the ROC curve evaluates the model's performance on classifying the target variable which has two categories. In our case, it measures the model's ability to differentiate students' positive and negative responses. AUC of 0.5 is the baseline, which indicates random prediction.

In the result section, we report the comparative results by providing the R^2 and AUC measurements across all four folds. To test the differences of the means, we also performed paired two-tailed t tests using the results from the crossvalidation with degrees of freedom of $N-1$, where N is the number of folds (i.e. $df=3$).

2. STUDENT MODEL COMPONENTS

Many student model components could be important for enabling a student model to achieve high accuracy in predicting student performance.

Student proficiencies on required skills are widely used in many student modeling techniques [4, 5, 19, 20]. Since the transfer model is responsible for providing which skills are required to solve a problem, we refer to "using student proficiencies on required skills to predict" as "using transfer models to predict". The transfer model is often treated as the primary component in student modeling, so is the first component we considered.

Our question was simple: how much variance do transfer models account for? Specifically, how much can a model's predictive accuracy benefit from observing a student's prior performances on required skills? To answer this question, we designed a model that solely considers student proficiencies on the transfer model. We accomplished the model on the basis of the PFA-item model by removing the predictor, item difficulty (β_q), from Equation 1, for the reason that item difficulty is not related to the transfer model. Therefore, the new model has student performances on a series of question as the single predictor, so the only variable predicting the possibility of a student's correct is his proficiencies on required skills.

Item difficulty (question difficulty) has been less studied in student modeling, but is used in Item Response Theory (IRT) [22], a generally effective technique for assessing students [8, 22] such as for computer-based testing [6, 14]. Therefore, it is reasonable to infer that item difficulty is an important predictor of student performance. Item difficulty hasn't been widely used in student modeling until recently when the PFA-item model was proposed [20], as well being integrated into Knowledge Tracing [5] in order to better predict student performance [18]. Hence, in student modeling, there were few attempts for exploring the ability of item difficulty to accurately predict student performance.

Similar to how we test the effect of the transfer model in isolation, in order to test the effect of item difficulty we modify the PFA-item model by dropping the part corresponding to student proficiencies (the part inside the Σ in Equation 1). So the model only has the parameter β_q . Since the model has excluded other features, it can be used to discover the pure ability of item difficulty to contribute the model's predictive accuracy.

The last component we are interested to see is skill, rather than item, difficulty. It is also not commonly used, although Learning Factors Analysis [4] uses skill difficulty in the model. Since the PFA-skill model was reconfigured based on the LFA model, it inherits this feature. To examine skill difficulty, we built a model based on the PFA-skill model (Equation 2) and removed the part corresponding to student proficiencies. Only the skill difficulty parameter (β_j) after the sigma sign is left to capture the effect of the required skills for the question.

2.1. RESULTS

In this section, we examine the predictive power provided by different student model components, including item difficulty, skill difficulty and student proficiencies on the skills in the transfer model. Since each of our models only consider a single feature, the results of testing the model can be attributed to that component.

With respect to modeling item difficulty, we were forced to make a compromise when designing the models. Due to a characteristic of the Cognitive Tutor data, it is not sensible to use the question's identity. In the Cognitive Tutor, a question can have multiple steps, each of which typically requires different skills. Therefore, in the Cognitive Tutor, if a question identity occurs multiple times in the student performance records, we cannot simply assume that they concern the same question. For example, a record might be the first step of a question, while another record with the same question identity might be the tenth step of the question. The difficulties of the two steps are probably not the same as they involve different skills and different aspects of the question. For modeling skill difficulty, there is no difficulty, but it presents clear problems for modeling item difficulty. A solution is to build a new question identity combining the original question identity and the skills required in a step [18]. For instance, if the original question id is Q1 and the first step of the question requires "Addition", we can build a new question id, Q1-Addition; while if the tenth step requires "Using small numbers", we have another question id, Q1-UsingSmallNumbers. However, this way results in a very large number of question identities, over 8000 in our data, and it causes a severe computational problem for logistic regression and an inability to fit the model within SPSS, even with increased memory. Therefore, we made a pragmatic decision: for each step, we represented its difficulty using the summation of the difficulty of the original question and the difficulties of the required skills in that step. In this way, the computational cost is greatly reduced and an approximate difficulty for the step can be estimated. The corresponding equation is shown Equation 3.

$$m(i, j, q \in \text{questions}, s, f) = \beta_q + \sum_{j \in \text{required_skills}} (\beta_j + \gamma_j s_{i,j} + \rho_j f_{i,j}) \quad (3)$$

The computationally viable method

Table I shows the comparative results of models, each of which was fit by a single student model component. First, we found that compared to the other student model components, the model using item difficulty results in higher predictive accuracy and the differences in the means are significant. In the comparison of item difficulty vs. skill difficulty, the t-tests resulted in $p=0.02$ in R^2 and $p=0.005$ in AUC. In the comparison between the model using item difficulty and the model using transfer models, the t-tests yielded $p=0.006$ in R^2 and $p=0.48$ in AUC. The p-value in AUC suggests that there is not enough evidence to show that the two models have different classification abilities for the student performances, while the predictive error made by the model using item difficulty is significantly smaller than its counterpart.

Table I Comparative performance on unseen students

Student model component	R^2	AUC
Item difficulty	0.149	0.739
Skill difficulty	0.139	0.720
Student proficiencies on the transfer model	0.132	0.738

The results concerning item difficulty suggest that contrary to the traditional belief that student proficiencies on the transfer model (required skills) are the most important

predictor, instead item difficulty is an even more powerful predictor of student performance. This finding is also consistent with the finding in the study using the data gathered from ASSISTments [13], suggesting that item difficulty can cover more variance of student performance is a general phenomenon across different computer tutors and different populations.

Table I also shows the results of comparing skill difficulty and student proficiencies on the transfer model. The results of the two metrics do not agree with each other, but both differences are found to be reliable: $p=0.03$ in R^2 and $p=0.02$ in AUC; therefore, it is still uncertain about whether skill difficulty or student proficiency is more important for predicting student performance.

3. STUDENT MODELS

Aside from getting knowledge about how components perform in isolation, it is also important to understand the predictive accuracy of complete models using multiple features, such as the full PFA-item model (Equation 1). It makes sense to examine a complete model as a whole for the following two reasons. First, from a scientific point of view, it is interesting to find out whether different features account for unique variation in predicting student behavior, or whether one feature largely subsumes another. Second, from a practical point of view, knowing whether adding a certain feature is a positive step for improving the model's predictive accuracy helps design a compact, yet effective student model.

3.1. THE TWO VERSIONS OF THE PFA MODEL

We examine the two PFA models, PFA-item and PFA-skill, because direct comparisons between these two have never been performed.

When the PFA-skill model was presented, the designers of the model, using data from Cognitive Tutors, performed evaluations against a well-established student model, Knowledge Tracing, and found that on the student population of Cognitive Tutor, the PFA-skill model is somewhat superior to KT [19]. On the other hand, our prior work applied the PFA-item model to another tutor, ASSISTments, and found that the PFA-item model was markedly superior to KT [12]. Since there have been no studies comparing PFA-item and PFA-skill at the same time and on the same population, we are unsure about the reason for this difference of results.

3.2. A VARIANT OF THE PFA MODEL: THE OVERALL PROFICIENCIES MODEL

We proposed the *overall proficiencies* model, a variant of the PFA-item model, in prior work [13]. This model incorporates the idea that student proficiencies on all skills, not just those the transfer model thinks are required for a particular item, could be important for better predicting student performance on the item. Prior work found that this model performed significantly better than the PFA-item model on ASSISTments data [13]. In this study, we wanted to extend this model to another tutoring environment, Cognitive Tutor, and another population, students of Cognitive Tutor. Since there are many differences between the two systems, we aimed to use this study to better understand the overall proficiencies model.

We had two hypotheses to support the reasonableness of the overall proficiencies model. The first is that the assumption of using transfer models to predict might not always hold, as transfer models assume that only student proficiencies on the required skills have impact on question solving. In other words, student proficiencies on non-required skills are independent of student performance on the problem. However, it is not always true for all ITSs, perhaps due to the possibility that there are relationships between required skills and non-required skills, which are not well captured by the

transfer model; or perhaps problems involve a broader range of skills than the subject matter expert believed and encoded in the transfer model. Second, since in some student modeling techniques, student ability is viewed as a factor helpful for producing higher model accuracy [4, 17], we assume that a student’s overall proficiencies can be treated as a sign to reflect the student’s overall ability. Thus using those is able to provide the model more information about the student, so as to enable the model to reach higher predictive accuracy.

The overall proficiencies model is built based on the PFA-item model. We modified the PFA-item model’s predictors by replacing *REQUIRED_skills* with *ALL_skills* the subscript on the Σ). The equation is shown as follows.

$$m(i, j \in ALL_skills, q \in questions, s, f) = \beta_q + \sum_{j \in ALL_skills} (\gamma_j s_{i,j} + \rho_j f_{i,j}) \quad (4)$$

The overall
proficiencies
model

3.3. RESULTS

In this section, we compare student models which consider a selected set of student model components. Table II shows the mean performance and per-fold performance for each model and metric. Note that both PFA-item and PFA-skill both outperform the item difficulty model in Table I. Since the transfer model is needed to train both of the PFA models (to get the success and failure counts on each required skill), there is evidence that transfer models are in fact helpful for student modeling.

Table II Model performance on test data

	PFA-item		PFA-skill		Overall proficiencies	
	R ²	AUC	R ²	AUC	R ²	AUC
Fold 1	0.194	0.768	0.181	0.756	0.090	0.694
Fold 2	0.177	0.762	0.179	0.760	0.035	0.709
Fold 3	0.144	0.756	0.142	0.748	-0.178	0.674
Fold 4	0.149	0.746	0.143	0.740	-0.082	0.660
Mean	0.166	0.758	0.161	0.751	-0.034	0.684

The first comparison is between the PFA-item model and the PFA-skill model. We noticed that both models’ predictive accuracies vary considerably across 4 folds. This similar trend is shown in both R² and AUC. In the PFA-item model, the R² values vary from as large as 19.4% to as small as 14.4% (standard deviation of 0.024 across folds). Prior study has applied the PFA-item model to ASSISTments data, but found less variation (standard deviation of 0.008 across folds). This finding suggests that across students, student performances of this data set of Cognitive Tutor have larger variance than that of ASSISTments, which is possibly because that there were only 18 or 19 students in each fold, and thus potentially making the model’s performance unstable.

Second, between the PFA-item and the PFA-skill models, the means of the two measurements suggest that the PFA-item model seems to outperform the PFA-skill model, but the p-value of R² is 0.22, while that of AUC is 0.051. The p value of 0.22 indicates that when comparing the two models in terms of their abilities to minimize error during predictions, we were not able to reject the null hypothesis that the two models achieved different predictive accuracy. In the classification ability, the PFA-item model is marginally reliably better than the PFA-skill model, suggested by p=0.051 in AUC.

Third, it is worth pointing out that on this data set, the PFA-item model produces many more parameters than the PFA-skill model. Since we used a compromised approach to implement the PFA-item model, there are around 950 more parameters (each

per original question identity). If we implemented the model in its original way, it would have around 8000+ parameters (each per created question identity). As a consequence of having additional parameters, the PFA-item model is prone to overfitting. To demonstrate overfitting, in Table III we report the R^2 on the *training* data for each fold. For each model, we compared the R^2 values on training data with the R^2 values on test data. We found that compared to the PFA-skill model, the PFA-item model's performance dropped considerably. Given that the two models performed closely on the test data, the better performance on training data of the PFA-item model did not transfer to test data, suggesting overfitting occurred. However, perhaps with a larger dataset the models' training- and test-set performances would be more similar.

Table III Model performance (R^2) on training data

	PFA-item	PFA-skill	Overall proficiencies
Fold 1	0.229	0.185	0.234
Fold 2	0.284	0.241	0.286
Fold 3	0.231	0.187	0.232
Fold 4	0.237	0.192	0.240
Mean	0.245	0.201	0.248

In this study, we also applied the overall proficiencies model on the Cognitive Tutor data. Interestingly, the model did best in all four folds on the training data, shown in the last column of Table III, but performed the worst on the test data, shown in the last two columns of Table II. Furthermore, the results in Table II have more variability than the PFA-item and PFA-skill models, indicating that the overall proficiencies model performed even more unstably on different students. The results suggest that the overall proficiencies model on the Cognitive Tutor data has serious overfitting problems, and is not suitable for their student records, at least with amount of data used in this study. More discussions about the potential reasons are presented in the section of future work.

4. CONTRIBUTIONS

This study performed explorations of student modeling and contributed basic knowledge to the community.

First, we provided insights in terms of what student model components matter for building an accurate student model of student performance. Different student model components have been used in various student modeling techniques [4, 5, 13, 19, 20], yet thorough inspections of the effectiveness of those components on producing accurate predictions were missing. As a replication and extension of our prior work [13], this work considered one more student model component, skill difficulty, and also tested student model components on another population: students of Cognitive Tutor. Similar to our previous finding, item difficulty is more accurate for predicting student performance than student proficiencies on skills related to the problem. The finding is important, especially given that student proficiencies on related skills are widely used in almost all well-established student modeling techniques. However, using item difficulty can result in a painful model fitting process, depending on the number of items in the data set. Take PFA as example, logistic regression is particularly time-consuming in the presence of a large number of predictors. Therefore, we suggest that although item difficulty works better for forming an accurate student model, decisions should be made based on concrete characteristics of the data, especially given that, for the Cognitive Tutor data, item difficulty only slightly outperformed skill difficulty.

Second, Performance Factors Analysis refers to two different models. In this paper we differentiated them as the PFA-item model and the PFA-skill model. The PFA-skill

model was evaluated against KT and found to be somewhat better [19]; while the PFA-item model was compared with KT as well, but shown with substantially better performance [12]. The direct comparison between the two models has never been performed, leading to uncertainty about their relative performance. In this study, we found that on the Cognitive Tutor data, the PFA-skill model is slightly worse than the PFA-item model, yet with much fewer parameters estimated. The PFA-item model by contrast, for our data set, estimates a large number of parameters. Even with the restricted to be computationally tractable method, it still produced 900+ more parameters, which resulted in a relative 3% improvement. In addition, the PFA-item model is more prone to overfitting. Our results suggest that the PFA-skill model is a good option for predicting student performance data similar to the Cognitive Tutor data.

Finally, we proposed a variant of the PFA model, the overall proficiencies model, in our prior work and showed that the model works substantially better than PFA-item on ASSISTments data [13]. Therefore, applying the model to data from a different tutor environment and a different student population helps achieve a deeper understanding of this new model. We found that the similar trend was not observed on Cognitive Tutor data, as the overall proficiencies model performed poorly on the test data, indicating that the model cannot be generalized on those held-out students. The results suggest the overall proficiencies model does not universally result in a stronger model fit. We have a number of hypotheses for what characteristics would be, so the detailed conditions that make the model perform better are still uncertain for us.

5. FUTURE WORK AND CONCLUSIONS

This study creates several unanswered questions that motivate further research work.

To establish the fundamental knowledge with respect to what component matters for a student model, broader inspections of the components involving different experimental populations and different tutors are still needed, especially given the uncertainty of whether skill difficulty and student proficiencies on the transfer model is able to produce more accurate prediction. In addition, since ASSISTments has several different features from Cognitive Tutor in its pedagogical policies, transfer models, student population, etc., it is meaningful to test the PFA-skill model on the ASSISTments data to see whether it is comparable to the PFA-item model, or whether the differences between the tutors cause one model to outperform the other.

We have no clear answers to explain what major differences between the Cognitive Tutor data and the ASSISTments data cause so different predictive performances of the overall proficiencies model. As we hypothesized in prior study [13], there were at least two potential reasons for the success of the model.

First, the transfer model used in ASSISTments might not be specific enough to explicitly designate all associations between a question and its required skills. Thus, student proficiencies on non-required skills are not independent of the proficiencies on required ones. In other words, there might be relationships between required and non-required skills. Given that the model performed poorly on the Cognitive Tutor data, we think it is due to the following two conditions of the Algebra Cognitive Tutor.

1. The comprehensiveness and correctness of the transfer model.

In fact, the domain expert of ASSISTments intensively encoded a smaller range of skills in the transfer model, assumed that the prerequisite skills are required by default, and thus did not indicate them in the transfer model. Therefore, in ASSISTments, if a question requires Pythagorean Theorem, it is highly likely that it also requires equation solving and square root, but the relationships are not captured by the transfer model. The Cognitive Tutor by contrast, has much more meticulous representation. For example, it

has skills such as “Remove constant” in equation solving, “Remove coefficient” in equation solving, “Entering a given”, etc. Those skills are all hidden beneath a single skill “equation solving” in ASSISTments. Specifically, there are 104 mathematical skills in ASSISTments, covering five strands of middle school Math: algebra, geometry, measurement, number sense and data analysis. By contrast, the Cognitive Tutor has 110 skills just for algebra. The comprehensive transfer model of Cognitive Tutor might be a reason to cause the overall proficiencies model to lose its advantage to deal with implicitly existing relationships between required and non-required skills. An additional factor is the degree of knowledge engineering. The Cognitive Tutors’ transfer models have been refined over years of experiments, while ASSISTments transfer models were made similarly to most ITS: a subject-matter expert designed them. Although we lack data, we suspect the Cognitive Tutor’s transfer models are more accurate, and this factor could certainly impact which student modeling approach works better.

2. The way of tutoring

In ASSISTments, a student enters a single answer to an item, and only has to answer subsidiary “scaffolding” questions in the event the student answers a main question incorrectly. In contrast, in the Cognitive Tutor, no scaffolding questions (steps) are allowed to be skipped. A main question in ASSISTments typically asks higher abstract-level skills, i.e. ask all detailed skills at once; while its scaffolding questions test more specific skills. Thus, flexibly accessing to scaffolding questions causes the model to miss chances to observe student performance associated with fine-grained skills. Consider that if a student makes a successful practice on a skill, it is likely that the student’s knowledge on many other skills benefits from it as well, and just simply we don’t have the chance to observe that. Contrariwise, Cognitive Tutor forces a question to be broken down into steps, so it is not possible for the model to miss any observations of a student practicing on any skills; a correct response of a skill probably has little impact on other skills.

Second, since scaffolding questions are not always used, there were fewer observations of students solving problems that test individual skills in the ASSISTments tutor [13]. Therefore, the student overall proficiencies provides useful evidence to the model to enable the model to more accurately predict. For the Cognitive Tutor data used in this study, due to solving each step being mandatory, there were many more observations for each skill. In addition, within the Cognitive Tutor there was more intensive usage by students. Specifically, for fine-grained algebra skills of the Cognitive Tutor, there were approximately ~100 observations per student per skill; in ASSISTments, with its 104 coarser-grained skills, there were on average fewer than 10 observations per student per skill. Therefore, for the Cognitive Tutor data, dense evidence for a student’s performance on those fine-grained skills might also be a reason for the poor performance of the overall proficiencies.

In summary, this study explored what matters for a student model in terms of producing higher accuracy in predicting student performance. Consistent with our prior finding, for predictive accuracy, item difficulty outperforms transfer models, the most widely used student model components, as well as skill difficulty. The comparisons between the PFA-item and the PFA-skill models brought up an insight that the PFA-skill model is slightly worse than the PFA-item model, but has fewer parameters, a smaller problem of overfitting, and is much more computationally tractable. We extended the overall proficiencies model to the data collected from Cognitive Tutor and found it performed worse than the PFA-item model, suggesting that the overall proficiencies model works well only under certain conditions of an ITS, an area that needs additional exploration.

ACKNOWLEDGEMENTS

For the full list of over a dozen funders please see <http://www.webcitation.org/5xp605MwY>.

REFERENCES

- [1] ARROYO, I., and WOOLF, B. 2005. Inferring Learning and Attitudes from a Bayesian Network of Log File Data. Proceedings of the 12th International Conference on Artificial Intelligence in Education. pp.33-40.
- [2] Baker, R.S., Corbett, A.T. and Koedinger, K.R. (2004) Detecting Student Misuse of Intelligent Tutoring Systems. Proceedings of the 7th International Conference on Intelligent Tutoring Systems, 531-540.
- [3] Bloom, B.S.: The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational Researcher* 13, 4-16 (1984)
- [4] Cen, H., Koedinger, K. and Junker, B.: Learning Factors Analysis - A General Method for Cognitive Model Evaluation and Improvement. Proceedings of the 8th International Conference on Intelligent Tutoring Systems. pp. 164-175. (2006)
- [5] Corbett, A. & Anderson, J. (1995) Knowledge tracing: Modeling the acquisition of procedural knowledge. User modeling and user-adapted interaction. 4: pp. 253-278.
- [6] www.ets.org
- [7] Evens, M., Michael, J.: One-on-one Tutoring By Humans and Machines. Erlbaum, Mahwah (2006)
- [8] Feng, M., Heffernan, N.T., & Koedinger, K.R. (2009). Addressing the assessment challenge in an Intelligent Tutoring System that tutors as it assesses. *The Journal of User Modeling and User-Adapted Interaction*. Vol 19: p243-266.
- [9] Feng, M., Heffernan, N. & Beck, J.(2009) Using Learning Decomposition to Analyze Instructional Effectiveness in the ASSISTment System. Proceedings of the 2009 Artificial Intelligence in Education Conference. IOS Press. pp. 523-530.
- [10] Gong, Y., Beck, J., Heffernan, N. T. & Forbes-Summers, E. (2010)The impact of gaming (?) on learning at the fine-grained level. In Aleven, V., Kay, J & Mostow, J. (Eds) Proceedings of the 10th International Conference on Intelligent Tutoring Systems (ITS2010) Part 1. Springer. Pages 194-203.
- [11] Gong, Y., Rai, D. Beck, J. E. & Heffernan, N. T. (2009) Does Self-Discipline impact students' knowledge and learning? In Barnes, Desmarais, Romero & Ventura (Eds) Proc. of the 2ndInternational Conference on Educational Data Mining. Pp. 61-70. ISBN: 978-84-613-2308-1.
- [12] Gong, Y., Beck, J. E., Heffernan, N. T. (2010) How to Construct More Accurate Student Models: Comparing and Optimizing Knowledge Tracing and Performance Factors Analysis. *International Journal of Artificial Intelligence in Education*. Accepted, 2010.
- [13] Gong, Y., Beck,J. E. (Accepted) Looking beyond transfer models: finding other sources of power for student models,. Accepted to the 19th International Conference on User Modeling, Adaptation and Personalization. Girona, Spain.
- [14] www.grockit.com
- [15] 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.
- [16] Pardos, Z.A., Heffernan, N.T. (2009). Determining the Significance of Item Order In Randomized Problem Sets. In Barnes, Desmarais, Romero & Ventura (Eds) Proc. of the 2nd International Conference on Educational Data Mining. pp. 111-120. ISBN: 978-84-613-2308-1
- [17] Pardos, Z. A., Heffernan, N. T. Modeling Individualization in a Bayesian Networks Implementation of Knowledge Tracing. The 18th Proceedings of the International Conference on User Modeling, Adaptation and Personalization. (2010)
- [18] Pardos, Z. A., Heffernan, N. T. (Accepted) KT-IDE: Introducing Item Difficulty to the Knowledge Tracing Model. Accepted to the 19th International Conference on User Modeling, Adaptation and Personalization. Girona, Spain.
- [19] Pavlik, P. I., Cen, H. & Koedinger, K. (2009) Performance Factors Analysis - A New Alternative to Knowledge. Proceedings of the 14th International Conference on Artificial Intelligence in Education, Brighton, UK, pp. 531-538.
- [20] Pavlik, P. I., Cen, H., Koedinger, K. : Learning Factors Transfer Analysis: Using Learning Curve Analysis to Automatically Generate Domain Models. Proceedings of the 2nd International Conference on Educational Data Mining. pp.121-130. (2009)
- [21] 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, pp141-148.2009
- [22] van der Linden, & W. J., & Hambleton, R. K. (eds.) (1997). *Handbook of modern item response theory*. New York, NY: Springer Verlag.

Avoiding Problem Selection Thrashing with Conjunctive Knowledge Tracing

K. R. KOEDINGER, P. I. PAVLIK JR., J. STAMPER,
Carnegie Mellon University, United States

T. NIXON, AND S. RITTER
Carnegie Learning Inc., United States

One function of a student model in tutoring systems is to select future tasks that will best meet student needs. If the inference procedure that updates the model is inaccurate, the system may select non-optimal tasks for enhancing students' learning. Poor selection may arise when the model assumes multiple knowledge components are required for a single correct student behavior. When the student makes an error, a deliberately simple model update procedure uniformly reduces the probability of all components even though just one may be to blame. Until now, we have had no evidence that this simple approach has any bad consequences for students. We present such evidence. We observed *problem selection thrashing* in analysis of log data from a tutor designed to adaptively fade (or reintroduce) instructional scaffolding based on student performance. We describe a *conjunctive knowledge tracing* approach, based on techniques from Bayesian networks and psychometrics, and show how it may alleviate thrashing. By applying this approach to the log data, we show that a third (441 of 1370) of the problems students were assigned may have been unnecessary.

Key Words and Phrases: Knowledge tracing, tutor log data analysis, Bayesian inference, blame assignment

1. INTRODUCTION

While educational data mining is often applied to discover patterns of students learning in data collected from instructional software, educational data mining can also be useful for identifying weaknesses in the tutoring systems that generated the data. This work presents an example of such identification revealed from analysis of the data and provides a detailed remedy based on Bayesian inference.

Student modeling depends on an accurate estimate of student knowledge to make effective instructional decisions. Making accurate inferences about what students know is challenging in situations where multiple knowledge components (skills, concepts, etc.) must be brought to bear, but where there is only one observation of student performance. If the student performs correctly, the credit assignment is straightforward. All the components get credit, because we have positive evidence that the student knows all the required components. However, if the student performs incorrectly, it is not necessarily appropriate to blame all the components. Any one or more of the components could be at fault. Determining which ones to blame is not straightforward. The Bayesian network [Millán et al. 2001] and psychometrics [Junker and Sijtsma 2001] literatures indicate how probability theory can be applied to address this problem. In this paper, we show how these ideas can be combined with Bayesian Knowledge Tracing [Corbett and Anderson 1995] to produce a “conjunctive knowledge tracing” approach.

Consider a simple example to illustrate the blame assignment problem. Imagine a tutor for teaching children to evaluate simple arithmetic expressions like “ $3*4+5$ ”. The student model could have knowledge components for each mathematical operator: addition, subtraction, multiplication, and division. The problem “ $3*4+5$ ” requires both multiplication and addition (we say “problem” here, but this argument applies more generally to any “step” in a problem solution that is performed as a separate observable action). If a student gets this problem step correct, we have evidence that they know both

Authors' addresses: K. R. Koedinger, Human Computer Interaction Institute, Carnegie Mellon U., Pittsburgh, PA,. Email: koedinger@cmu.edu; P. Pavlik Jr, HCII, CMU, Email: ppavlik@andrew.cmu.edu; J. Stamper, HCII, CMU, Email: jstamper@cs.cmu.edu; T. Nixon, Carnegie Learning Inc., Pittsburgh, Email: tnixon@carnegielearning.com; S. Ritter, Carnegie Learning Inc. Email: sritter@carnegielearning.com.

the multiplication and addition components. If the student is incorrect, it could be that the student does not know multiplication and does not know addition, but it is also possible that the student knows addition but not multiplication or even multiplication but not addition. Consider the case where we have evidence from previous problems that the student is near mastery on addition, but has been struggling with multiplication. For example, the student has been successful on most problem steps that involve addition alone, like “14+3”, but has struggled on problems that involve multiplication alone, like “4*8”. In such a case, if a student makes an error on “3*4+5”, it is less likely to be a failure of addition and more likely a failure of multiplication. That is, the student is less likely to have been wrong because of not knowing addition and more likely to have been wrong because of not knowing multiplication.

In such a case, it does not seem appropriate to reduce the probability that the student knows addition as much as we would reduce the probability that the student knows multiplication. Nevertheless, equal blame assignment is simpler and was implemented as part of the original development kit for Cognitive Tutors [Corbett and Anderson 1995] and is currently used in practice in the widely distributed Carnegie Learning Cognitive Tutors [Ritter et al. 2007]. We pursue the problem of assigning blame in proportion to how likely it is that a knowledge component caused the error. Bayesian analysis provides a principled solution [cf. Millán et al. 2001, Junker and Sijtsma 2001].

We want a solution that not only works for two knowledge components (KCs) in combination, but one that generalizes to multiple KCs. For instance, in a harder problem step like 8-3*6, the student model might have two more KCs like “following order of operations” and “dealing with negative numbers”. In this case, we want to distribute the blame appropriately across all four KCs depending on prior estimates of the KC difficulties. KCs with a higher prior probability of being known should receive less blame than KCs with lower probability. Pardos, Heffernan and Ruiz discuss this multiple-KC problem [Pardos et al. 2008]. Their proposed solution is to use additional diagnostic follow-up questions to determine the incorrect KC, and ignore the initial incorrect response to the question as a whole. Similarly, Cognitive Tutor interfaces are typically engineered so that correctness data on multiple individual steps in a problem solution strategy are available [Corbett and Anderson 1995]. However, in both approaches, the fine-grained diagnostic questions or steps (call them “scaffolds”) still sometimes have multiple KCs associated with them. Perhaps more importantly, in situations when this scaffolding is faded and a full question is given, neither approach provides an integrated diagnosis of the knowledge needed both for the relevant steps and for composing the steps together [Heffernan and Koedinger 1997]. A more elegant solution would be useful.

2. REVIEW OF KNOWLEDGE TRACING

Knowledge tracing is the student model update procedure used in Cognitive Tutors [Corbett and Anderson 1992]. For each knowledge component (KC), there is a two state hidden Markov model wherein there is a probability that the student is initially in either the known state (we use K_1 to represent this probability for “knowing” KC_1 or $Know-KC_1$) or the unknown state ($1-K_1$). There are three other parameters per KC: a slip probability (S) that a student will be incorrect even though they know the KC, a guess probability (G) that a student will be correct even though they do not know the KC, and a learning transition probability (T) that the student will learn at a particular tutoring opportunity and thus transition from the unknown to the known state. Because the challenge of the multiple-KC problem is in blame assignment, we only review here how the probability the student knows a KC is updated after an error observation (see Reye [1998] for a complete set of equations for knowledge tracing and related alternatives).

$$\begin{aligned} P(\text{Know-KC}_1 | \text{Error}) &= \frac{P(\text{Error} | \text{Know-KC}_1)}{S} * \frac{P(\text{Know-KC}_1)}{K_1} / \frac{P(\text{Error})}{[K_1 * S + (1 - K_1) * (1 - G)]} \end{aligned} \quad (1)$$

The simplistic generalization of Equation 1 to the case where multiple KCs are involved on an incorrect step is to update each KC in the same way, that is, all required components are fully and equally blamed.

Table I. Example Consequences of Alternative Knowledge Tracing (KT) Approaches

Step	Knowledge Estimates				
	KC Required			Standard KT	Conjunctive KT
	Add	Mult	Correct		
3*4+5	1	1	0	0.960	0.300
6+3	1	0	1	0.700	0.270
7+4	1	0	1	0.938	0.270
4*7+3	1	1	0	0.990	0.999
				0.893	0.267
				0.999	0.287

Table I illustrates the results of standard knowledge tracing (see Standard KT columns) for a situation like the one described above. This simplified example is intended to clarify the process and consequences of the simplistic rule for blame, but, as we describe below, this example has the essential character of actual student data collected by an intelligent tutor in school use. The example assumes the student has mastered the knowledge component Add ($K_1 = .96$) but not Multiply ($K_2 = .3$). The probabilities of slipping, guessing, and learning parameters are set at 0.05, 0.2, and 0.25, respectively, for both KCs in this example. When a student makes an error on a problem step involving both Add and Multiply, like “3*4+5”, the estimates of knowing Add and Multiply are updated as follows. The estimate for Add (K_1) is updated according the formula above ($.05 * .96 / [.96 * .05 + (1 - .96) * (1 - .2)]$) to be 0.6. Knowledge tracing has a Markov property such that KCs have a probability of transitioning from the unknown state to the known state, that is, of being learned at each opportunity to learn. The transition probability in this example is 0.25 and when we apply it ($.6 + (1 - .6) * .25$) we get a new value for $K_1 = 0.7$. The analogous computations yield a new value for the Multiply, $K_2 = 0.27$.

The key point is that the Add KC drops significantly, to 0.70 – exactly as much as if the student had made an error on a problem step involving addition only (like 5+7). A sensible response of an intelligent tutor to this updated student model is to help the student get Add back up to mastery (a .95 threshold is used in Cognitive Tutors) by giving the student further practice (and as-needed instruction) on a problem involving Add (e.g., “6+3”). In fact, in this scenario, a student would have to get two problems involving Add right before getting back to mastery – see the 6+3 and 7+4 rows in Table I. The first raises the estimate to .938, still below a .95 mastery threshold, and the second to .990. If the student subsequently gets another problem with both KCs (e.g., “4*7+3”) wrong, the estimate for Add would again drop back below mastery. Another problem involving Add would then be selected. This would be wasting student time and energy if, in fact, they got the combined problem (“3*4+5”) wrong because of not knowing Multiply. In fact, the tutor and student might continue to thrash with the tutor repeatedly giving unneeded easy problems after the student errs on a harder problem.

Gong, Beck, & Heffernan [Gong et al. 2010] mentioned limitations of the knowledge-tracing algorithm when a problem or step is coded with multiple knowledge components. They were not addressing the issue, like we are, of on-line updates of the student model estimates of the probability a component has been learned. Others [Millán et al. 2001,

Junker and Sijtsma 2001] have presented relevant applications of Bayesian inference to address conjunctive combinations of skills and we build on that work.

3. CONJUNCTIVE KNOWLEDGE TRACING FOR FAIR BLAME ASSIGNMENT

The algorithm we present modifies knowledge tracing by changing the equations that deal with updating the student model after a student error (see Eq 1). The equations for updating after correct student responses are kept the same.

We present the case for two KCs first and generalize below to the case where multiple KCs are needed. Both the $P(\text{Error}|\text{Know-KC}_1)$ and $P(\text{Error})$ equations need to be modified. We use K_1 and K_2 to indicate the probabilities that KC_1 and KC_2 are known, S_1 and S_2 for their slip parameters, and G_1 and G_2 for their guess parameters. We start with $P(\text{Error})$, because it is simpler. An observed error can result from an unobserved error either in the execution of KC_1 or in the execution of KC_2 . An error in the execution of a KC occurs either when the KC is known but the student slips (e.g., $K_1 \cdot S_1$) or when the KC is unknown and the student does not guess correctly (e.g., $(1-K_1) \cdot (1-G_1)$). This formulation is shown in Equation 2.

$$\begin{aligned} P(\text{Error}) = & K_1 S_1 + (1 - K_1)(1 - G_1) + K_2 S_2 + (1 - K_2)(1 - G_2) - \\ & [K_1 S_1 + (1 - K_1)(1 - G_1)][K_2 S_2 + (1 - K_2)(1 - G_2)] \end{aligned} \quad (2)$$

We can find $P(\text{Error}|\text{Know-KC}_1)$ by plugging $K_1=1$ into the Equation 2 above and the result is shown in Equation 3.

$$\begin{aligned} P(\text{Error}|\text{Know-KC}_1) = & S_1 + K_2 S_2 + (1 - K_2)(1 - G_2) - [S_1][K_2 S_2 + (1 - K_2)(1 - G_2)] \\ = & S_1 + (1 - S_1)[K_2 S_2 + (1 - K_2)(1 - G_2)] \end{aligned} \quad (3)$$

An alternative formulation of Equation 2 that is easier to compute and easier to generalize to many KCs is shown in Equation 4.

$$\begin{aligned} P(\text{Error}) &= 1 - P(\text{Correct}) \\ &= 1 - [K_1(1 - S_1) + (1 - K_1)G_1][K_2(1 - S_2) + (1 - K_2)G_2] \end{aligned} \quad (4)$$

Equation 4 computes the probability of error as one minus the probability of correct performance. To get a step correct requires that both KC_1 and KC_2 are executed correctly, which can be computed as the product of the probabilities of executing each KC correctly (this approach assumes KC execution is independent). Correct execution of a KC occurs either when the KC is known and the student does not slip (e.g., $K_1(1-S_1)$) or when the KC is unknown and the student guesses correctly (e.g., $(1-K_1)G_1$).

The combined update formula (Equation 5) gets applied for each KC, as was done in the example above. Applying this approach to the example above, we get the values shown in the “Conjunctive KC” columns in Table I. After the student made an error on “3*4+5”, the estimate for Add (K_1) was updated according to the formulas above to 0.94.

$$\begin{aligned} & P(\text{Know-KC}_1|\text{Error}) \\ &= \frac{P(\text{Error}|\text{Know-KC}_1)}{P(\text{Error})} \quad * \quad \frac{P(\text{Know-KC}_1)}{P(\text{Know-KC}_1)} / \quad P(\text{Error}) \\ &= \frac{Eq.3}{Eq.4} \quad * \quad \frac{Eq.3}{Eq.4} / \quad Eq.4 \\ &= \frac{(.05+(1-.05)[.3*.05+(1-.3)(1-.2)])}{.96} * \quad \frac{.96}{1-[.96(1-.05)+(1-.96).2][.3(1-.05)+(1-.3).2]} \end{aligned} \quad (5)$$

Applying the learning (or transition) probability $(.94 + (1-.94)*.25)$ yields a new value for $K_1 = 0.955$. The analogous steps yield a new value for the Multiply, $K_2 = 0.297$. Unlike Standard Knowledge Tracing, the estimate for Add, at 0.955, stays above the mastery threshold of .95 and thus the tutor would not assign a potentially unnecessary addition problem. The potential is thus reduced for unproductive cycling back and forth or thrashing between hard and easy problems that may occur with standard knowledge tracing (as illustrated in Table I).

The key insight for blame assignment with two KCs is that the probability of being incorrect given that KC_1 is known is no longer just the probability of slipping on KC_1 .

There is also a chance that the student made an error in executing KC_2 . To generalize to multiple KCs, we need the $P(\text{Error}|\text{Know-}KC_1)$ formula to account for the possibility that an error can result from failure to execute on any of the other needed KCs.

First, Equation 6 shows the general equation for $P(\text{Error})$ when we use the 1- $P(\text{Correct})$ formulation (as anticipated in Equation 4) and compute $P(\text{Correct})$ as the product of executing all of the N KCs correctly:

$$\begin{aligned} P(\text{Error}) &= 1 - P(\text{Correct}) \\ &= 1 - \prod_{i=1}^N [K_i(1 - S_i) + (1 - K_i)G_i] \end{aligned} \quad (6)$$

Now, for the general equation of $P(\text{Error}|\text{Know-}KC_j)$ we need to a way to compute the disjunction (logical or) of executing incorrectly all of the required KCs besides KC_j . Because conjunctions are simpler to compute than disjunctions, we use the transformation in Equation 7 to formulate Equation 8.

$$\begin{aligned} P(A \text{ or } B \text{ or } C) &= \text{not (not } P(A) \text{ and not } P(B) \text{ and not } P(C)) \\ &= 1 - [1 - P(A)][1 - P(B)][1 - P(C)] \end{aligned} \quad (7)$$

Equation 8 replaces the term in Equation 3 for incorrect execution of K_2 with the disjunction of incorrect execution of all the required KCs but KC_j . Thus, note the use of “excluding KC_j ” in Equation 8. And note, as per Equation 7, the use “1-” both outside and inside the product (\prod).

$$\begin{aligned} P(\text{Error}|\text{Know-}KC_j) &= S_j + (1 - S_j) * (1 - \prod_{i \in \text{KCs, excluding } KC_j} [1 - [K_i(1 - S_i) + (1 - K_i)G_i]]) \end{aligned} \quad (8)$$

Finally, Equation 9 is the Conjunctive Knowledge Tracing alternative to blame assignment in Standard Knowledge Tracing (Equation 1) and it completes the generalization from two KCs (Equation 5) to any number of KCs.

$$\begin{aligned} P(\text{Know-}KC_1|\text{Error}) &= P(\text{Error}|\text{Know-}KC_1) * P(\text{Know-}KC_1) / P(\text{Error}) \\ &= Eq.8 * K_1 / Eq.6 \end{aligned} \quad (9)$$

4. CONJUNCTIVE KNOWLEDGE TRACING ON REAL DATA

In the introduction, we illustrated the possibility of a thrashing problem that can result from unfair blame assignment whereby a student is repeatedly assigned a hard problem (which they get wrong) and then unnecessary easy problems (which they tend to get right). We turn to a demonstration of this thrashing problem in real student use of a tutor. We then describe how use of Conjunctive Knowledge Tracing can alleviate this problem. The data come from 120 students working on a geometry area unit of the Bridge to Algebra Cognitive Tutor and, in particular, from an experiment to test a new KC model produced through a human-machine discovery method [Stamper and Koedinger 2011].

This implementation of the tutor used standard knowledge tracing, but we did make a change to the problem selection algorithm designed to create a better learning experience. The original problem selection tries to find problems that have the most opportunities for the student to address their least-mastered KCs (along with other factors, like minimizing the number of mastered KCs and encouraging variety). In the usual situation where there is only one KC per problem step this has been a reasonable approach. However, when there are multiple KCs per step, this current “maximize unmastered” algorithm criteria for problem selection will prefer problems that involve more unmastered KCs per step (harder problems) over problems that have fewer unmastered KCs per step (easier problems). In order to create a gentle slope in the learning trajectory, we modified the original problem selection algorithm to select problems that have as few unmastered KCs (but at least 1) as possible. Thus, students are more likely to be given easier (but not mastered) problems first and then, once these appear to be mastered, more complex

problems are selected. If, in turn, evidence from poor performance on complex problems suggests weaknesses in specific component KCs, easier problems will be selected again to bolster student mastery before returning to hard problems. The intention, then, is to adjust difficulty (fading or reintroducing scaffolding) to optimally adapt to student needs. This change revealed the thrashing problem and a practical weakness of standard knowledge tracing when multiple KCs are required on a step. The goals of the change in problem selection were to adaptively fade and “unfade” (reintroduce) scaffolding based on student performance. Fading occurs in transition from “scaffolded” problems, which tend to have 1 KCs per step, to “unscattered” problems, which tend to have key steps with multiple KCs. It is adaptive in that the transition occurs after students have demonstrated mastery of the KCs in the scaffolded problems. Scaffolding may be reintroduced based on evidence of too much failure on unscattered problems.

4.1. Results: Problem selection thrashing from poor blame assignment

Similar to the arithmetic example above, we modified a geometry area unit of the Bridge to Algebra Cognitive Tutor to include a mix of harder problem types in which some steps require many KCs (e.g., setting and executing subgoals to find a square area, circle area, and the difference) and easier problem types in which steps require just one or a few KCs (e.g., subtracting two areas). Four types of problems culminated with the student finding the area of an irregular shape (e.g., the left-over area when a circle is cut from a square) from the regular shapes that make it up. To aid understanding of the example of real student performance shown in Table II, we describe these problem types. The easiest problem type, called an “area scaffold problem” and displayed as Easy in Table II, gives the areas of the component shapes to focus students’ attention on how to combine them to find the irregular shape rather than on finding component areas themselves. The student need only recognize the need for area composition (the Comp KC in Table II) and perform the addition or subtraction (AddAreas and SubtrAreas KCs in Table II). The slightly less easy “table scaffold” problems (displayed as Easy’ in Table II) require the student to find the regular areas on their own, but explicitly prompt (or scaffold) the student to do so with a labeled column in a table interface widget where the areas are to be entered. While these problems require area computations (see the Area KC in Table II), those computations are separate steps in the interface and so the Area KC is not involved in the “composition” step to compute the irregular area that is displayed in Table II. In the harder “no scaffold” problems, students are asked to enter only the final irregular area (requiring up to four KCs in a single step) without any interface support to first find the component areas.

Turning to the student performance data, we found that the new problem selection algorithm described above worked well in that the easiest problem type (area scaffold) tended to be selected before the somewhat less easy problem type (table scaffold) and these before the hardest problem types (problem scaffold and no scaffold). However, we were surprised at how many of the easier problems students were given. On closer inspection we found the kind of cycling between easy and hard we illustrated above.

Table II provides an example from one of the students. The results are displayed starting after the student has been successful on two Easy problems and failed on a Hard problem. Before describing this example in more detail, first note how the student keeps getting assigned many Easy problems (and succeeds at them). These problems were assigned based on standard knowledge tracing (SKT), but, if conjunctive knowledge tracing (CKT) had been used, the five problems in the bolded row numbers (5, 8, 10, 12, and 14) would not have been assigned. In these rows, all of the CKT estimates are above 0.95 whereas some of the SKT estimates are not (see the bolded numbers). SKT assigns

these Easy problems because when errors are made on Hard problems, it attributes too much blame to easy KCs (SubtrAreas & AddAreas) that should be primarily attributed to hard KCs (SubGoal).

Table II. Problem selection thrashing from poor blame assignment in real student data.

Row	Prob	Corr	Standard KT Estimates					Conjunctive KT Estimates				
			Comp	Subtr	Add	Area	Sub	Comp	Subtr	Add	Area	Sub
			Type	Areas	Areas		Goal		Areas	Areas		Goal
1			0.98	1.00	0.62	0.62	0.48	0.98	1.00	0.86	0.86	0.70
2	Easy'	0	0.98	1.00				0.98	1.00			
3	Easy'	1	0.89	0.98				0.94	0.99			
4	Easy	1	0.98		0.62			0.99		0.86		
5	Easy	1	1.00		0.91			1.00		0.97		
6	Hard	0	1.00	1.00		1.00	0.48	1.00	1.00		1.00	0.70
7	Hard	0	1.00	0.98		1.00	0.38	1.00	1.00		1.00	0.68
8	Easy	1	0.99	0.87				1.00	1.00			
9	Hard	0		0.97		0.98	0.35		1.00		1.00	0.67
10	Easy	1	1.00	0.86				1.00	1.00			
11	Easy'	0	1.00		0.98			1.00		0.99		
12	Easy	1	1.00		0.90			1.00		0.98		
13	Hard	0		0.97		1.00	0.34		1.00		1.00	0.58
14	Easy	1	1.00	0.85				1.00	1.00			
15	Hard	1		0.97		0.97	0.34		1.00		1.00	0.51
16	Hard	1		0.99		0.99	0.79		1.00		1.00	0.88

Going through Table II in more detail, row 1 shows the KC estimates for SKT and CKT just before this sequence begins. Row 2 shows that an Easy problem was selected next. The estimates of only the KCs that are required for the composition step in that problem are shown. Even though the required KCs are above the 0.95 mastery threshold (at 0.98 and 0.997 respectively), the selection of an Easy problem is appropriate because there are other Area steps (not shown) in this problem (indicated as Easy', rather than just Easy) that are not above mastery (at 0.62). The student gets this composition step wrong (indicated by 0 in the Correct column). The updates for the relevant KCs can be seen in row 3 for both SKT (now 0.89 and 0.98) and CKT (now 0.94 and 0.99). Another Easy problem is selected (row 3), which is appropriate according to both models as the Compose KC is below .95 in both (.89 and .94). The student gets it right.

Now two easy problems are selected (rows 4 and 5) where area addition (AddAreas) is needed instead of area subtraction (SubtrAreas). The SKT estimate of AddAreas is below mastery for both problems, but goes above mastery before the second problem for the CKT estimate (see the bolded .97 vs. .91 in row 5). If problem selection had been driven by CKT, this problem would not have been selected and, arguably, the students' time would not have been wasted practicing mastered skills. (Note that the difference in the AddAreas estimates in Row 1 is caused by the difference in blame attribution on the one Hard problem the student saw before the data shown in Table II.) Rows 6-8 more clearly illustrate this difference in blame attribution. The student gets two consecutive Hard problems wrong and the SKT estimate of SubtrAreas drops to 0.87. However, it is likely that the student's difficulty is not with SubtrAreas, but with the SubGoal KC (knowing to find the areas of an irregular shape by finding the areas of the regular shapes

that make it up). Indeed, the CKT model puts most of the blame for these errors on SubGoal and little blame on SubtrAreas (which does drop slightly from .998 to .997).

4.2. Results: Fair blame assignment saves instructional time

To demonstrate that the example above is not idiosyncratic to the one student, we repeated the analysis illustrated above for all 120 students. We focused on the data from the first curriculum section where some steps are coded with multiple KCs (this is section 3 in Geometry Area unit). We used CKT to produce new KC estimates on each problem solved by each student as illustrated in Table II. We then identified the problems where all KCs involved were above the 0.95 mastery level according to the CKT estimates – like the 5 bolded problems in Table II. Of the 1370 problems, 441 or about 1/3 involved only mastered KCs according to CKT! If the problem selection had been driven by CKT, these problems would not have been given to students. These problems are likely to be unnecessary and are taking student time away from learning more difficult skills. (While the problem selection algorithm is designed to avoid giving mastered problems, 15 of the 1370 problems selected using SKT were mastered – still far below 441.)

Some of the 120 students, those with more prior knowledge, finished this section in as few as four problems (by getting all steps correct). Many others struggled and, like the student shown in Table II, got stuck in this thrashing between too many easier problems they tended to be able to solve and too few harder problems that exercised the composition (or subgoaling) skills they needed to acquire. The student in Table II is typical of these struggling students and, according to conjunctive knowledge tracing, five of the sixteen problems this student was given were unnecessary. For 33 of the struggling students, the tutor ran out of relevant problems and moved them on to the next section even though some KCs had not been mastered.

Current cognitive tutors have many steps coded with multiple KCs, for instance, in the algebra tutor some steps are coded with broad arithmetic skill categories (e.g., large vs. small numbers, rationals vs. whole numbers) in addition to the target algebraic skill. However, multiple KC coding occurs less often than it should. Doing so has often been avoided through the use of highly scaffolded interfaces, which have the downside of not assessing students in the unscaffolded context. Further, many steps that are currently coded with a single KC may be better modeled with multiple KCs [cf. Yudelson, Pavlik, and Koedinger, 2011].

5. DISCUSSION & CONCLUSIONS

We have presented an illustration of the problem of assigning blame when multiple knowledge components are required for an action and the student performs it incorrectly. A simple approach, currently used in practice, is to blame all components equally even though it may be just one (or some subset) that the student has not yet mastered. Until now, there has appeared to be little consequence to this simple approach. However, when we modified the problem selection algorithm to facilitate fading and unfading of problems with scaffolding, we found a negative consequence in the form of thrashing in problem selection. In the data from the Geometry Cognitive Tutor we found that real students were being assigned too many easy problems and not enough hard ones. Based on prior Bayesian student modeling work [Junker and Sijtsma 2001; Reye 1998; VanLehn et al. 1998], we adapted the standard knowledge tracing algorithm to create Conjunctive Knowledge Tracing (CKT), which provides a practical solution to fair blame assignment. CKT has the potential to make much better use of students' time in curricula that provide students with an adaptive learning trajectory from simple problems isolating key components of knowledge to difficult problems where multiple skills or concepts are required to produce a single response.

Alternative solutions to the blame assignment problem have been proposed [Conati, Gertner and VanLehn 2002; Pardos and Heffernan to appear; Reye 1998; VanLehn, Niu, Siler and Gertner 1998]. One simpler approach is to only blame the “hardest” KC, that is, the one with the lowest current probability. There are two potential limitations of this approach. First, if KCs are truly conjunctive and independent, such an approach will overly penalize the hardest KC and under penalize the others. We can see the difference in penalty in the KC values displayed in Row 1 of Table II (these values results from a failure on a hard problem just before this excerpt begins). Blaming only the hardest KC, which is SubGoal in this case, would yield a value of 0.49 (same as SKT would produce for this KC) whereas CKT yields a value of 0.70 (shown under Subgoal in the Conjunctive KT section). Thus, this blame-the-hardest approach could result in inappropriately requiring students to practice too many (harder) problems requiring the over-penalized KC and too few (easier) problems requiring the under-penalized KCs. A second limitation of the blame-the-hardest approach is that it does not facilitate the possibility of “unfading”, that is, of returning to scaffolded problems in the case that repeated failure on unscaffolded problems suggests (even with the softer penalty that CKT produces) the need to revisit easier problems.

Another simpler approach is to concatenate multiple KCs into a single combined KC. This approach has the downside that the student model has no information about knowledge overlap in related tasks and thus cannot be used in problem selection for the kind of gradual fading of scaffolding (going to harder problems when the student is ready) or reintroduction of scaffolding (going back to easy problems if needed) that is possible with CKT.

A more complex approach to the multiple-KC problem is to use a complete Bayesian network for the student model [e.g., Conati et al. 2002]. One immediate point of contrast with CKT is in the high effort required to engineer a student model as a Bayesian network. CKT can be relatively simply added to an existing model-tracing or constraint-based tutor as a plug-in, replacing the existing Knowledge Tracer if present. On the other hand, a full Bayesian network can represent dependencies between KCs and is not restricted to modeling KC learning only in terms of students direct experiences with those KCs. A Bayes net gives a modeler more freedom to hypothesize more complex interrelationships, like the learning of one KC enhancing another. Such freedom, however, may come at the loss of parsimony relative to the more constrained CKT approach whereby a set of KCs and a few direct computations on the KC parameter estimates may well represent all task difficulty and learning transfer relationships.

CKT is one solution within the broader space of Bayesian networks and Markov models for student modeling. As already mentioned, past work [Junker and Sijtsma 2001; Millán, Agosta and Pérez de la Cruz 2001] has articulated the multiplicative combination of noisy components. We have adapted this approach into the standard knowledge tracing by maintaining the Markov transition probability, but replacing the blame assignment with this multiplicative combination. Others have also incorporated the independence assumption and thus the multiplicative combination of components, but have put the noise (guess and/or slip parameters) at the level of the conjunction (sometimes called a “noisy-AND”) rather than at the level of the components [Conati et al. 2002]. In the psychometrics literature [Junker and Sijtsma 2001], the difference in whether the noise parameters are at the component level or the conjunction level is characterized by the contrast between the DINA (deterministic inputs noisy AND) and NIDA (noisy inputs deterministic AND) models. CKT is an extension of NIDA (adding the transition probability), with a slip and guess parameter for each conjunct in the AND. While the CKT and NIDA models have more parameters per AND relation than DINA, they can have fewer parameters in an overall student model in the case that there more AND

relations than components. For instance, there are four ($2n-n-1$) possible AND relationships of three (n) components. Whether or not these theoretical differences make any practical difference will require future empirical comparison.

Whether and when CKT provides a more or less effective user model than more complex formulations such as Bayes nets will have to await future research. Nevertheless, an important contribution of this paper is the empirical evidence that comparing such alternatives is worth it. The problem selection thrashing we observed indicates that fair blame assignment can be a real problem and better solutions may have significant impact on student users of tutoring systems. The need for such a solution comes about in situations where we want a tutoring system to make dynamic and adaptive decisions about the fading of scaffolding or the “unfading” or reintroduction of scaffolding. Such capability would seem to be an important feature of a truly adaptive tutoring system and one that can be driven by educational data mining.

ACKNOWLEDGEMENTS

Thanks for support from the Pittsburgh Science of Learning Center (NSF-SBE #0354420; see learnlab.org), assistance from Carnegie Learning Inc. (carnegielearning.com) and the DataShop team, and support from the U.S. Department of Education (IES-NCER #R305B070487) and Ronald Zdrojkowski.

REFERENCES

- CONATI, C., GERTNER, A. AND VANLEHN, K. 2002. Using Bayesian networks to manage uncertainty in student modeling. *User Modeling and User-Adapted Interaction* 12, 371-417.
- CORBETT, A.T. AND ANDERSON, J.R. 1992. Student modeling and mastery learning in a computer-based programming tutor. In *Intelligent Tutoring Systems: Second International Conference on Intelligent Tutoring Systems*, C. FRASSON, G. GAUTHIER AND G. MCCALLA Eds. Springer-Verlag, New York, 413-420.
- CORBETT, A.T. AND ANDERSON, J.R. 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction* 4, 253-278.
- GONG, Y., BECK, J. AND HEFFERNAN, N. 2010. Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting Procedures. In *Intelligent Tutoring SystemsConference Name*, V. ALEVEN, J. KAY AND J. MOSTOW Eds. Springer Berlin / Heidelberg, 35-44.
- HEFFERNAN, N. AND KOEDINGER, K. 1997. The composition effect in symbolizing: The role of symbol production vs. text comprehension. In *Proceedings of the Nineteenth Annual Conference of the Cognitive Science Society*, P. LANGLEY AND M.G. SHAFTO Eds. Lawrence Erlbaum Associates, 307-312.
- JUNKER, B.W. AND SIJTSMA, K. 2001. Nonparametric Item Response Theory in Action: An Overview of the Special Issue. *Applied Psychological Measurement* 25, 211-220.
- MILLÁN, E., AGOSTA, J.M. AND PÉREZ DE LA CRUZ, J.L. 2001. Bayesian student modeling and the problem of parameter specification. *British Journal of Educational Technology* 32, 171-181.
- PARDOS, Z.A. AND HEFFERNAN, N. to appear. Using HMMs and bagged decision trees to leverage rich features of user and skill from an intelligent tutoring system dataset. *Journal of Machine Learning Research - Special Issue on the 2010 KDD Cup Competition*.
- PARDOS, Z.A., HEFFERNAN, N.T. AND RUIZ, C. 2008. Effective Skill Assessment Using Expectation Maximization in a Multi Network Temporal Bayesian Network. In *Proceedings of the The Young Researchers Track at the 9th International Conference on Intelligent Tutoring Systems*.
- REYE, J. 1998. Two-Phase Updating of Student Models Based on Dynamic Belief Networks. In *Intelligent Tutoring Systems*, B. GOETTL, H. HALFF, C. REDFIELD AND V. SHUTE Eds. Springer Berlin/ Heidelberg, 274-283.
- RITTER, S., ANDERSON, J.R., KOEDINGER, K.R. AND CORBETT, A. 2007. Cognitive Tutor: Applied research in mathematics education. *Psychonomic Bulletin & Review* 14, 249-255.
- STAMPER, J. AND KOEDINGER, K.R. 2011. Human-machine student model discovery and improvement using DataShop. In *Proceedings of the 15th International Conference on AI in Education*.
- VANLEHN, K., NIU, Z., SILER, S. AND GERTNER, A. 1998. Student Modeling from Conventional Test Data: A Bayesian Approach without Priors. In *Intelligent Tutoring SystemsConference Name*, B. GOETTL, H. HALFF, C. REDFIELD AND V. SHUTE Eds. Springer Berlin / Heidelberg, 434-443.
- YUDELSON, M.V., PAVLIK JR., P. I. AND KOEDINGER, K.R. 2011. Towards Better Understanding of Transfer in Cognitive Models of Practice. In *Proceedings of the Fourth International Conference on Educational Data Mining*.

Less Is More: Improving the Speed and Prediction Power of Knowledge Tracing by Using Less Data

BAHADOR NOORAEI

ZACHARY A. PARDOS

NEIL T. HEFFERNAN

RYAN S.J.D. BAKER

Worcester Polytechnic Institute, USA

Knowledge Tracing is perhaps the most widely used student model in the field of educational data mining. In this paper we report on the effects of using only a subset of data in training the Bayesian Network that represents this student model. The standard practice is to use all of the students' data for a given skill to fit the model. We analyze two datasets; one from the Algebra Cognitive tutor and the other from the Genetics Cognitive tutor. We found that in both datasets, the difference in accuracy between using all the students' data versus only the most recent 15 data points of each student was not significantly different. Using only 15 responses however, resulted in an EM training time which was 15 times faster than using all data. This result suggests that the Knowledge Tracing model needs only a small range of data in order to learn reliable parameters. The implications of this result is a substantial savings in model training time that allows for more complex models to be fit or individualized models to be trained online.

Keywords and Phrases: Knowledge Tracing, Expectation Maximization, Prediction, Cognitive Tutor, Data filtering

1. INTRODUCTION

Knowledge Tracing (KT) [Corbett & Anderson 1995] is perhaps the most widely used student model in the field of educational data mining and has been used in many cognitive tutors [Koedinger, Anderson, Hadley & Mark 1997]. The standard practice is to use all of the students' data for a given skill to fit the model; and a model trained for each skill in the system.

In [Ritter, Harris, Nixon, Dickison, Murray & Towle 2009] it is discussed that reducing the parameter space of KT by means of clustering gives us models of student performance that are as good as the standard approach that gives us a different fit for each skill. So instead of 9600 parameters for the 2400 skills in the dataset, each fit differently, they settle on a set of 92 parameters, without changing the behavior of the system. In a similar vein, we aim to reduce the Knowledge Tracing training time by reducing the training data while retaining predictive performance.

In this paper we explore another question: how sensitive is the KT model to the amount of data used in its training. We train the model with different limits imposed on the maximum length of interaction instance sequences that is allowed for each student, and see their effect on prediction power of the system. To our knowledge this work is the first

to explore using less data to do better when training a student model. As it is later shown, limiting the amount of data can reduce the training time of KT model using Expectation Maximization (EM) substantially. We analyze two datasets; one from the Algebra Cognitive tutor and the other from the Genetics Cognitive tutor.

1.1 KNOWLEDGE TRACING

Corbett & Anderson's Bayesian Knowledge Tracing model is one of the most popular methods for estimating students' knowledge. It underlies the Cognitive Mastery Learning algorithm used in Cognitive Tutors for Algebra, Geometry, Genetics, and other domains [Koedinger & Corbett 2006].

The canonical Bayesian Knowledge Tracing (BKT) model assumes a two-state learning model: for each skill/knowledge component the student is either in the learned state or the unlearned state. At each opportunity to apply that skill, regardless of their performance, the student may make the transition from the unlearned to the learned state with *learning* probability $P(T)$. The probability of a student going from the learned state to the unlearned state (i.e. forgetting a skill) is fixed at zero. A student who knows a skill can either give a correct performance, or *slip* and give an incorrect answer with probability $P(S)$. Similarly, a student who does not know the skill may *guess* the correct response with probability $P(G)$. The model has another parameter, $P(L_0)$, which is the probability of a student knowing the skill from the start. After each opportunity to apply the rule, the system updates its estimate of student's knowledge state, $P(L_n)$, using the evidence from the current action's correctness and the probability of learning:

$$P(L_{n-1}|Correct_n) = \frac{P(L_{n-1}) * (1 - P(S))}{P(L_{n-1}) * (1 - P(S)) + (1 - P(L_{n-1})) * (P(G))}$$

$$P(L_{n-1}|Incorrect_n) = \frac{P(L_{n-1}) * P(S)}{P(L_{n-1}) * P(S) + (1 - P(L_{n-1})) * (1 - P(G))}$$

$$P(L_n|Action_n) = P(L_{n-1}|Action_n) + ((1 - P(L_{n-1}|Action_n)) * P(T))$$

The four parameters of BKT, $(P(L_0), P(T), P(S)$, and $P(G)$, are learned from existing data, historically using curve-fitting [7], but more recently using expectation maximization (EM). For EM the parameters were unbounded and initial parameters were set to a $P(G)$ of 0.14, $P(S)$ of 0.09, $P(L_0)$ of 0.50, and $P(T)$ of 0.14. These initial values were the average parameter values across all skills in prior modeling work conducted on a different algebra tutor [Pardos, Heffernan, Ruiz and Beck, 2008].

2. DATASETS

2.1 KDD DATASET (BRIDGE TO ALGEBRA)

This dataset comes from the Carnegie Learning Bridge to Algebra Tutor, which is an Intelligent Tutoring System (ITS) used by many students over the course of the 2007-2008 school year. This was the dataset that was one of the KDD 2010's "development" datasets [Pardos & Heffernan, In Press].

This dataset contains 1323 unit-skills (from now on, we call each unit-skill in this dataset simply a skill), and 1,817,476 data points (student actions). In order to demonstrate the effects of using less data, we limited our experiment only to those skills that have a

median of student response sequence of 40 or more. So we ended up with 33 skills with 663,491 data points (36% of all data points in the original dataset).

In this paper we refer to this dataset as KDD dataset.

2.2 GENETICS 2009 DATASET

This data was taken from a Cognitive Tutor for Genetics [Corbett, et al. 2010]. The dataset contains the results of in-tutor performance data of 76 students on 9 different skills, with data from a total of 11,581 student actions.

Six of the skills included in this dataset have average interaction sequence lengths of around 10. The 3 remaining skills have 20, 30, and 41 as average length of interaction sequence. But these 3 larger skills cover 60% of data points in the dataset.

For the students in this dataset, we also have the results of a problem solving post-test, covering some of the skills that were exercised in the tutor. Having this data, we could correlate student knowledge estimates of different configurations with the post-test data, as it is discussed in section 4.3.

3. METHODOLOGY

In our work, we group the dataset based on *skills* and fit a separate set of KT parameters for each skill. So an *instance* of interaction with the tutor in the dataset is a specific user's performance record when encountered with problems that exercise a specific skill. Here's where the idea using less data comes into the picture: should we use the full length of data in each instance, or should we limit the length of the data we feed into the EM? In order to explore the effects of imposing this limitation on the prediction-ability of KT model, we ran our experiments with different limitations on the number of data points used: from using all the data in each interaction sequence to using only the *most recent 5 items*.

For example, suppose we have two student with interaction instance sequences of $A = 0110011101111$ and $B = 1010110$ for a skill in the dataset (0 here denotes an incorrect response (wrong answer or request for help), and 1 denotes a correct response). Now if we limit the interaction sequence length to 5, the following two sequences are presented to the EM as interaction instance data related to the skill: $L_5(A) = 01111$ and $L_3(B) = 10110$. But if the limit is set at 10, we will have the following sequences: $L_{10}(A) = 0011101111$ and $L_{10}(B) = 1010110$. Notice that in the second case, whole sequence of B is used (length = 7).

For KDD dataset we tried fitting parameters with these interaction sequence length limits: 5, 10, 15, 20, 25, 35, 40, 75, 100, 150, 200, 400, and no-limit. The maximum interaction sequence in this dataset was 679, but as it is shown in the results section, not so many instances of interactions with that length are present in the dataset.

For Genetics dataset, the range of limits tried for this study is shorter because this dataset is considerably smaller and the lengthiest interaction sequence contains 88 data points. The limits we've tried for this dataset are: 5, 10, 15, 20, 25, 30, 40, 50, 60, and no-limit.

3.1 TRAINING AND TESTING OF THE KNOWLEDGE TRACING MODELS

For both datasets, we used trained KT models with different levels of student interaction data cut-off. Then all those models are used to predict student actions with cross-validation. For this prediction (or *trace*) phase, we used two different version of the

model: one that even limits the input to the *trained model* when it demands a prediction from it, and one that feeds the whole available history of student performance to the model, regardless of the limitation imposed at training stage.

Then we calculated the Root Mean Square Errors (RMSEs) at the student level and averaged them to get the number that is reported here. This way we can be sure that the reported accuracy is not biased towards students who have more those points for that skill. It also enables us to calculate a measure of statistical significance for the results.

We use Kevin Murphy's Bayes Net Toolbox for Matlab¹ for this experiment. 5-fold cross-validation was used for the Genetics dataset and a 2-fold for KDD to evaluate KT prediction performance. We used 2-fold cross-validation for KDD dataset solely to reduce the total amount of time needed to run all the experiments on that large dataset. The folds were created by randomly assigning students (and their associated responses) to folds. In each run of the cross-validation, one fold served as the test set and the other folds served as a training set.

4. RESULTS

4.1 KDD DATASET RESULTS

Root Mean Square Errors (RMSE) of cross-validated prediction of in-tutor performance of students are shown in figure 1, as the red trend line, and table 1. The x-axis in the graph (figure 1) shows the number of data points across all skills included in the EM training. As it is evident in the graph, increasing the amount of the data in training does not contribute to the model's accuracy, past a certain point (around a max of 75 responses per student). The prediction difference between five and 75 data points per student is not significant; however, the increase in runtime is substantial, shown by the blue line.

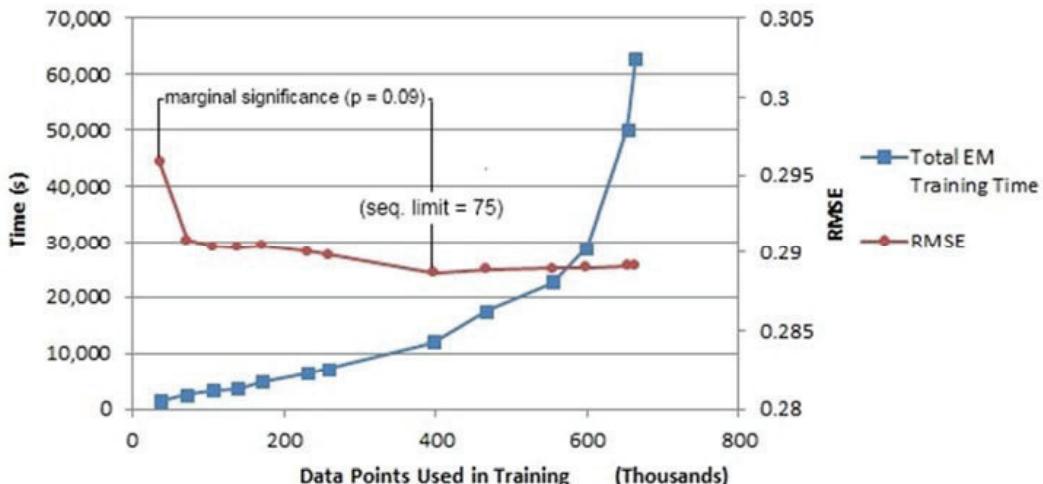


Figure 1. KDD In-Tutor Prediction Results

Figure 1 also shows the time it takes for EM to fit parameters for different amounts of data. It shows the potential exponential time complexity of the BN toolbox EM algorithm. We speculate that the change from linear to exponential time increase may have been attributed to the dataset exceeding the machine's 8 GIG memory capacity and disk swapping occurring.

¹ <http://code.google.com/p/bnt/>

Table I KDD In-Tutor Prediction Results (sorted by RMSE)

Sequence Limit	data points	RMSE
75	397,921	0.288648
100	467,471	0.288871
150	554,999	0.288948
200	598,774	0.288977
400	653,664	0.289098
No limit	663,491	0.289123
40	257,395	0.289811
35	230,149	0.290057
20	138,571	0.290328
15	105,442	0.290411
25	170,511	0.290428
10	71,276	0.290854
5	36,066	0.295847

When we put a maximum sequence limit of 10 on the training data, the trained model only became 0.6% less accurate than the fully trained one. The best accuracy was achieved by training with a limit of 75 on the each student's sequence length (represented 60% of all data points). Note that this was more accurate than using all the data yet it takes one-fifth of the time required to run the full training.

As mentioned in section 2, the KT model is a Bayesian Network with four parameters. So all differences in prediction ability of models, or lack thereof, is a consequence of the four parameters that is fit by EM. Figure 2 shows a graph of these four parameters as they are fit by using different amounts of data. The parameter values are an average of the parameter values in the 33 different KT skill models. The most dramatic change occurs in the prior parameter, which decreases monotonically. One explanation for the decrease in prior with longer sequences is that the longer the sequence, the more likely the data is produced by a student who is off task. Since students stop answering questions of a given skill when they master it, the students still answering questions after 75 opportunities are likely low achieving students with a low prior.

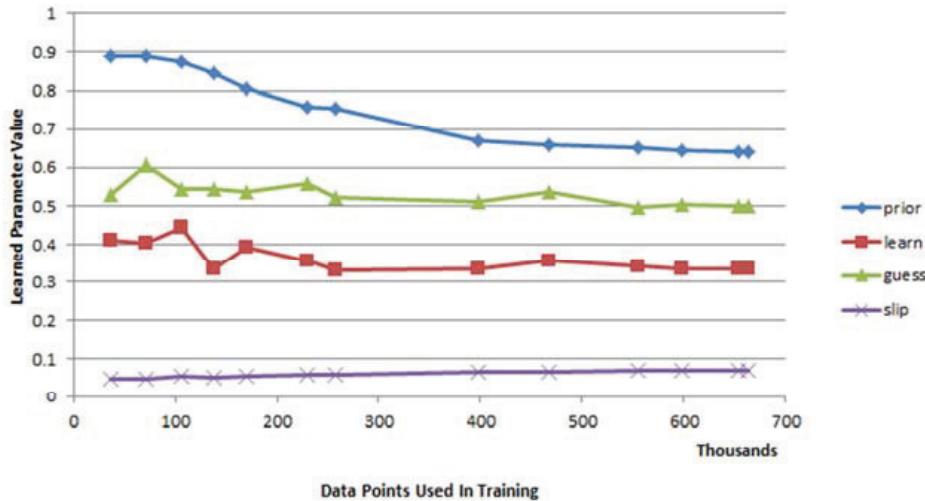


Figure 2 Average learned parameters with varying amounts of data

The slip rate, however, was extremely stable; remaining around 0.08 for almost any amount of data. One interesting implication of this, at least when fitting models to Cognitive Tutor data, is that KT could be reduced to a three parameter model by learning the slip with very little data and then fixing that parameter to the value learned while the other three parameters are trained on more data.

4.2 GENETICS DATASET RESULTS

In the case of the genetics dataset, we are dealing with much less data than the KDD (Bridge to Algebra) dataset. Figure 3 shows that error does decrease steadily with more data, however, the decrease is very small and none of the errors are statistically significant. However, while the RMSE axis is zoomed to a scale that demonstrates the small change in error (the errors fall between 0.31 and 0.32 RMSE), the time axis (on the left) ranges between 10 minutes with 5 data point cut-off and 100 minutes with full data. This is a 10x training time increase to achieve no significant increase in prediction.

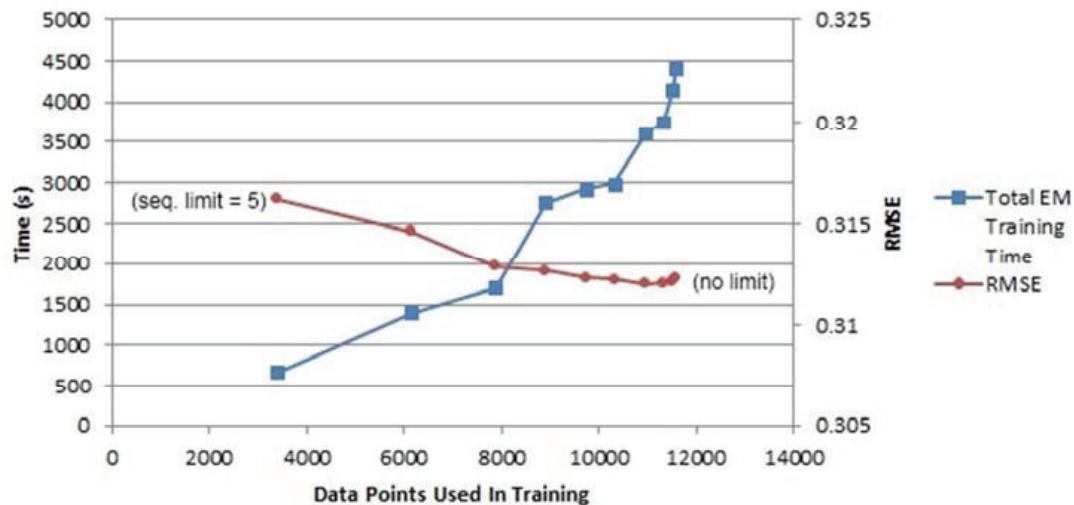


Figure 3 Genetics In-Tutor Prediction Results

In other words, limiting response sequence lengths to 5 (denoted by the first notch in the trend lines), which results in using only 29% of data points available, does not affect the prediction ability of the model at all. The best accuracy for in-tutor prediction is attained when using a sequence limit of 40, which includes 95% of data points; this is an increase in average RMSE of 0.00393 or 1% as shown in Table II.

Table II Genetics In-Tutor Prediction Errors (sorted by RMSE)

Sequence Length Limit	Number of Data Points Included	RMSE
40	10959	0.31198
50	11336	0.31203
60	11506	0.31206
30	10322	0.31223
No Limit	11581	0.31228
25	9734	0.31230
20	8898	0.31263
15	7875	0.31287
10	6156	0.31462
5	3386	0.31621

Figure 4 shows the average KT learned parameters for the Genetics dataset. A similar trend can be observed here as in the Cognitive Tutor dataset. The prior drops with more data and the slip remains nearly constant throughout. Unlike the Cognitive tutor, the guess rate decreases and the learn rate increases with more data. More investigation is necessary to explain these trends.

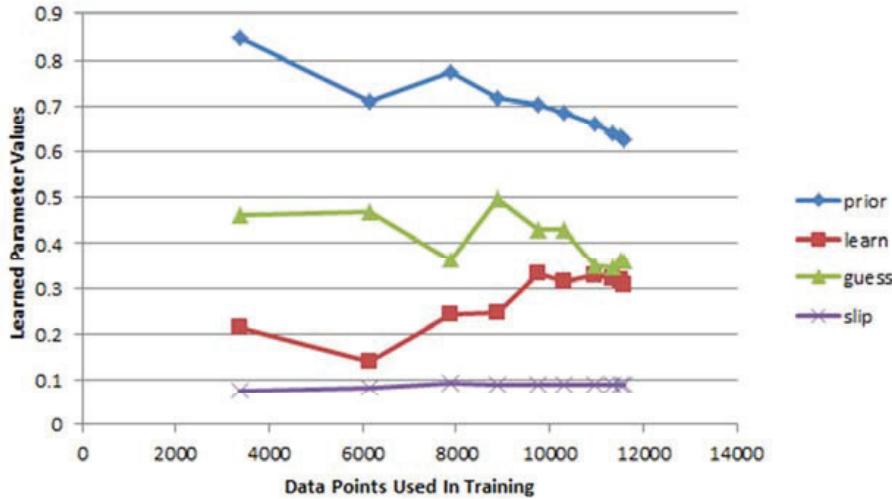


Figure 4 Average of Learned Parameters for Genetics Dataset

4.3 PREDICTING POST-TEST RESULTS FOR GENETICS DATASET

In predicting the post-test, we account for the number of times each skill will be utilized on the test. Of the nine skills in the dataset, one is not exercised on the test, and is eliminated from the model predicting the post-test. Of the remaining seven skills, four are exercised once, two are exercised twice and one is exercised three times, in each of the two posttest problems. These first two skills are each counted twice and the latter skill three times in our attempts to predict the post-test. We use Pearson's correlation as the goodness metric since the model estimates and the post-test scores are both numerical. Correlation between each model and the post-test is given in table III.

The best correlation happens when we use a sequence limit of 20 in training (77% of data). The fact that using less data gives us better predictions for the Genetics Tutor students post-test was mentioned in a recent work focusing on ensemble methods by the same authors [Baker, Pardos, Gowda, Nooraei, Heffernan In Press].

In all our experiments at predicting post-test we tried limiting the data in the tracing step as well: when using the trained model to predict student performance (we call this action *tracing*) we limited the amount past information we feed to the Bayesian Network. In other words, the same limit was imposed in tracing phase too. The results were no different from the normal *full* trace, so we eliminated any mention of them in this paper. But here, when predicting post-test results related to genetics dataset, we see an interesting phenomenon that a trace limited to only 5 most recent student data, yields a much better prediction of post-test results (table III and figure 5).

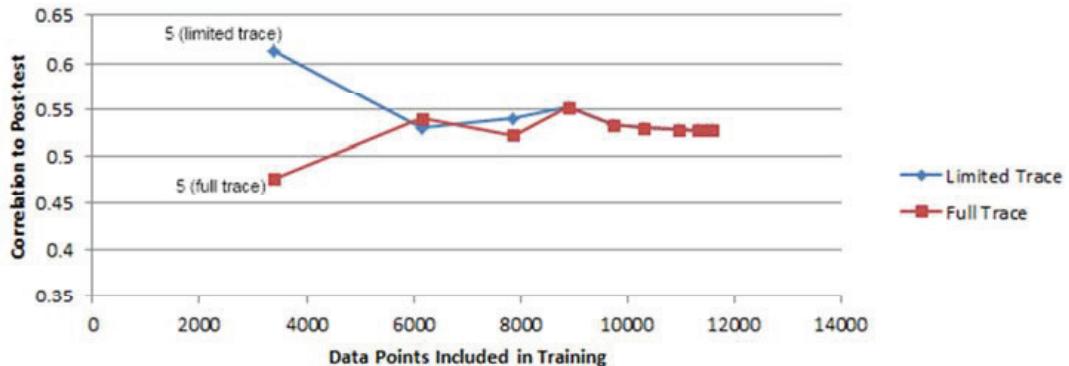


Figure 5 Genetics post-test correlation

Table III Genetics Post-Test Prediction Correlations

Sequence Length Limit	Data Points	Correlation with Post-test
5 (limited trace)	3386	0.61356
20	8898	0.55217
10	6156	0.54004
25	9734	0.53303
30	10322	0.52965
40	10959	0.52793
60	11506	0.52766
50	11336	0.52762
No limit	11581	0.52758
15	7875	0.52207
5 (normal trace)	3386	0.4761

5. CONCLUSION AND FUTURE WORKS

There are many practical reasons why one might be interested in decreasing the time it takes to fit/refit a model. In previous research [Pardos & Heffernan In Press], when we wanted to work with different variations of KT, long EM training runs were a huge impediment to rapid research cycles, so it motivated us to explore more in this area. In this paper we showed that fitting KT using EM with only a small subset of data gives us a model practically the same as a model fit with the whole available data. We also show that using only the most recent 5 data points to trace on provided the best correlation to post-test. This suggests that student's past history can be severely discounted when predicting their future performance. Tractability of individualized student models have been limited in part by the resources and time required to fit models. With our result that a good fit model can be achieved with very few data points, individualized models trained on the client can now be considered.

Our findings were largely unexpected; using 10% of the data in the case of the KDD dataset and 29% of the data in the case of the Genetics dataset lead to the same predictive power as using all the data. Given these results, ITS administrators can more wisely train their models, knowing the potential low benefit and high cost of using a student's entire response sequence to train their models. Researchers interested in predicting post-test measures from tutor data should also benefit from this finding that severely discounting the past can not only save model training time but also produce improved prediction results.

ACKNOWLEDGMENT

This research was supported by the National Science foundation via grant “Graduates in K-12 Education” (GK-12) Fellowship, award number DGE0742503 and the Department of Education IES Math center for Mathematics and Cognition grant. We would like to thank the Pittsburg Science of Learning Center for the Cognitive Tutor KDD dataset and Sujith Gowda for Genetics dataset preparation help. We would also like to thank Joseph Beck for his advice early on in the research.

We also acknowledge the many additional funders of ASSISTments Platform found here: <http://www.webcitation.org/5ym157Yfr>

REFERENCES

- BAKER, R.S.J.D., PARDOS, Z.A., GOWDA, S.M., NOORAEI, B.B., HEFFERNAN, N.T. 2011 (in press.) Ensembling Predictions of Student Knowledge within Intelligent Tutoring Systems. To appear in *Proceedings of the 19th International Conference on User Modeling, Adaptation, and Personalization*.
- CORBETT, A.T., ANDERSON, J.R. 1995. Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction*, 4, 253-278.
- CORBETT, A., KAUFFMAN, L., MACLAREN, B., WAGNER, A., JONES, E. 2010. A Cognitive Tutor for Genetics Problem Solving: Learning Gains and Student Modeling. *Journal of Educational Computing Research*, 42, 219-239.
- 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.
- KOEDINGER, K. R., CORBETT, A. T. 2006. Cognitive tutors: Technology bringing learning science to the classroom. In K. Sawyer (Ed.), *The Cambridge handbook of the learning sciences* (pp. 61-78). New York: Cambridge University Press.
- PARDOS, Z.A., HEFFERNAN, N. T. 2001 (in press.) Using HMMs and bagged decision trees to leverage rich features of user and skill from an intelligent tutoring system dataset. To appear in the *Journal of Machine Learning Research* W & CP, In Press
- PARDOS, Z. A., HEFFERNAN, N. T., RUIZ, C. & BECK, J.E. 2008. Effective Skill Assessment Using Expectation Maximization in a Multi Network Temporal Bayesian Network. The Young Researchers Track at the 20th International Conference on Intelligent Tutoring Systems. Montreal, Canada.
- RITTER, S., HARRIS, T.K., NIXON, T., DICKISON, D., MURRAY, R.C., AND TOWLE, B. 2009. Reducing the Knowledge Tracing Space. In *Proceedings of EDM 2009*, 151-160.

Analysing frequent sequential patterns of collaborative learning activity around an interactive tabletop

R. MARTINEZ, K. YACEF, J. KAY,

School of Information Technologies, University of Sydney, Australia

A. AL-QARAGHULI,

School of Computing Science, Newcastle University, UK and

Faculty of Information Science and Technology, UKM, Malaysia

AND

A. KHARRUFA

School of Computing Science, Newcastle University, UK

Electronic traces of activity have the potential to be an invaluable source to understand the strategies followed by groups of learners working collaboratively around a tabletop. However, in tabletop and other co-located learning settings, high amounts of unconstrained actions can be performed by different students simultaneously. This paper introduces a data mining approach that exploits the log traces of a problem-solving tabletop application to extract patterns of activity in order to shed light on the strategies followed by groups of learners. The objective of the data mining task is to discover which frequent sequences of actions differentiate high achieving from low achieving groups. An important challenge is to interpret the raw log traces, taking the user identification into account, and pre-process this data to make it suitable for mining and discovering meaningful patterns of interaction. We explore two methods for mining sequential patterns. We compare these two methods by evaluating the information that they each discover about the strategies followed by the high and low achieving groups. Our key contributions include the design of an approach to find frequent sequential patterns from multiuser co-located settings, the evaluation of the two methods, and the analysis of the results obtained from the sequential pattern mining.

Keywords and Phrases: Collaborative Learning, Sequence Mining, Hierarchical Clustering, Interactive Tabletops

1. INTRODUCTION

Recently, the need to explore, share and manipulate tangible data, *in situ*, has brought forth the development of new user interfaces offering large display areas and multiple input capabilities. These groupware interfaces are becoming available for educational purposes in the form of whiteboards, multi-display settings and horizontal tabletops. Interactive tabletops offer the potential for new ways to support collaborative learning activities by enabling face to face interactions between students and, at the same time, providing a great opportunity to investigate groups' learning processes by capturing their physical actions. This paper reports our work in the context of Digital Mysteries [Kharrufa et al. 2010], a tabletop collaborative learning tool for the development of students' problem-solving skills. When using this tool, students have to examine the information they are provided with and formulate an answer to a posed question (the mystery). The students' cognitive processes become evident through their physical

Authors' addresses: R. Martinez, J. Kay, K. Yacef, School of Information Technologies, The University of Sydney, Australia. E-mail: {roberto,judy,kalina}@it.usyd.edu.au; A. Al-Qaraghuli, Faculty of Information Science and Technology, UKM, Malaysia. E-mail: aalqaraghuli@gmail.com. A. Kharrufa, School of Computing Science, Newcastle University, UK. E-mail: ahmed@diwan.com.

manipulation of the information on the tabletop to solve the mystery and thus observable for researchers [Leat and Nichols 2000]. However, when a class of typical size (20 to 30 students) is divided into several small groups working in parallel, it is very difficult for facilitators to keep track of the learning processes followed by all the groups and they usually end up just looking at the final results. This is a problem as it means that the higher level strategies followed by groups are lost. The work described in this paper addresses this problem by mining and analysing frequent sequences of activity and highlighting key differences between high and low achieving groups.

The use of Data Mining techniques in collaborative learning environments has proven successful in getting insights on the interactions within groups that lead to high-quality results in terms of collaboration [Anaya and Boticario 2011; D'Mello et. al. 2011], conflict resolution [Prata et al. 2009], teamwork [Perera et. al. 2009] and *correctness* of the task [Talavera and Gaudioso 2004]. However, most of these efforts have focused on studying collaboration supported by online learning systems (e.g. chat, forums, wikis, networked ITS's) rather than tackling the context of supporting small groups collaborating around shared devices, for which there is much less research [Jeong and Hmelo-Silver 2010]. In this paper we focus on the latter. We report our work on the analysis of groups' interactions with the resources at the tabletop and the exploration of two different approaches to consider the raw physical touch actions. We detail these on a technical level and then discuss the patterns resulting from each of them.

This paper is organised as follows. Next section describes other studies that have applied machine learning techniques to analyse groups' interactions. Section 3 introduces the tabletop system and dataset. Section 4 explains the data mining methods. We conclude with reflections and future work in sections 5 and 6.

2. RELATED WORK

A number of research projects have studied the collaborative learning processes applying artificial intelligence techniques; however, they have focused mostly on assisting groups in online learning activities. Talavera and Gaudioso [2004] applied clustering in e-learning data to build student profiles based on the interactions with the user interface performed by the students. Anaya and Boticario [2011] acutely described a method to classify learners according to their level of collaboration using clustering and decision trees. Prata et.al. [2009] presented an automated detector of the nature of the utterances written at a math online system in terms of collaboration focusing on the identification of conflict between peers.

Additionally, several researchers have specifically addressed the analysis of collaboration using sequential pattern extraction. Perera et. al. [2009] modelled key aspects of teamwork on groups working with an online project management system. They clustered groups and learners according to quantitative indicators of activity and also proposed the use of alphabets to represent sequential patterns of interactions that can distinguish strong from weak groups. Other techniques have also been used to mine sequential patterns from collaborative data including Hidden Markov Models [Soller and Lesgold 2007], Social Network Analysis [Casillas and Daradoumis 2009] and Process Mining [Reimann et al. 2009].

In terms of co-located collaboration, Martinez et. al. [2011a] proposed a method to discern the extent of collaboration in groups of learners solving an optimisation problem in a multi-display face-to-face setting. The authors also applied a set of techniques to derive a user model of collaboration from a co-located multi-display setting. This also proved give information about the extent of communication and collaboration of students

at the tabletop [Martinez et. al. 2011b]. The work reported in this paper is the first effort we are aware of that has made use of data mining techniques to analyse and discover patterns of interaction from data generated by a multi-user tabletop educational application.

3. THE TABLETOP TASK: DIGITAL MYSTERIES

Digital Mysteries is a collaborative learning tool for the development and assessment of students' higher level thinking skills [Kharrufa et al. 2010]. The task provided to the students is to solve a mystery with an open question in any subject such as mathematics, history, or physics. Students are given the question and a number of data slips which may hold direct clues for solving the mystery, background information, or even red-herrings. They are asked to analyse these to formulate their answer to the question. Among the main design concepts behind the original paper-based mysteries tool [Leat and Nichols 2000] is that the students' cognitive processes become evident through their physical manipulation of these data slips to solve the mystery.

Digital Mysteries divides the task of solving a mystery into *three stages* and provides a set of externalisation tools at each of these. i) For the first "*information gathering*" stage, users are provided with 20-26 data slips. Initially, these slips are displayed in a minimised pictorial form to save space at the tabletop. Consequently, users have to expand them to read the contained clues (see Figure 1, right). ii) For the second "*grouping*" stage, students are provided with a tool for creating "*named*" groups of slips and they are asked to categorise the slips into meaningful groups. Students usually create groups in support of or against a particular claim, or groups containing information related to a particular person, topic, or event. Students move to the next stage after putting all the slips into a minimum of four named groups. iii) For the third and last "*sequencing and webbing*" stage, students are asked to use a sticky tape tool to build a branched structure that reflects cause-and-effect relations and time sequences embodying the students' answer to the question. After completing this stage, students are asked to write down their answer.

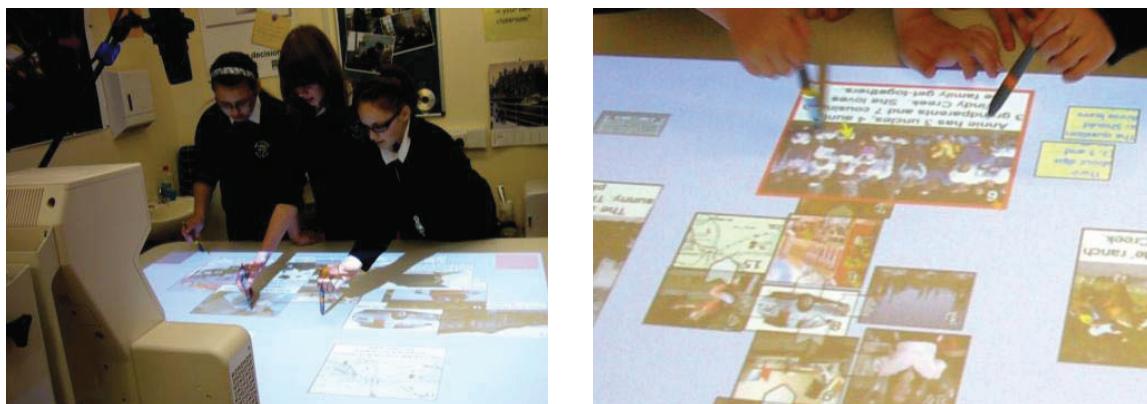


Fig. 1. Left: Three children solving a Digital Mystery. Right: Participants reading a clue

Digital Mysteries was implemented using a prototype of the multi-pen horizontal Promethean Activboard¹. Using a pen-based tabletop makes it possible to identify the author of each action. In this way, Digital Mysteries captures a rich set of interaction data throughout the mystery solution process that includes user identification or authorship as we will refer to in the rest of this paper.

¹ Promethean Interactive Whiteboards: <http://www.prometheanworld.com/>

Participants and data collection. Every action on the tabletop was logged and all sessions were video recorded. The study involved 18 participants, forming 6 groups of 3 participants each (see Figure 1, left). Some of the groups solved more than one mystery, generating a total of 12 logged sessions. Participants were elementary school students aged between 11 and 14 years. Each group was asked to find the answer to a mystery. They had to read and understand the clues, cluster them into meaningful groups, discuss which clues were related with each other and formalise a response to the mystery. Triads performed between 970 and 2017 actions per session, for a total of 17130 logged actions.

Data exploration. The raw data was coded as a series of Events, where Event= {Time, Author, Action, Object}. The possible actions that can be performed on the data slips are: moving (M), enlarging to maximum size (E), resizing to medium size (N), shrinking (S), Rotating (R), making unions with other data slips (U), add data slip to a group (G) and remove a data slip from a group (R). Out of the 12 sessions, 5 were coded as low achieving groups of students, 5 as high achieving groups and 2 as average groups. The level of achievement was coded considering: the quality of the discussions, the degree of logic thinking and the soundness of the justification for the solution of the mystery. A full report of this analysis can be found in [Kharrufa 2010]. We focus from now on the 10 groups that clearly showed evidence of low orhigh achievement.

4. MINING AND CLUSTERING SEQUENTIAL PATTERNS

From a Data Mining perspective, the dataset collected from our co-located setting poses challenges to general data mining techniques. A first challenge is that there is a diversity of spontaneous actions that can be performed when using a tabletop as opposed to online systems, such as wikis or forums, in which learners have more time to think their actions. As a result, our data might contain more non-relevant human-computer interaction events. The second challenge is the especial importance of the authorship of the low level events performed on Digital Mysteries. To address these issues we have set out to attend two research questions: i) what are the key insights that can be gained from raw and compact logged actions? (e.g. consider N similar actions as a group of actions rather than N individual actions), and ii) what information can be obtained by including authorship information in the post-processing stage of data mining?

The data mining task we set out to solve is to discover sequences of interactions between group members and the data slips at the tabletop that were more frequent in high-achieving groups than in low-achieving ones, and vice-versa. Two important attributes of our data are the sequential order and, as mentioned above, the authorship. One technique that provides insights on the timing of the events is sequential pattern mining. A sequential pattern is a consecutive or non-consecutive ordered sub-set of a sequence of events [Jiang and Hamilton 2003]. However, as noted by Perera et.al. [2009], a frequent pattern of two actions X-Y might not be meaningful if many other events or large gaps of inactivity occur between such actions. We focused on the *consecutive* ordered sub-set of events that can potentially form a pattern. We will refer to these as *frequent sub-sequence sequential patterns*. Our algorithm seeks consecutive and also repeated patterns within the dataset of sequences. A generic flow diagram of our system is shown in Figure 2 (left).

Raw dataset. Our original *raw data* consists of the events performed at the tabletop, along with the authorship information of each of these events. We present a sample excerpt from a group session log in Figure 2 (right). In Digital Mysteries each resource (data slip) provided to solve the mystery is present at the tabletop from the beginning to the end of the session. We took advantage of this to explore how learners interact with

the resources at the tabletop. We first broke down each group's long and unique sequence of events into sub-sequences of actions per data-slip. Then, to preserve meaningfulness in the patterns, we broke down these data slips' sub-sequences when a gap of inactivity longer than 120 seconds was detected.

We describe the above with a short scenario: the group decide to read a data-slip D and performs actions to enlarge it (move and enlarge actions), they read the data slip, close it and re-arrange it (more moves and shrink actions); if after this sequence there is a "group action" for the same data slip, but 5 minutes later, we can assume that the "group action" is not directly related with the previous actions. We chose a gap of 120 seconds as a maximum threshold beyond which the set of actions are considered as unrelated. This time frame was chosen based on the observations made on the videos of the sessions and the log files. In summary, the raw dataset we started with as input of step 1 is a dataset of 1618 sequences generated by breaking down the actions of each session in this order: by stage, resource (*data slip*) and long inactivity gap. The length of each sequence obtained was between 4 and 40 elements. In this dataset of sequences, each sequence is related with the session, stage and resource it comes from. Each element within each sequence contains information on timing, authorship and action type.

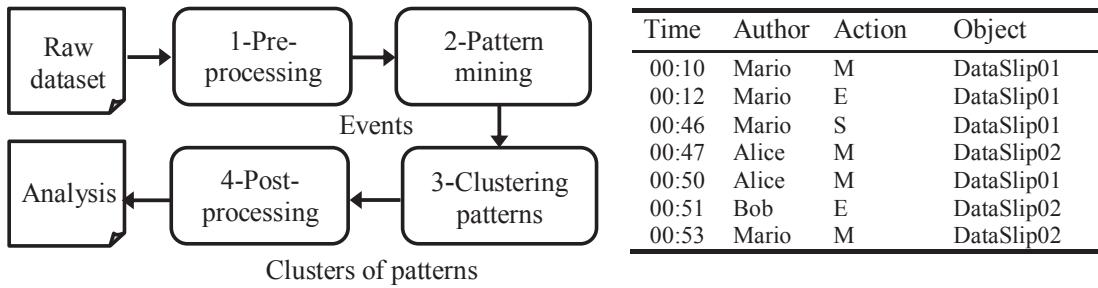


Fig. 2. Left: Steps of our data mining approach. Right: Excerpt from the application logs of activity.

Step 1. We explored two pre-processing approaches: the first method consists in going straight into the sequential mining (hence a void step 1). The second method consists in compact similar contiguous actions before applying the sequence mining. Both methods are described in detail in the next section. The output of the first step for both cases is a pre-processed *dataset of sequences*.

Step 2. The *sequence mining* step is generic for both approaches. As mentioned before, our aim is to look for frequent ordered patterns within the action sequences. With the purpose of exploiting not just the frequency but also the redundancy of the patterns we are searching for, we chose an algorithm to extract the frequent sub-sequences from sequences using n-grams [Masataki and Sgisaka 1996]. An n-gram is a subsequence of n items from a given sequence. We set the minimum support threshold to consider a pattern as frequent if this was present in at least one quarter of the total number of data slips. We also set the maximum error in 1 to allow the matching of patterns with sub-sequences if there was an edit distance of 0 (perfect match) or 1 (one different action in the sub-sequence) between them. The output of this step is a list of frequent sequential patterns that meet the minimum given support.

Step 3. The purpose of step 3 is to cluster the patterns found in step 2. Indeed, without further treatment, patterns obtained from step 2 offer limited information to differentiate groups of learners. There can also be many similar patterns. As a result, it is tedious to analyse each pattern distribution across the groups. The patterns were clustered based on their *edit distance*. The edit distance between two patterns was defined as the

minimum number of changes needed to convert one pattern of actions into the other, with the allowed operations: insertion, deletion, or substitution of a single action. We used a hierarchical agglomerative clustering technique [Witten and Frank 1999] whose input is a matrix that contains all the edit distances between each pair of patterns. We chose this technique as it has proven successful in mining human-computer interaction data [Fern et al. 2010]. The end result can be visually represented by a dendrogram, showing different levels in which patterns are clustered. These visual representations served to supervise the cluster formation and decide which level of clustering was considered as acceptable.

Step 4. Post-processing and analysis. In the post-processing stage we included the authorship information, by considering the number of students who were involved with the patterns. We also examined the benefits of each method employed at step 1, i.e. the use of raw versus compacted data.

We now describe in detail the specifications of each approach and the results of the data mining outcomes in collaborative learning terms.

4.1. Method 1: Authorship in the post processing

The first method consists in exploring the information that can be obtained by mining the Human-Computer Interaction logs of physical actions without reducing the events.

Pre-processing and sequence mining for method 1. The input data for the sequence mining consisted of a list of sequential raw sequences of events (e.g. {M-E-M-M-S-M-N-G-S-M-R} where M=move, E=enlarge to maximum size, N=resize to normal size, G=add to group, S= shrink and R=remove from group). The output was a list of frequent patterns. Only sequences of at least 4 actions were considered. The final result included 259 frequent patterns found of length varying from 4 to 10 actions.

Post-processing and clustering for method 1. Based on direct observations made on the video recorded sessions and the sequential patterns found, we obtained that many patterns had a similar meaning, although the order or quantity of actions they contained were somewhat different. For example, the sequential patterns S1={M-E-M-M-S} and S2={M-M-M-E-S-M-M} (where M=move, E=enlarge, S=shrink) are both related with the same strategy: read a data-slip, close it and re-arrange it immediately afterwards (presumably to keep the interface organised and tidy). These observations led us to use clustering to group similar patterns. In this part of the process, the input for the hierarchical clustering algorithm was a similarity matrix of 259 x 259 that contained the edit distances of all pairs of sequences. The algorithm produced a dendrogram of 4 hierarchies as output. The clusters obtained were supervised to inspect the extent in which the groups were similar. After analysing the dendrogram, the second highest level was selected to form eight meaningful clusters. This is the only part of the approach in which the results were manually supervised.

Results of method 1. We examined the results of the clustering by looking at the trends observed between patterns and groups of learners that presented a prominent level of achievement. We found that some sessions (high or low achievers) showed behaviour associated with certain clustered patters. Therefore, we used unpaired student tests ($p \leq 0.05$) to statistically analyse whether there were significant differences between such sessions. Table I summarises the clusters found using this approach and the results of such analysis.

The first two clusters are related with the strategies that learners followed to gather information from the data slips. Cluster 1 contained sequences related with the strategy of reading the slips by enlarging the object and then, after a reasonable time, closing them to keep the interface tidy. Some of these groups positioned the slips in a certain region of

the table to indicate they had already read them. On the other hand, Cluster 2 contained sequences of actions in which groups maximised the data slips without closing them. The observations on the videos indicated that some of the groups which followed this behaviour skipped the reading of some slips. We found that high achievers favoured the strategy of *reading, minimising and arranging immediately* (*cluster 1 mean = 124.75, cluster 2 mean = 61.25*). On the contrary, low achievers used both strategies for the information gathering, performing more actions contained in Cluster 2 in which they did not close the slips immediately after reading (*cluster 1 mean = 104.40, cluster 2 mean = 114.80*). This simple change in the strategy for collecting information suggests that reading without re-arranging increases clutter, making the task more difficult to be controlled by the group. Indeed, cluster 3, which contains patterns related with making space actions (moving and shrinking), showed a strong link with low achieving groups ($t=2.47, p= 0.039$). As a result, low achievers spent much more time than the high achievers arranging the elements at the table.

Clusters 6, 7 and 8 contain “union” actions in which learners established links between the data slips they considered to be tightly related. Cluster 6 includes sensible amount of union actions (at most two unions) performed along with arrangement actions. Cluster 7 presented a moderate amount of union actions and cluster 8 presented patterns with an enormous amount of union actions. Low achieving groups favoured clusters 7 and 8 ($t=2.97, p=0.018$ and $t=3.98, p=0.0041$ respectively). Based on this trend, low achievers created too many unions related to a specific data slip in short periods of time. On the contrary, high achieving groups favoured patterns with modest quantity of unions ($t=2.81, p=0.023$). Clusters 4 and 5 included patterns related with ungroup and group actions. In this case we obtained some differences among sessions. Low achievers made more “corrections” on categorising data slips than high achievers.

Table I. Results for clusters of patterns found by mining the raw events.

Cluster	Example sequence	Favoured Groups	Participants
1- Read and arrange	{M-M-E-M-S-M}	Slightly more in high achievers	Both groups 1-2 authors
2- Read slip	{M-E-M-M}	Slightly more in low achievers	Both groups 1-2 authors
3- Arrangement	{M-M-S-M-M}	Substantially more in low achievers	Low achievers 2-3 authors
4- Ungroup	{M-R-M-G}	Slightly more in low achievers	Both groups 1-2 authors
5- Group	{M-N-M-G-M-S}	Both groups	Both groups 1-2 authors
6- Few unions	{M-M-U-M-M}	Substantially more in high achievers	Low achievers 2-3 authors
7- Moderate unions	{M-U-M-U-M-U}	Substantially more in low achievers	Low achievers 2-3 authors
8- Too many unions	{U-U-U-M-U-U-U}	Substantially more in low achievers	Low achievers 2-3 authors

In regards to authorship, we analysed the way in which participants collectively interacted with the resources in terms of number of authors involved with the data slip in each pattern. For clusters 3, 6, 7 and 8 we obtained a strong statistical difference in the number of participants working together with the same data slip. Low achieving groups presented more sequences in which the *three* authors performed actions sequentially compared with high achieving groups ($p<0.05$ in all cases). For the rest of the clusters there were no significant differences in the number of authors involved with the patterns. For the clusters related with the strategies for gathering information (clusters 1 and 2) and grouping data-slips (clusters 4 and 5) the sequences were performed mostly by one author, and in some cases, by two authors in both high and low achieving groups (see Table I, column Participants).

What we learnt from these findings is that having many hands on the same object at the same time does not imply improved work. In fact, the sequences in which the low achievers have the three participants involved are mostly focused on non-cognitively demanding tasks, such as arranging the elements on the tabletop (cluster 3). In the case of

the “union actions” clusters (7 and 8) even when the activity is a cognitively demanding task, we learnt from the analysis described above and from observing the videos that lower achieving groups created a larger number of unions on particular slips that were not necessarily meaningful. Grounding on these results and the video analysis we obtained that the high level groups worked more collaboratively and participants were keener to interact on one data slip at a time, even if they worked in parallel with different objects.

We also explored the possible significant differences between the patterns and the stages in which they appear. As expected, clusters related to gathering information (clusters 1 and 2) are mainly related with stage 1, cluster 3 (re-arrangement) with all the stages, Clusters 4 and 5 with stage 2 (grouping and ungrouping actions) and the clusters related with union actions are evidently related with the third stage (sequencing and webbing). Thus, no further special consideration was put on the staging information.

4.2. Method 2: Authorship in the post processing and generalisation in the pre-processing

The second approach consists of generalising (compacting). Then, we looked at the similarities of this method outcomes with method 1 results.

Pre-processing and sequence mining for method 2. The dataset of sequences was compressed. The aim of the compression was to see how much information will be lost or gained if we generalised the user interface actions that can be attributed to user slips. A simple alphabet was applied which follows a single rule: the sequential actions of the same type (such as the action M in {M-M-M-E-M} or S in {S-U-U-U}) were compacted adding the quantifier for regular expressions + ({M+-E-M} and {S-U+}). The minimum length of the patterns was set to 3 actions, or 2 actions if at least one of the actions contained the quantifier +. In this case the minimum support was also set to one quarter of the data slips. The final result included 261 frequent patterns found of size between 3 and 5 actions each.

Post-processing for method 2. The 261 patterns were clustered following the same process used in method 1. We obtained a dendrogram with 5 levels. The first issue found in this method was that patterns were more difficult to cluster accurately as they contained less contextual information (fewer items). The solution was to choose a lower clustering level and manually merge the smaller clusters which contained similar actions. Seven meaningful clusters resulted from the clustering supervision and also 2 extra very small clusters that could not be considered into any other cluster.

Results of method 2. Even though some details in the sequences were lost, we found similar observable tendencies in the presence of patterns of high and low achieving sessions (see table II). This approach provided a deeper difference between the ways the higher and lower achieving groups gather information to solve the problem. For example we found a stronger difference in the strategy of reading without minimising the data slips performed mostly by the low achieving groups (Cluster 2, $t = 2.69$, $p=0.0272$). We also found a significant difference with respect to the strategy “read – close and arrange data slips” favouring the high achieving groups (Cluster 1, $t=3.05$, $p=0.0158$). Results also confirmed that low achieving groups performed a huge amount of unions between data slips in short periods of time (Cluster 7, $t = 3.05$, $p=0.0158$).

For the authoring information, the results from method 1 were also confirmed. The cluster that contains sequences with high amounts of union actions performed by 2 and 3 users at the same time were present mostly in the low achieving groups ($t=2.714$, $p=0.0265$). Some information is lost though; there were no significant differences

between groups in any other aspect. In general, this approach confirmed the insights obtained applying method 1 but the quality of the results decreased in some cases.

Table II. Results for clusters of patterns found mining compacted events.

Cluster	Example sequence	Favoured Groups	Participants
1- Read and arrange	{M+-N-S-M+}	Substantially more in high achievers	Both groups 1-2 authors
2- Read slip	{M-E-M+}	Substantially more in low achievers	Both groups 1-2 authors
3- Arrangement	{M+-S-M+}	Slightly more in low achievers	Both groups 1-2 authors
4- Ungroup	{M-R-M+}	Both groups	Both groups 1-2 authors
5- Group	{M+-G-S}	Both groups	Low achievers 2-3 authors
6- Few unions	{M+-U-M-U}	Slightly more in high achievers	Low achievers 2-3 authors
7- Many unions	{M-U+-M-U+-M}	Substantially more in low achievers	Low achievers 2-3 authors

5. DISCUSSION

The design of our approach was motivated by the goal of exploiting the large amounts of data generated from learners' interactions with the interactive tabletop. Our approach shows promise to follow up research on supporting collaborative learning through the use of tabletops and machine learning techniques. Our data mining approach consisted of mining both the raw human computer interactions and the compact logged actions, clustering similar frequent patterns based on edit distance and analyse the proportion of these clusters among group sessions. Both methods we explored produced similar results therefore the compacting method provides very interesting insights even when some details are lost. However, this loss of information impacted negatively on the clustering step, thus this method is unsuitable for being used for automatic support.

Method 1 also requires some human supervision to code the level of achievement of groups. Further research needs to be done on the ways to automatically extract indicators of collaboration. In regards to the educational value of the results, the video analyses confirmed the presence of serial patterns of interaction in the trials. Group members of high achieving groups tried to interact and externalise their thinking. They tended to read all the slips to get clues about the mystery and parallel interactions were clearly observed along with engagement in conversations. The results of our approach do not tell the whole story but are good indicators of desired and undesired patterns of behaviour related with strategies that are followed by groups.

The goal of this line of research is to offer adapted support to groups in the form of direct feedback to students or to their facilitator. The insights obtained in the work reported in this paper are the first steps towards such adapted support that machine learning techniques can offer to the use of tabletop devices. We addressed two questions posed at the beginning of this paper, regarding (i) the key insights that can be acquired from mining raw or compact logged actions and (ii) the information offered by the authorship element of the data logs.

We observed that the results obtained with both methods reflected similar patterns of behaviour, such as the strategies followed by the groups to gather information, arrange resources and the creation of links between data slips. Some elements of the interactions came up by compacting the redundant actions in method 2 (gathering information strategies), but in other elements some information was lost. The most important issue with the compacting method is that more empirical interpretation was needed after the clustering step whilst method 1 offered better clusters. In general, the raw HCI actions offer an adequate degree of detail to obtain meaningful results when studying the interactions *by resource*. In regards to the question related with authorship, results indicated that low achieving groups tended to work sequentially with the same objects. We confirmed from the video analysis that high achieving groups tended to discuss their thoughts and work in parallel with different objects.

6. CONCLUSION AND FUTURE DIRECTIONS

This paper presented an outline of distinctive techniques to extract elements of collaborative interaction at the tabletop. These techniques reveal the importance of the design of specific data mining methods for exploiting traces of collaboration from co-located situations. Our work grounds upon educational data mining research on online collaborative learning and we have proposed a methodology that can be used as a starting point to guide future research on the identification of patterns from educational tabletop settings. An important goal of our work is to mirror useful information about groups to help facilitators and the students themselves to reflect on and improve their learning activity. There are still a number of open questions that we want to address. The next step in this line of research will be the exploration of other ways to analyse sequential actions considering parallel work, looking at the high level problem-solving processes, designing alphabets to include authorship in earlier stages of the data mining.

REFERENCES

- ANAYA, A.R. AND BOTICARIO, J.G. 2011. Application of machine learning techniques to analyse student interactions and improve the collaboration process. *Expert Systems with Applications* 38, 1171-1181.
- CASILLAS, L. AND DARADOUMIS, T. 2009. Knowledge extraction and representation of collaborative activity through ontology based and Social Network Analysis technologies. *BI and Data Mining* 4, 141-158.
- D'MELLO, S., OLNEY, A. AND PERSON, N. 2011. Mining Collaborative Patterns in Tutorial Dialogues. *JEDM - Journal of Educational Data Mining* 2, 1-37.
- FERN, X., KOMIREDDY, C., GRIGOREANU, V. AND BURNETT, M. 2010. Mining problem-solving strategies from HCI data. *ACM Trans. Comput.-Hum. Interact.* 17, 1-22.
- JEONG, H. AND HMELO-SILVER, C.E. 2010. An Overview of CSCL Methodologies. In *Proceedings of the 9th International Conference of the Learning Sciences*, Chicago, USA2010, 920-921.
- JIANG, L. AND HAMILTON, H. 2003. Methods for Mining Frequent Sequential Patterns. In *Advances in Artificial Intelligence*, Y. XIANG AND B. CHAIB-DRAA Eds. Springer Berlin / Heidelberg, 992-992.
- KHARRUFA, A.S. 2010. Digital tabletops and collaborative learning. PhD thesis. School of Computing Science, Newcastle University, Newcastle Upon Tyne.
- KHARRUFA, A., LEAT, D. AND OLIVIER, P. 2010. Digital mysteries: designing for learning at the tabletop. In *Interactive Tabletops and Surfaces* ACM, 197-206.
- LEAT, D. AND NICHOLS, A. 2000. Brains on the Table: Diagnostic and formative assessment through observation. *Assessment in Education: Principles, Policy & Practice* 7, 103 - 121.
- MARTINEZ, R., WALLACE, J., KAY, J. AND YACEF, K. 2011a. Modelling and identifying collaborative situations in a collocated multi-display groupware setting. In *Proceedings of the International Conference on Artificial Intelligence in Education* 2011.
- MARTINEZ, R., KAY, J., WALLACE, J. AND YACEF, K. 2011b. Modelling symmetry of activity as an indicator of collocated group collaboration. In *Proceedings of the International Conference on User Modeling, Adaptation and Personalization*.
- MASATAKI, H. AND SGISAKA, Y. 1996. Variable-order N-gram generation by word-class splitting and consecutive word grouping. In *Proceedings of the Conference on Acoustics, Speech, and Signal Processing*, 19961996 IEEE Computer Society, 1256487, 188-191.
- PERERA, D., KAY, J., KOPRINSKA, I., YACEF, K. AND ZAIANE, O. 2009. Clustering and Sequential Pattern Mining of Online Collaborative Learning Data. *IEEE Tran. on Knowledge and Data Engineering* 21, 759-772.
- PRATA, D., BAKER, R., COSTA, E., ROSÉ, C., CUI, Y. AND DE CARVALHO, A. 2009. Detecting and Understanding the Impact of Cognitive and Interpersonal Conflict in Computer Supported Collaborative Learning Environments. In *2nd International Conference On Educational Data Mining*, 131-140.
- REIMANN, P., FREREJEAN, J. AND THOMPSON, K. 2009. Using process mining to identify models of group decision making in chat data. In *Proceedings of the international conference on Computer Supported Collaborative Learning* 2009 International Society of the Learning Sciences, 98-107.
- SOLLER, A. AND LESGOLD, A. 2007. Modeling the process of collaborative learning. *The Role of Technology in CSCL*, 63-86.
- TALAVERA, L. AND GAUDIOSO, E. 2004. Mining Student Data to Characterize Similar Behavior Groups in Unstructured Collaboration Spaces. In *Proceedings of the European Conf. Artificial Intelligence* 2004.
- WITTEN, I.H. AND FRANK, E. 1999. *Data mining: Practical machine learning tools and techniques with Java implementations*. Morgan Kaufmann, San Francisco.

Acquiring Item Difficulty Estimates: a Collaborative Effort of Data and Judgment

K. WAUTERS

Katholieke Universiteit Leuven, Belgium

P. DESMET

Katholieke Universiteit Leuven, Belgium

AND

W. VAN DEN NOORTGATE

Katholieke Universiteit Leuven, Belgium

The evolution from static to dynamic electronic learning environments has stimulated the research on adaptive item sequencing. A prerequisite for adaptive item sequencing, in which the difficulty of the item is constantly matched to the knowledge level of the learner is to have items with a known difficulty level. The difficulty level can be estimated by means of the item response theory (IRT), as often done prior to computerized adaptive testing. However, the requirement of this calibration method is not easily met in many practical learning situations, for instance, due to the cost of prior calibration and due to continuous generation of new learning items. The aim of this paper is to search for alternative estimation methods and to review the accuracy of these methods as compared to IRT-based calibration. Using real data, six estimation methods are compared with IRT-based calibration: proportion correct, learner feedback, expert rating, paired comparison (learner), paired comparison (expert) and the Elo rating system. Results indicate that proportion correct has the strongest relation with IRT-based difficulty estimates, followed by learner feedback, the Elo rating system, expert rating and finally paired comparison.

Key Words and Phrases: IRT, proportion correct, learner feedback, expert rating, paired comparison, graded response model and Elo rating

1. INTRODUCTION

Most e-learning environments are static, in the sense that they provide for each learner the same information in the same structure using the same interface. One of the recent tendencies is that they become dynamic or adaptive. An adaptive learning environment creates a personalized learning opportunity by incorporating one or more adaptation techniques to meet the learners' needs and preferences (Brusilovsky 1999). One of those adaptation techniques is adaptive curriculum/item sequencing, in which the sequencing of the learning material is adapted to learner-, item-, and/or context characteristics (Wauters, Desmet & Van den Noortgate 2010). Hence, adaptive item sequencing can be established by matching the difficulty of the item to the proficiency level of the learner. Recently, the interest in adaptive item sequencing has grown, as it is found that excessively difficult items can frustrate learners, while excessively easy items can cause learners to lack any sense of challenge (e.g. Pérez-Marín, Alfonseca & Rodriguez 2006, Leung & Li 2007). Learners prefer learning environments where the item selection procedure is adapted to their proficiency, a feature which is already present to a certain extent in computerized adaptive tests (CATS; Wainer 2000).

A prerequisite for adaptive item sequencing is to have items with a known difficulty level. Therefore, an initial development of an item bank with items of which the difficulty level is known is needed. This item bank should be large enough to include at any time an item with a difficulty level within the optimal range that has not yet been presented to the learner. In CAT, the item response theory (IRT; Van der Linden & Hambleton 1997) is often used to

Authors' addresses: K. Wauters, ITEC/IBBT, Faculty of Psychology and Educational Sciences, Katholieke Universiteit Leuven, Kortrijk, Belgium. E-mail: kelly.wauters@kuleuven-kortrijk.be; P. Desmet, ITEC/IBBT, Faculty of Arts, Katholieke Universiteit Leuven, Kortrijk, Belgium. E-mail: pietdesmet@kuleuven-kortrijk.be; W. Van den Noortgate, ITEC/IBBT, Faculty of Psychology and Educational Sciences, Katholieke Universiteit Leuven, Kortrijk, Belgium. E-mail: wim.vandennoortgate@kuleuven-kortrijk.be

generate such a calibrated item bank. IRT is a psychometric approach that emphasizes the fact that the probability of a discrete outcome, such as the correctness of a response to an item, is function of qualities of the item and qualities of the person. Various IRT models exist, differing in degree of complexity, with the simplest IRT model stating that a person's response to an item depends on the person's proficiency level and the item's difficulty level. More complex IRT models include additional parameters, such as an item discrimination parameter and a guessing parameter. Obtaining a calibrated item bank with reliable item difficulty estimates by means of IRT requires administering the items to a large sample of persons in a non-adaptive manner. The sample size recommended in the literature varies between 50 and 1000 persons (e.g. Kim 2006, Linacre 1994, Tsutakawa & Johnson 1990). Because IRT has been a prevalent CAT approach for decades, it seems logical to apply IRT for adaptive item sequencing in learning environments that consist of simple items. However, the difference in data gathering procedure of learning and testing environments has implications for IRT application in learning environments. In many learning environments, the learners are free to select the item they want to make. This combined with the possibly vast amount of items provided within the learning environment leads to the finding that many exercises are only made by few learners (Wauters et al. 2010). Even though IRT can deal with structural incomplete datasets (Eggen 1993), the structure and huge amount of missing values found in the tracking and logging data of learning environments can easily lead to non-converging estimations of the IRT model parameters. In addition to this, the maximum likelihood estimation procedure implemented in IRT has the disadvantage of being computationally demanding.

Due to these impediments that go together with IRT based calibration, we are compelled to search for alternative estimation methods to estimate the difficulty level of items. Some researchers have brought up alternative estimation methods. However, the accuracy of some solutions were not compared to IRT based calibration and the various methods were not compared in a single setting. The purpose of this study is to review the accuracy of some alternative estimation methods as compared to IRT-based calibration in a single setting.

2. EXPERIMENT

2.1 Related Work

2.1.1 Item Response Theory. To estimate the item difficulty, the IRT model with a single item parameter proposed by Rasch (Van der Linden & Hambleton 1997) is used. The Rasch model models the probability of answering an item correctly as a logistic function of the difference between the person's proficiency level (θ) and the item difficulty level (β), called the item characteristic function:

$$\frac{\exp(\theta_p - \beta_i)}{1 + \exp(\theta_p - \beta_i)}$$

The IRT-based estimation of the difficulty level will be estimated on the basis of the learners' data obtained in this study. In addition to that, IRT-based calibration conducted on preliminary examinee data by Selor, the selection agency of the Belgian government, serves as true difficulty parameter values.

2.1.2 Proportion Correct. A simple approach to estimate the difficulty level of items is to calculate the proportion of correct answers by dividing the number of learners who have answered the item correctly by the number of learners who have answered the item. To obtain the item difficulty parameter, the proportion correct scores has to be converted as follows:

$$\beta_i = \log \left[\frac{1 - \frac{n_i}{N_i}}{\frac{n_i}{N_i}} \right],$$

where β_i denotes the item difficulty level of item i , n_i represents the number of learners who have answered item i correctly, and N_i represents the number of learners who have answered item i .

The advantage of this approach is that the item difficulty can be calculated online due to the easy formula which does not require many computational resources. Furthermore, the item difficulty can be updated after each administration. The lower the proportion of students who have answered the item correctly, the more difficult the item is. Johns, Mahadevan and Woolf (2006) have compared the item difficulty level obtained by IRT estimation with the percentage of students who have answered the item incorrectly, and found a high correlation ($r=0.68$).

2.1.3 Learner Feedback. Some researchers have applied learner's feedback in order to provide adaptive sequencing of courseware in e-learning environments (e.g. Chen, Lee & Chen 2005, Chen, Liu & Chang 2006, Chen & Duh 2008). After a learner has studied a particular course material, he is asked to answer two simple questions: "Do you understand the content of the recommended course material?" and "How do you think about the difficulty of the course materials?". After a learner has given feedback on a 5-point Likert scale, scores are aggregated with those

of other learners who previously answered this question by taking the average of the scores. The new difficulty level of the course material is based on a weighted linear combination of the course difficulty as defined by course experts and the course difficulty determined from collaborative feedback of the learners. The difficulty parameters slowly approach a steady value as the number of learners increases.

In this study the procedure of Chen et al. (2005) for adjusting the difficulties of the items is slightly altered. The course difficulty as defined by course experts is not taken into account. Instead, the difficulty estimates are solely based on the collaborative feedback of the learners. After an item is presented, the learner is asked a feedback question “How difficult did you find the presented item?”. The learner answers on a 5-point Likert scale (Likert, 1932), ranging from -2 (“very easy”) over -1 (“easy”), 0 (“moderate”), 1 (“difficult”) to 2 (“very difficult”). The item difficulty based on learner feedback is then given by the arithmetic mean of the scores.

2.1.4 Paired Comparison. Another method, already used in CAT, to estimate the difficulty level of new items is paired comparison (Ozaki & Toyoda 2006, 2009). In order to prevent content leaking, experts are asked to assess the difficulty of items through one-to-one comparison or one-to-many comparison. In this method, items for which the difficulty parameter has to be estimated, are compared with multiple items, of which the item difficulty parameter is known. The underlying thought that prompts this item difficulty estimation approach is Thurstone’s paired comparison model. While Thurstone (1994) modelled the preference judgment for object i over object j , Ozaki and Toyoda (2006, 2009) modelled the difficulty judgment of item i over item j .

In this study a similar procedure of the one employed by Ozaki and Toyoda (2009) is adopted to estimate the difficulty level by means of paired comparison. After an item is presented, the learner has to judge where the presented item should be located in a series of 11 items ordered by difficulty level from easy to difficult. This means that the raters have to make a one-to-many comparison with 11 items of which the item difficulty parameter is known. The probability that item i is more difficult than item 1, according to N raters is expressed as:

$$P_1(\beta_i) = \frac{1}{1 + \exp[-1(\beta_i - b_1)]},$$

Where β_i is the difficulty of item i judged by the raters, b_1 is the difficulty parameter of item 1 as estimated by the preliminary IRT analysis, conducted by Selor.

In this study 11 items are presented simultaneously and the raters have to select one out of 12 categories: $i < 1$, $1 < i < 2, \dots, 10 < i < 11, 11 < i$. Because the 11 items are ordered according to their difficulty level from easy to difficult, the idea of the graded response model (Samejima, 1969) can be adopted to extract the boundary response function of each category as:

$$\begin{aligned} P_{i < 1}(\beta_i) &= 1 - P_1(\beta_i) \\ P_{1 < i < 2}(\beta_i) &= P_1(\beta_i) - P_2(\beta_i) \\ \dots \\ P_{10 < i < 11}(\beta_i) &= P_{10}(\beta_i) - P_{11}(\beta_i) \\ P_{11 < i}(\beta_i) &= P_{11}(\beta_i) \end{aligned}$$

The final estimation of β_i is obtained by maximizing the log likelihood, while fixing b_j s, i.e. the difficulty parameters of item 1 to 11 as estimated by the preliminary IRT analysis.

2.1.5 Expert Rating. Another approach to obtain item parameter estimates is allowing subject domain experts to estimate the value of the difficulty parameter (Yao 1991, Linacre 2000, Fernandez 2003, Lu, Li, Liu, Yang, Tan & He 2007). There is some evidence in the measurement literature that test specialists are capable of estimating item difficulties with reasonable accuracy (e.g., Chalifour & Powers 1989), although other studies found contradictory results (Hambleton, Bastari & Xing 1998). As indicated by Impara and Plake (1998), a distinction has to be made between the ability of experts to rank order items accurately with reference to the difficulty level, and the ability of experts to estimate the proportion of persons who will answer the items correctly. Experts seem to be capable conducting the former task, but have difficulties conducting the latter where they have to be able to conceptualize the reference group and predict how well such persons will perform on each item.

Hence, two methods for obtaining expert ratings were included in this study: a paired comparison method and an evaluation on a proportion correct metric. The formula’s to obtain the item difficulty parameter estimates based on these two methods are described in the subsections “Paired Comparison” and “Proportion Correct” respectively.

2.1.6 Elo Rating. The Elo Rating approach (Brinkhuis & Maris 2010) for estimating the item difficulty level is an educational implementation of the Elo Rating system used for rating chess performances and sports (Elo 1978). In sport, for example, two players compete with each other, resulting in a win, a loss or a draw. These data are known as paired comparison data, for which the Elo rating system is developed. In the educational field, a person is seen as a player and an item is seen as its opponent. The Elo Rating formula expresses the new rating after an event as a

function of the pre-event rating, a weight given to the new observation and the difference between the actual score on the new observation and the expected score on the new observation. Brinkhuis and Maris (2010) estimated the expected score on the new observation by means of the Rasch model. The formula implies that when the difference between the expected score and the observed score is high, the change in both the person's knowledge level and the item difficulty level will be high. Because the estimation of the difficulty level becomes more stable when more persons have answered an item, the weight given to new observations decreases when the rating of items is based on many observations. The same is true for the rating of the persons. When the rating of the person's knowledge level is based on a large amount of answered items, the weight given to new observations decreases.

In this study, the Elo Rating system implemented by Brinkhuis and Maris (2010) was used to estimate the item difficulty level. This Elo Rating system enables continuous measurement, since the rating is updated after every event. The formula for updating the item difficulty level, and on the same time the person's knowledge level, is given by:

$$\beta_n = \beta_0 + W(Y - Y_e),$$

where β_n is the new item difficulty rating after the item is answered by a person, β_0 is the pre-event rating, W is the weight given to the new observation, Y is the actual observation (score 1 for incorrect, 0 for correct), and Y_e is the expected observation which is estimated on the basis of the Rasch model. Hence, the formula for updating the item difficulty level after a correct response becomes:

$$\beta_n = \beta_0 + W \left[0 - \frac{\exp(\theta_0 - \beta_0)}{1 + \exp(\theta_0 - \beta_0)} \right],$$

where θ_0 is the estimated person's knowledge level before that person has answered this specific item. In this study, the weight has been set to 0.4. Preliminary analysis have shown that a weight of 0.4 results in good estimates as it is not too large, resulting in too much fluctuation, and it is not too small, resulting in a nearly invariant difficulty estimate.

Next to the comparison of the different alternative estimation methods with IRT-based calibration, we are interested whether the alternative estimation methods that are based on the binary response data of the learners, i.e. 1 for a correct response and 0 for an incorrect response, are sample dependent. If the correlation between these methods and the true difficulty parameter values are lower than the correlation between these methods and the difficulty parameter values obtained on the basis of IRT-calibration with the data gathered in this study, then these alternative methods are somewhat sample dependent. Furthermore, on the basis of the study of Impara and Plake (1998) it is hypothesized that the correlation between the true difficulty parameter values and the ones obtained by means of expert rating will be lower than the correlation between the true difficulty parameter values and the ones obtained by means of paired comparison conducted by the experts.

2.2 Method

2.2.1 Participants. Students from ten educational programs in the Flemish part of Belgium (1st and 2nd Bachelor Linguistics and Literature – K.U.Leuven; 1st, 2nd and 3rd Bachelor Teacher-Training for primary education – Katho Tielt; 1st and 2nd Bachelor Teacher-Training for secondary education – Katho Reno; 1st and 2nd Bachelor of Applied Linguistics – HUB and Lessius; and 1st Bachelor Educational Science – K.U.Leuven) were contacted to participate in the experiment. Three hundred eighteen students decided to participate. Sixteen teachers French from the above mentioned educational programs were contacted as experts. Thirteen experts decided to participate.

2.2.1 Material and Procedure. The study took approximately half an hour. The learning material consisted of items on French verb conjugation, supposedly measuring one single skill. The instructions, consisting of information on the login procedure for the learning environment and on the proceedings of the experimental study were sent to the participants by email. Once logged into the learning environment, the procedure for students was different from the procedure for experts.

Students were given an informed consent. Next, they completed the pretest used as an example. This pretest consisted of one item with three subquestions. First, the student had to fill in the correct French verb conjugation. Second, the student was asked: "How difficult did you find the previous item?" and the student has to answer on a 5 point Likert scale, ranging from -2 ("Very easy") to 2 ("Very difficult"). Finally, the student was asked to judge where the presented item should be located in the given series of 11 items ordered by difficulty level from easy to difficult. After the pretest sample, students completed the actual test, which consisted of 25 items each with three subquestions.

Experts completed the pretest used as an example. This pretest consisted of one item with three subquestions. First, the expert had to fill in the correct French verb conjugation. Second, the expert was asked: “What is, according to you, the percentage of students that will answer this item correctly after completing secondary education?”. Finally, the expert was asked to judge where the presented item should be located in the presented series of 11 items ordered by difficulty level from easy to difficult. After the pretest sample, experts completed the actual test, which consisted of 25 items each with three subquestions.

2.3 Results

The inter-rater agreement for the classification of the item difficulty was calculated by means of the intraclass correlation coefficient (ICC; Shrout & Fleiss 1979). Shrout and Fleiss (1979) report the magnitude for interpreting ICC values where $ICC < 0.40$ = “poor”, $0.40 \leq ICC \leq 0.59$ = “fair”, $0.60 \leq ICC < 0.74$ = “good”, and $ICC \geq 0.74$ = “excellent”. The inter-rater agreement for the classification of the item difficulty by students was fair ($ICC[3,1]=0.42$ for learner feedback; $ICC[3,1]=0.43$ for paired comparison). The inter-rater agreement for the classification of the item difficulty by experts was good ($ICC[3,1]=0.68$ for expert rating and for paired comparison). The inter-rater agreement for the classification of the item difficulty for paired comparison by experts and learners combined was fair ($ICC[3,1]=0.44$). The inter-rater agreement, when considering the mean of the paired comparison feedback given by learners and the mean of the paired comparison feedback given by experts, was excellent ($ICC[3,1]=0.88$).

The criterion used to evaluate the efficacy of the item difficulty estimation methods was the Pearson correlation between the estimated item parameter and its corresponding true parameter. The true difficulty parameter value for each item was estimated in advance by Selor, using examinee data for conducting the IRT analysis. Additionally the Pearson correlation was measured between the estimated item parameter and its corresponding IRT difficulty parameter value based on calibration with the data gathered in this study. The Pearson correlation between the estimated item difficulty parameter and the true item difficulty parameter is a measure for the strength of their linear relationship.

Detailed correlation results for the item difficulty estimates are shown in table I.

Table I. Pearson correlation matrix of the item difficulty estimates for the different estimation methods.

Item Difficulty Estimation Method	Item Difficulty Estimation Method							
	True β	IRT- Study	Proportio n Correct	Learner Feedback	Expert Rating	Paired Comparis on (Learner)	Paired Comparis on (Expert)	Elo Rating
True β	1.00							
IRT-Study	.90	1.00						
Proportion Correct	.90	1.00	1.00					
Learner Feedback	0.88	0.88	0.88	1.00				
Expert Rating	0.80	0.80	0.80	0.95	1.00			
Paired Comparison (learner)	0.62	0.50	0.50	0.58	0.55	1.00		
Paired Comparison (Expert)	0.56	0.44	0.44	0.53	0.51	0.98	1.00	
Elo Rating	0.85	0.92	0.92	0.81	0.73	0.45	0.39	1.00

The results of the Pearson correlation between the estimated item difficulty parameter and the true item difficulty parameter indicates that proportion correct has the strongest relation ($r(23)=0.90, p<0.01$), followed by learner feedback ($r(23)=0.88, p<0.01$), Elo rating ($r(23)=0.85, p<0.01$), expert rating ($r(23)=0.80, p<0.01$), paired comparison based on learners' feedback ($r(23)=0.62, p<0.01$) and paired comparison based on expert data ($r(23)=0.56, p<0.01$). The Pearson correlation between the estimated item difficulty parameter and the difficulty parameter estimated by means of IRT with the data of the 318 students in this study shows similar results. The correlation with proportion correct is the highest ($r(23)=1.00, p<0.01$), followed by Elo rating ($r(23)=0.922, p<0.01$), learner feedback ($r(23)=0.88, p<0.01$), expert rating ($r(23)=0.80, p<0.01$), paired comparison based on learners' feedback ($r(23)=0.50, p<0.05$) and finally paired comparison based on expert data ($r(23)=0.44, p<0.05$).

The difference between the correlation coefficient of proportion correct with the true difficulty parameter value and the correlation coefficient of proportion correct with the difficulty parameter value estimated by means of IRT with the data of this study is significant ($t(22)=-19.18, p<0.05$). The difference between the correlation of the Elo rating system with the true difficulty parameter value and the correlation of the Elo rating system with the difficulty parameter value estimated by means of IRT with the data of this study is also significant ($t(22)=-2.09, p<0.05$). The correlation coefficient of proportion correct with the IRT calibration based on the study data differs significantly from the correlation coefficient of the Elo rating system with the IRT calibration based on the study data ($t(22)=20.7485, p<0.05$). The significance disappears when proportion correct and the Elo rating system are compared with the true difficulty parameter value ($t(22)=1.46, p=0.16$).

There is no significant difference between the correlation of the true item difficulty parameter values with the ones obtained by means of expert rating, and the correlation of the true item difficulty parameter values with the ones obtained by means of paired comparison based on expert ratings ($t(22)=1.89, p=0.07$). The difference between the correlation coefficient of learner feedback with the true difficulty parameter value and the correlation coefficient of expert rating with the true difficulty parameter value is significant ($t(22)=2.71, p<0.05$). However, the difference between the correlation coefficient of paired comparison based on learner feedback with the true difficulty parameter value and the correlation coefficient of paired comparison based on expert rating with the true difficulty parameter value is not significant ($t(22)=1.85, p=0.08$).

3. DISCUSSION

As the tracking and logging data of many learning environments fail to contain the required amount and structure of data needed for IRT estimation, this article searches for appropriate alternative methods to estimate the difficulty level of items. Based on the response data and the judgment data of a sample of learners and experts, the difficulty level of twenty five items was estimated by means of six estimation methods: (1) IRT calibration based on the study data, (2) proportion correct, (3) learner feedback, (4) expert rating, (5) paired comparison (based on learners' judgment and based on experts' judgment), and (6) the Elo rating system.

The findings indicate that proportion correct has the strongest relation with the true difficulty parameter values, followed by learner feedback, the Elo rating system, expert rating and paired comparison. Furthermore, proportion correct also has the strongest relation with the difficulty estimates obtained with IRT calibration on the study data, followed by the Elo rating system, learner feedback, expert rating and paired comparison. Considering the alternative estimation methods that are based on the binary response of the learners (correct vs. incorrect response to an item), it is shown that IRT calibration, proportion correct and the Elo rating system do not differ. The high correlation found between IRT calibration (both true difficulty parameter and IRT calibration on the study data) and proportion correct is not surprising as the total score is a sufficient statistic for the Rasch model. Furthermore, it is clear that proportion correct and the Elo rating system are sample dependent as they correlate higher with the IRT calibration on the study data than with the true difficulty parameter values.

Results contradict the postulation of Impara and Plake (1998) that experts perform better in estimating the difficulty by rank ordering the items than by estimating the proportion of persons who will answer the items correctly. Furthermore, findings indicate that learners perform better on judging the difficulty of items than experts. However, this difference disappears when learners and experts need to rank order the items according to their difficulty level. It needs to be considered that the estimation by means of learner feedback is based on a larger sample than the estimation by means of expert rating, which could explain the difference between learner feedback accuracy and expert rating accuracy. The finding that the correlation of paired comparison with the true difficulty parameter is moderate could be due to the small sample size, resulting in some outlier estimations. The paired comparison data are analyzed by means of the graded response model, which is a more complex IRT model than the Rasch model, and hence may need a larger sample size to obtain reliable item difficulty estimates.

Even though this study indicates that the difficulty of items can be estimated on the basis of alternative estimation methods, it should be considered that the size of the item set that was used to compare the alternative estimation methods was rather small. We recognize that a total number of twenty five items is limited, but considering raters fatigue, we were compelled to keep the item set rather small. Furthermore, we made sure that the twenty five items covered a broad range of difficulty.

Future research will focus on the sample size requirement for reliable difficulty estimates. The different alternative estimation methods will be compared for different sample sizes. If results would indicate that alternative estimation methods provide reasonable accurate difficulty level estimates, these estimation methods could be used to provide adaptive curriculum sequencing. Those alternative estimation methods could also be used to make IRT estimation more efficient by using the estimates as prior in a Bayesian estimation method. A limitation of this study, which should be tackled in future research, is the fact that even though some of the alternative item difficulty estimation methods seem to be a viable alternative for IRT-based calibration in this study, no generalization can yet be made to other domains and to items requiring more than one skill.

REFERENCES

- BRINKHUIS, M.J.S., AND MARIS, G. 2010. Adaptive Estimation: How to Hit a Moving Target. Report No.2010-1, Measurement and Research Department, Cito, Arnhem.
- BRUSILOVSKY, P. 1999. Adaptive and Intelligent Technologies for Web-Based Education. *Künstliche Intelligenz*, 4, 19-25.
- CHALIFOUR, C.L., AND POWERS, D.E. 1989. The Relationship of Content Characteristics of GRE Analytical Reasoning Items to Their Difficulties and Discriminations. *Journal of Educational Measurement*, 26, 120-132.
- CHEN, C.M., LEE, H.M., AND CHEN, Y.H. 2005. Personalized e-Learning System Using Item Response Theory, *Computers & Education*, 44, 237-255.
- CHEN, C.M., LIU, C.Y., AND CHANG, M.H. 2006. Personalized Curriculum Sequencing Utilizing Modified Item Response Theory for Web-Based Instruction. *Expert Systems with Applications*, 30, 378-396.
- CHEN, C.M., AND DUH, L.J. 2008. Personalized Web-Based Tutoring System Based on Fuzzy Item Response Theory. *Expert Systems with Applications*, 34, 2298-2315.
- EGGEN, T.J.H.M. 1993. Itemresponstheorie en onvolledige gegevens. In Eggen, T.J.H.M., Sanders, P.F. (Eds) *Psychometrie in de Praktijk*, Cito, Arnhem.
- ELO, A.E. 1978. *The rating of chess players, past and present*, B.T. Batsford Ltd., London.
- FERNANDEZ, G. 2003. Cognitive Scaffolding for a Web-Based Adaptive Learning Environment. *Advances in Web-Based Learning - IcwL 2003, Proceedings*, 2783, 12-20.
- HAMBLETON, R.K., BASTARI, AND XING, D. 1998. Estimating Item Statistics. Report No.298, Laboratory of Psychometric and Evaluative Research, School of Education, University of Massachusetts.
- IMPARA, J.C., AND PLAKE, B.S. 1998. Teachers' Ability to Estimate item Difficulty: a Test of the Assumptions in the Angoff Standard Setting Method. *Journal of Educational Measurement*, 35(1), 69-81.
- JOHNS, J., MAHADEVAN, S., AND WOOLF, B. 2006. Estimating Student Proficiency Using an Item Response Theory Model. *Intelligent Tutoring Systems, Lecture Notes in Computer Science*, 4053, 473-480.
- KIM, S. 2006. A Comparative Study of IRT Fixed Parameter Calibration Methods. *Journal of Educational Measurement*, 43, 355-381.
- LEUNG, E.W.C. AND LI, Q. 2007. An Experimental Study of a Personalized Learning Environment Through Open-Source Software Tools. *IEEE Transaction on Education*, 50, 331-337.
- LIKERT, R. 1932. A Technique for the Measurement of Attitudes. *Archives of Psychology*, 22, 1-55.
- LINACRE, J.M. 1994. Sample Size and Item Calibrations Stability. *Rasch Measurement Transaction*, 7(4), 328.
- LINACRE, J.M. 2000. Computer-Adaptive Testing: A Methodology whose Time has Come. In Chae S., Kang U., Jeon E., Linacre J.M. (Eds) *Development of Computerized Middle School Achievement Test*, Komesa Press, Seoul.
- LU, F., LI, X., LIU, Q.T., YANG, Z.K., TAN, G.X., AND HE, T.T. 2007. Research on Personalized e-Learning System Using Fuzzy Set Based Clustering Algorithm. *Computational Science - ICCS 2007, Part 3, Proceedings*, 4489, 587-590.
- OZAKI, K., AND TOYODA, H. 2006. Paired Comparison IRT Model by 3-Value Judgment: Estimation of Item Parameters Prior to the Administration of the Test. *Behaviormetrika*, 33, 131-147.
- OZAKI, K., AND TOYODA, H. 2009. Item Difficulty Parameter Estimation Using the Idea of the Graded Response Model and Computerized Adaptive Testing. *Japanese Psychological Research*, 51, 1-12.
- PÉREZ-MARÍN, D., ALFONSECA, E., AND RODRIGUEZ, P. 2006. On the Dynamic Adaptation of Computer Assisted Assessment of Free-Text Answers. *Adaptive Hypermedia and Adaptive Web-Based Systems, Proceedings*, 4018, 374-377.
- SAMEJIMA, F. 1969. Estimation of Latent Trait Ability Using a Response Pattern of Graded Scores. *Psychometrika Monograph*, 17.
- SHROUT, P.E., AND FLEISS, J.L. 1979. Intraclass Correlations: Uses in Assessing Rater Reliability. *Psychological Bulletin*, 86, 420-428.
- THURSTONE, L.L. 1994. A Law of Comparative Judgment. *Psychological Review*, 101(2), 266-270.
- TSUTAKAWA, R.K., AND JOHNSON, J.C. 1990. The Effect of Uncertainty of Item Parameter Estimation on Ability Estimates. *Psychometrika*, 55, 371-390.
- VAN DER LINDEN, W.J., AND HAMBLETON, R.K. 1997. *Handbook of Modern Item Response Theory*, Springer, New York.
- WAINER, H. 2000. *Computerized Adaptive Testing: a Primer*, Erlbaum, London.
- WAUTERS, K., DESMET, P., AND VAN DEN NOORTGATE, W. 2010. Adaptive Item-Based Learning Environments Based on the Item Response Theory: Possibilities and Challenges. *Journal of Computer Assisted Learning*, 26(6), 549-562.
- YAO, T. 1991. CAT with a Poorly Calibrated Item Bank. *Rasch Measurement Transactions*, 5, 141.

Spectral Clustering in Educational Data Mining

SHUBHENDU TRIVEDI, ZACHARY A. PARDOS,

GÁBOR N. SÁRKÖZY, NEIL T. HEFFERNAN

Worcester Polytechnic Institute, United States

Spectral Clustering is a graph theoretic technique for metric modification such that it gives a much more global notion of similarity between data points as compared to other clustering methods such as k-means. It thus represents data in such a way that it is easier to find meaningful clusters on this new representation. It is especially useful in complex datasets where traditional clustering methods would fail to find groupings. In previous work we have shown the utility of using k-means clustering for exploiting structure in the data to affect a significant improvement in prediction accuracy on educational datasets. In this work we show that by using Spectral Clustering we are able to further improve the student performance prediction. We evaluate an educational data mining prediction task: predicting student state test scores from features derived from a tutor and also present some preliminary results on some other EDM tasks using spectral clustering.

Categories and Subject descriptors: I 2.7 [Artificial Intelligence]

Key Words and Phrases: Educational Data Mining, Intelligent Tutoring Systems, Bootstrap Aggregating, Clustering, Spectral Clustering, Ensemble Learning, Mixture of Experts

1. INTRODUCTION

The highly inter-disciplinary field of Educational Data Mining (EDM) has resulted from a fusion of many different areas, some of which include Machine Learning, Cognitive Science and Psychometrics. The main task in EDM is to construct computational models and tools to mine data that originated in an educational setting. With rapidly increasing data repositories from different educational contexts (paper tests, e-learning, Intelligent Tutoring Systems etc.), good practices in EDM can potentially answer important research questions about student learning. This goal of EDM is proving instrumental in combining the knowledge derived from the data to combine with theories from cognitive psychology to formulate the best learning settings and methodologies.

Within data mining, clustering is perhaps one of the most important tools for both exploratory and confirmatory analysis. It is a technique to discern meaningful patterns in unlabeled data by grouping together data points that are “similar”. In EDM, clustering has been used in a variety of contexts: Ritter *et al.* In an already influential work essentially used the implicit information compression (albeit lossy) handed by clustering to reduce the Knowledge Tracing parameter space [Ritter 09] without compromising the performance of the system. Dominguez *et al.* used clustering as a tool to generate individualized hints for students [Dominguez 10]. In another interesting work, Shih *et al.* employed clustering for unsupervised discovery of student learning tactics [Shih 10]. Clustering has also been used for curriculum planning [Maull 10], for estimating skill set profiles [Nugent 10] amongst numerous other tasks. However, interestingly most of these works employ k-means clustering, expectation maximization based clustering or subspace clustering. This paper aims to introduce to the field of EDM the utility handed by spectral clustering over other clustering algorithms, which is also an easy to implement algorithm with numerous toolboxes available as well [Chen 10].

Authors' addresses: S. Trivedi, e-mail: shubhendu_trivedi@ieee.org; Z. A. Pardos, e-mail: zpardos@cs.wpi.edu, G. N. Sárközy, e-mail: gsarkozy@cs.wpi.edu, N. T. Heffernan, e-mail: nth@cs.wpi.edu, Department of Computer Science, Worcester Polytechnic Institute, 100 Institute Road, Worcester, MA – 01609. United States.

To understand the weakness of methods such as k-means, a useful way of looking at clustering is the following: Consider a set of K distributions, $\mathcal{D} = \{D_1, D_2 \dots D_K\}$ such that each of these distributions has an associated weight, the collection of which is given by $\{w_1, w_2 \dots w_K\}$ such that $\sum_i w_i = 1$. Suppose a dataset is generated by sampling these K distributions, such that a point in this dataset might be picked from distribution D_i with probability w_i . The objective of clustering methods is to identify these K distributions given a dataset. Methods such as k-means and Expectation Maximization (EM) are based on estimating explicit models of the data. While k-means finds the clusters by assuming that the set of distributions \mathcal{D} that generated the data was a set of spherical Gaussians, EM algorithms in general learn a mixture of Gaussians with arbitrary shapes. More formally, k-means finds the clusters by minimizing the distortion function:

$$J(c, \mu) = \sum_{i=1}^m \|x^{(i)} - \mu_{c^{(i)}}\|^2 \quad (1)$$

Where μ_c is the cluster centroid to which a point x has been assigned. In spite of the great popularity of the k-means algorithm very few theoretical guarantees on its performance are known [Dasgupta 99]. In practice however, k-means performs well on data that at least approximately follows its assumption of being generated by a mixture of well-separated spherical Gaussians [Chaudhuri 09]. This, coupled with its simplicity makes it a handy tool for a data-miner. However, k-means performs poorly when these assumptions of data generation are not met, which is usually the case in real world datasets. Fig. 1 illustrates this problem by a toy synthetic dataset.

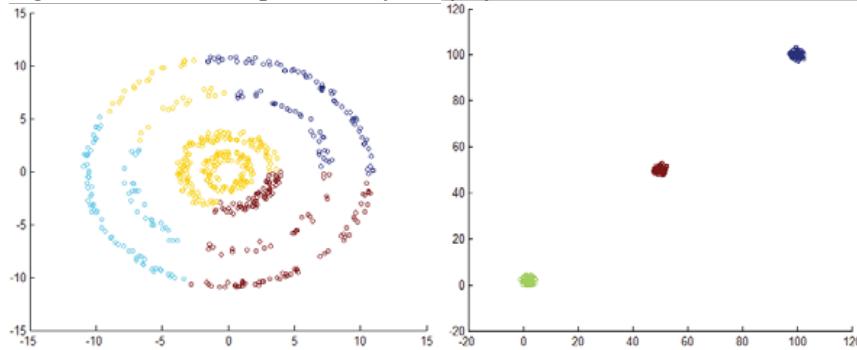


Fig 1: Results of using k-means on synthetic datasets. k-means is unable to identify clusters when the data is distributed in concentric groups (left), while it clearly finds the clusters in well separated and tight spherical Gaussians (right). The clusters identified are indicated by different colors. Both sets have 600 points.

Spectral Clustering makes no such assumptions for data generation. It instead finds groupings by analyzing the top eigenvectors of the affinity matrix and hence usually returns better results.

The rest of the paper is organized as follows: The next section discusses Spectral Clustering, giving a tutorial overview of the same. Section 3 uses the spectral clustering method to improve the prediction of post-test scores employing student features from an Intelligent tutor using a bootstrap aggregating method developed by the authors [Trivedi, Pardos 11] [Trivedi, Pardos 11]. Section 4 is a discussion of results and future work.

2. SPECTRAL CLUSTERING

One of the most important developments in Machine Learning in the past decade has been the use of spectral methods in clustering. They have created a new wave of excitement to understand the problem of clustering and the notion of similarity between points better and formulate it precisely. One major reason for this excitement is that

spectral clustering is based on solid graph theoretic principles. Given its strengths, it would be highly beneficial to the EDM community if it is used more widely in the same.

The broad idea of clustering is essentially to group points that are “similar” in one cluster and points that are “dissimilar” into different clusters. The notion of similarity that is employed in k-means is the Euclidean distance between data points and the cluster centroids to which they are assigned to (which get updated in each iteration). In a sense, the idea of similarity used in k-means restricts what could be known about the geometry of the data. In k-means we work with the data directly, in spectral clustering however, we work with a representation of the data that gives a more global (and hence better) encoding of the similarities between points. This “similarity” in spectral clustering is represented in the form of a graph called the similarity graph, represented by $\mathcal{G} = (V, E)$ where V is the set of vertices and E is the set of edges. The idea is that points in the dataset can be represented by a graph with each data point as a vertex of the graph \mathcal{G} and the edges connecting them encoding a notion of similarity $w_{ij} > 0$ between them. Two points are connected in the graph if the similarity or weight between them is either non-zero or above some threshold. The clustering problem can then be re-stated using information from the similarity graph as: We want to find partitions of this graph such that weights between points in the same group are high and those between points in different groups are low. Before talking how we cluster using this representation, we introduce some notation and discuss how the graph \mathcal{G} is used to represent the dataset.

Given the similarity graph \mathcal{G} of n data points $\{x_1, x_2 \dots x_n\}$, there are essentially two things about it that tell us something about the global structure of the data:

1. The degree of a vertex (a data-point in our case): The degree of a vertex tells us the sum of weights of all the edges that originate from a vertex i to all other vertices j . It is given by:

$$d_i = \sum_{j=1}^n w_{ij}$$

This definition is somewhat non-standard but more general. The standard definition for degree of a vertex is only defined for $w_{ij} = \{0, 1\}$, and thus is only the count of vertices a given vertex is connected to. Given this definition, the degree matrix of the similarity graph is the diagonal matrix \mathbf{D} with the degrees d_i on the diagonal.

$$\mathbf{D} = \begin{pmatrix} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_n \end{pmatrix}$$

Intuitively the degree matrix of a graph tells us how many points each point is connected to (we could connect all points, or choose to connect k-nearest neighbors of each point) and by “how much” (hence the summation of the weights).

2. The weighted similarity matrix or the affinity matrix of the similarity graph, \mathbf{W} on the other hand is a representation of similarity between all the points. Each element in the affinity matrix is given by w_{ij} , which is the weight or edge between two points i and j . A common way of representing the weight is:

$$w_{ij} = \exp\left(\frac{-\|x_i - x_j\|^2}{2\sigma^2}\right) \quad (2)$$

Notice that w_{ij} is simply the exponentiated Euclidean distance between two points (points in \mathcal{R}^n) scaled by a parameter called the scaling or weighing parameter σ . This parameter is to be tuned and varying it changes the weight between points. A point to note is that if all the points are connected then all w_{ij} such that $i \neq j$ will be non-zero values. If points are connected to only their k-nearest neighbors and not every other point, then most of the matrix W will be populated by zeros.

The matrices W and D tell us something about the global structure of the data, but we don't work with them directly. We instead work with the graph Laplacian matrix given by

$$L = D - W$$

The above is the un-normalized version of the Laplacian. There are two normalized versions that are represented as:

$$\begin{aligned} L_{sym} &= D^{-1/2}WD^{-1/2} \\ L_{rw} &= D^{-1}W \end{aligned}$$

The first is called the symmetric Laplacian while the second is called the random-walk Laplacian. The Laplacian in a way combines both the degree and the affinity matrix and also has some mathematically interesting properties (such as being positive semi-definite) that make it easier to work with [Mohar 91]. Since the Laplacian is a representation of the similarity between the data-points, we can now work with it to find groups in the data.

Given the above background, clusters in a dataset can be found by the following method [Ng 01]:

1. For the dataset having n data points, construct the similarity graph \mathcal{G} . The similarity graph can be constructed in two ways: by connecting each data point to the other $n - 1$ data points or by connecting each data point to its k-nearest neighbors. A rough estimate of a good value of the number of nearest neighbors is $\log(n)$. The similarity between the points is given by equation 2. This will give the matrix W .
2. Given the similarity graph, construct the degree matrix D .
3. Using D and W find L_{sym} .
4. Let K be the number of clusters to be found. Compute the first K eigenvectors of L_{sym} . Sort the eigenvectors according to their eigenvalues.
5. If $u_1, u_2 \dots u_K$ are the top eigenvectors of L_{sym} , then construct a matrix U such that $U = \{u_1, u_2 \dots u_K\}$. Normalize rows of matrix U to be of unit length.
6. Treat the rows in the normalized matrix U as points in a K dimensional space and use k-means to cluster these.
7. If $c_1, c_2 \dots c_K$ are the K clusters, Then assign a point in the original dataset s_i to cluster c_K if and only if the i^{th} row of the normalized U is assigned to cluster c_K .

It is noteworthy that we don't cluster the original dataset directly. We first transform it to find its top K eigenvectors. These being the most important eigenvectors of L_{sym} , encode the maximum information about it. At the same time, this reduces the dimensionality which without throwing away much information which makes the task of clustering much easier. To illustrate the power given by this change of representation, we demonstrate it on a toy dataset (Fig. 2). A detailed tutorial that explains various spectral clustering

algorithms and some point of views on why it works is by Luxburg [Luxburg 07]. In the next section we discuss a specific application of spectral clustering in EDM.

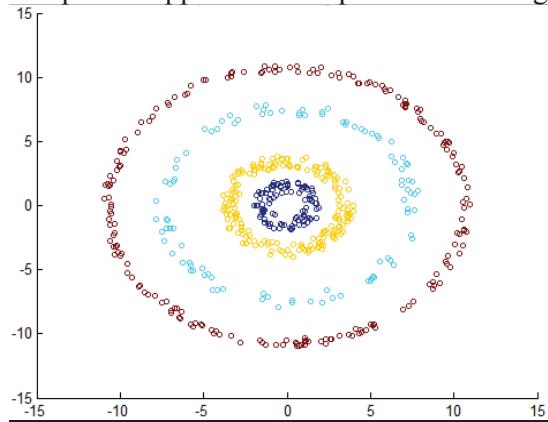


Fig 2: Result of using spectral clustering on a synthetic dataset. This synthetic set has 600 points. The colors indicate the clusters found by spectral clustering. Such groups cannot be found by k-means clustering.

3. IMPROVING PREDICTION ON STUDENT PERFORMANCE IN POST TESTS

Bayesian Knowledge Tracing [Corbett 95] has long been used to model student knowledge in an intelligent tutoring system (ITS). This knowledge estimate is used to calibrate the amount of training a student gets to ensure skill mastery. One of the goals of such modeling is to ensure that students perform well on actual post tests. In fact it is reasonable to say that perhaps one of the most important measures of success of an ITS is how well performance on it transfers to actual post tests.

Traditionally, performance on a post-test is predicted by using practice tests. The percentage of questions answered correctly on these practice tests give a crude estimate of how well a student would perform on the actual post test. Improving this estimate would be highly beneficial to both students and educators. For the improvement of such assessment, dynamic assessment [Grigerenko 98] has been advocated as an effective method. The big idea of dynamic assessment is that assessment is based on the amount of help students require to get questions correct and it enables the tutor to assess as it assists. This is a major advantage as it not only allows students to learn while being assessed, but can also predict student performance on post-tests better. Traditional testing, in which only the percentage of questions is considered is called static assessment. The notion of dynamic assessment makes intuitive sense as it gives a finer grained estimate of a student's knowledge. If a student gets a question wrong, it might not imply that the student has no knowledge pertaining to the question. The level of knowledge that the student has might be estimated by measuring the amount of help that the student required to get the question correct. Given the interactive nature of ITS, they are the ideal test bed for measuring the utility of dynamic assessment.

Feng *et al.* [Feng 09] reported the result that data from an ITS could better predict state test scores (MCAS or Massachusetts State Test Scores in their experiment) if it only considered the extra measures collected in dynamic assessment as compared to the static assessment condition. The paper had a weakness that time was never held constant. Feng & Heffernan went one step ahead and controlled for time in following work [Feng 10]. They reported better predictions on the MCAS state test scores by the dynamic condition, but not a statistically reliable difference. This work effectively showed that dynamic assessment led to better predictions on the post test. This prediction was done by fitting a linear regression model on the dynamic assessment features and making predictions on the MCAS test scores. They concluded that while Dynamic Assessment gave good assessment of students, the MCAS predictions made using those features were only

marginally statistically significant as compared to the static condition. Trivedi *et al.*[Trivedi 11] investigated further if the dynamic assessment data could be better utilized to increase prediction accuracy over the static condition (and hence establish the superiority of dynamic assessment). They used a newly introduced method [Trivedi 11] that clusters students using the k-means algorithm and uses multiple cluster models and then ensembles the predictions made by each cluster model to achieve a reliable improvement. Here we show that by using spectral clustering we further improve the prediction on the MCAS post-test based on the dynamic features. The improvement obtained by using spectral clustering is not only significant over the static condition, but also over results obtained using k-means after $K = 3$ (p -value < 0.03 on a paired t-test).

3.1 Data and Methodology

The data used for this study was the same as used by Feng *et al.*[Feng 10] and Trivedi *et al.* [Trivedi 11]. The data is from the 2004-05 school year and was collected using the ASSISTments tutor in two schools in Massachusetts. ASSISTments [Razzaq 05] is an ITS developed at Worcester Polytechnic Institute, MA, USA. The data is for 628 students and the features included the various dynamic features [Feng 10]. These features were: 1) Student's percent correct on main problems 2) Number of problems done 3) Percent correct on the help questions 4) Average time spent per item 5) Average number of attempts per item 6) Average number of hints per item. The first feature was a static feature and was used to make predictions on the static condition, while the others were used to make predictions in the dynamic condition. The prediction made was for the MCAS test scores that was available for the same students in the following year. A 5 fold cross validation was done.

The methodology used for making the prediction is a new bootstrap aggregation ensemble method [Trivedi, Pardos 11]. The procedure is summarized as follows:

1. Cluster the training data into K clusters.
2. For each cluster train a separate linear regression model using the points from that cluster as the training set.
3. Each such trained predictor (such as Linear Regression) represents a model of the cluster and is hence appropriately called a cluster model.

This is represented in figure 3 below:

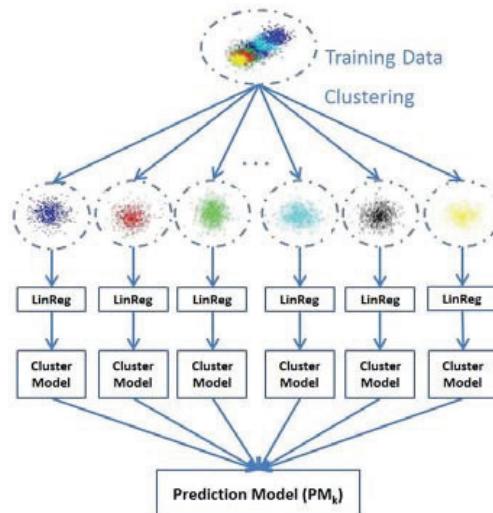


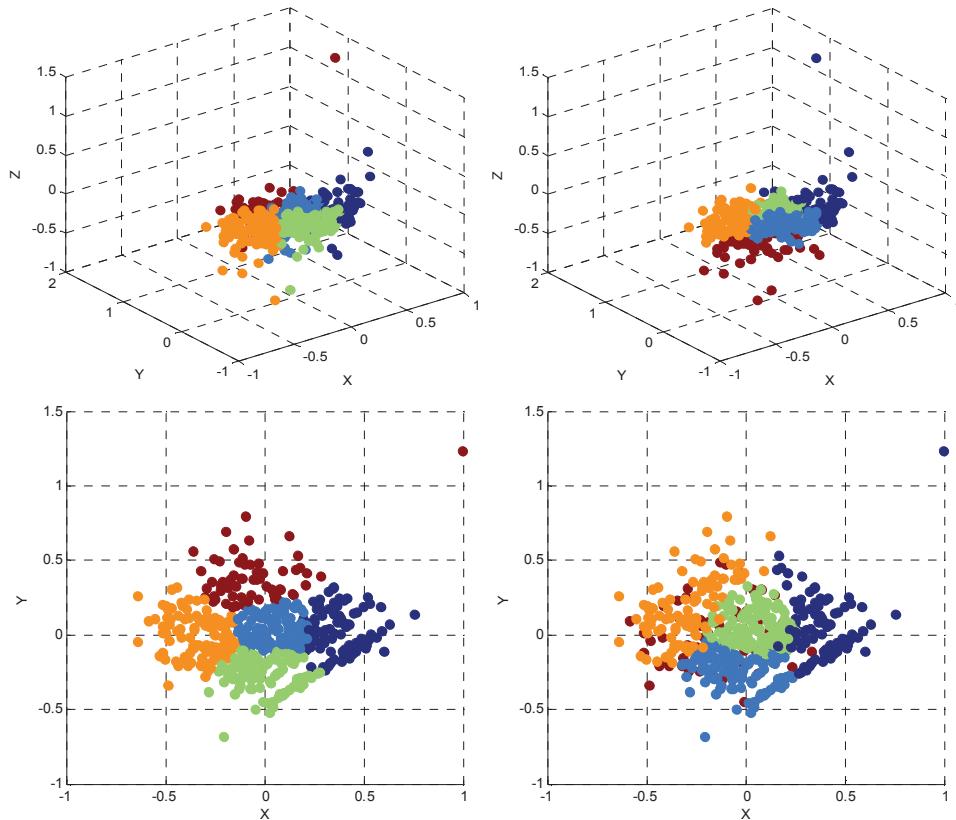
Fig 3: The first step in the methodology for using clustering to bootstrap and making a prediction on the training set. The scale of clustering can be varied to generate a number of predictions that can then be aggregated.

This collection of cluster models that make a prediction on the entire test set is called a prediction model (PM_K , the subscript denotes the number of clusters in each Prediction Model). Making a prediction for a test point would involve: Locating the cluster to which the point belongs, and then using the model trained for that cluster to make the prediction for it. However, by using the number of clusters as a free parameter we generate a set of K prediction models (PM_2 , PM_3 ... PM_K), such that each has a different number of cluster models. And thus, we can obtain K different predictions on the test set. These predictions are then averaged to obtain a single strong prediction.

This method can be thought of like an adaptive mixture of local experts [Jacobs, Hinton 91] that uses clustering to bootstrap. But unlike in other bagging methods, which select a random subset to bootstrap, this method has a specific expert for each cluster of the data. By varying the granularity of the clustering we are able to obtain a mixture of experts on the data at different levels each of which gives a prediction on the test set which are then averaged to get one prediction.

3.2 Results Using Spectral Clustering

The results of clustering the data using both kmeans and spectral clustering are represented in figure 4 below. Since the data is high dimensional and the actual partitions cannot be pictured, this visualization is done by doing a multi-dimensional scaling on the dataset to three dimensions with each cluster identified by a different color. This visualization is for $K = 5$.



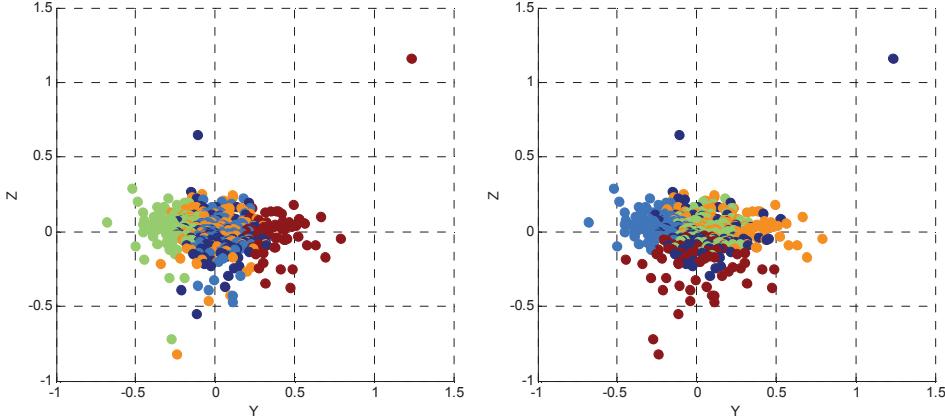


Fig 4: The images on the left column are for k-means and those on the right are for spectral clustering. The top row represents the plot of the ASSISTment data scaled down by multi-dimensional scaling to three dimensions and the clusters identified by both k-means and spectral clustering. The rows below are simply different planar views of the plot in row 1.

The spectral clustering ensemble results are not only significant over the static condition ($K = 1$ in figure 6) but also are significant for the kmeans generated ensemble beyond $K = 3$ with $p < 0.03$ in each case on a paired t-test.

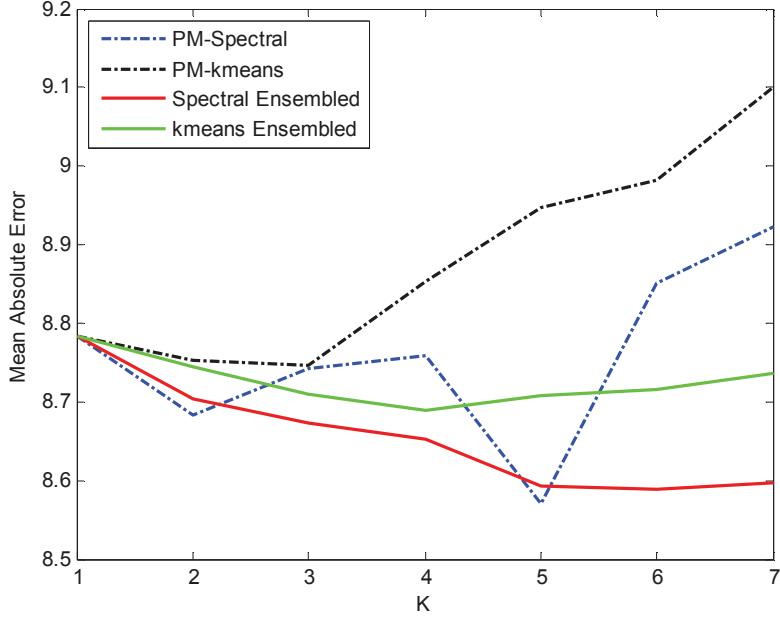


Fig 5: The plots of the 5 fold cross validated errors by the various prediction models and ensembles (from 1 to $K = 7$) for both kmeans and spectral clustering. The K ensemble prediction is the average of predictions returned by prediction models from 1 to K .

4. CONCLUSION, DISCUSSION AND FUTURE WORK

The methodology described in the paper was employed on some other EDM tasks as well, such as making an in-tutor prediction on the KDD Cup 2010 dataset and on the Performance Factor Analysis (PFA) task [Gong 10]. Preliminary results (summarized for PFA below) have indicated an improvement in the prediction accuracies.

Table 1: Preliminary work on Performance Factor Analysis

Spectral Ensemble	K = 1	K = 2	K = 3	K = 4	K = 5
AUC	0.5861	0.6153	0.6252	0.6291	0.6307

The results indicate an improvement over the base condition as more prediction models are averaged. But this result is not cross-validated and is a work in progress. Also, given the prohibitive size of the dataset, spectral clustering was not used for all the rows in the training set, but a random subset of them was used. This was done to save time, however this is the reason the detailed results are not reported in this work. Also, more work needs to be done to use spectral clustering methods more efficiently for massive datasets such as the KDD cup dataset.

A deeper way of looking at clustering is essentially as a scheme for lossy data compression. Improvement in prediction accuracy using spectral clustering over k-means indicates that spectral clustering is a better information compression method than k-means and hence tells something deeper about the structure of the data that k-means misses. This would mean an interesting application to reduce the knowledge tracing space like by Ritter *et al* [Ritter 2009] and see how it compares with performance returned by k-means clustering.

The objective of this work was to introduce to the domain of EDM the great utility of using spectral clustering. We used spectral clustering to enhance the performance of a new ensemble method proposed in an earlier work by the authors. While the objective was to introduce the use of spectral clustering, a very significant result of the work is proving the efficacy of Dynamic Assessment as compared to static assessment. These results show that an ITS that can assess as it assists offers a significant advantage to students and teachers. This is important because it can not only save time that is wasted on assessment for instruction, but it can also be a better predictor of their performance in post-tests.

The results for the task of predicting the post test scores have been very encouraging; however there are some areas that need further work and could improve prediction accuracy further. One such area of possible improvement is allowing for fuzzy clustering. To make a prediction, the cluster closest to a test point was identified and then the expert for that cluster was used to make a prediction on it. In many real world examples, membership of a data point to a particular cluster is a tricky question to answer. A more realistic view is to allow for fuzzy clustering. That is, given a test point, we determine its probability of occurring in each of the clusters. Then, we can obtain predictions by the cluster model /expert for each of the clusters and obtain one prediction for one test point that is a weighted average of the predictions returned by each cluster model (earlier a prediction was made on the test point by only one cluster model), with the weights being the probability that the point lies in that cluster. While fuzzy counterparts to the k-means algorithm such as fuzzy c-means are well known, the idea of doing fuzzy spectral clustering is something to be explored. Clearly, spectral clustering uses k-means at a lower dimensional representation of the laplacian and fuzzy c-means can be used at this level. However, the effectiveness of doing the same is not known.

Another possible area of improvement is using methods to merge clusters that are sparsely populated [Cheng 06]. By this method we could improve both the quality of clustering (if the task is purely unsupervised) and prediction accuracy (if the task like in the application discussed is a prediction task).

In this work we combine predictions by averaging them. Clearly this is a sub-optimal choice. Ideally, we would want to pick those predictions (made by prediction models) which are good in prediction and have less correlation with each other (are diverse). Since the method used to make the post-test predictions was an ensemble method, it can be used to combine the predictions themselves. Preliminary work utilizing this idea of using clustering to boot-strap the predictions returned by various prediction models has shown promise.

ACKNOWLEDGEMENTS

We are grateful to the following funders for supporting this research: <http://www.webcitation.org/5ym157Yfr>. We would also like to thank the Pittsburgh Science of Learning Centre for the Cognitive Tutor KDD dataset. Comments about this work by Dr Carolina Ruiz, Dr Sergio Alvarez and Dr Alexandru Niculescu-Mizil were especially helpful and are appreciated.

REFERENCES

- CHAUDHURI., K., DASGUPTA S., VATTANI, A., 2009, Learning Mixture of Gaussians using the k-means Algorithm, In *CoRR vol.abs/0912.0086*, <http://arxiv.org/abs/0912.0086>, 2009
- CHEN., W. Y., SONG, Y., BAI, H., LIN, C. J, CHANG, E. Y., 2010, (ACCEPTED) Parallel Spectral Clustering in Distributed Systems, In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.
- CHENG D., KANNAN, R., VEMPALA, S., WANG, G., 2006, A Divide and Merge Methodology for Clustering. In *The Journal of the ACM, Vol V*, 2006
- CORBETT A. T., ANDERSON, J. R., 1995, Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. In *User Modeling and User Adapted Interaction* 4, pp. 253-278, 1995
- DASGUPTA S., 1999, Learning Mixture of Gaussians, In *The 40th Annual IEEE symposium on Foundations of Computer Science*, 349-358.
- DOMINGUEZ., A. K., YACEF, K., CURRAN, J. R., 2010, Data Mining for Individualized Hints in eLearning, In *Proceedings of the Third International Conference on Educational Data Mining*, 2010, 91-100.
- 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.
- FENG., M., HEFFERNAN, N. T., 2010, 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). *Proceedings of the 3rd International Conference on Educational Data Mining*, 41-50.
- GONG, Y., BECK, J. E, HEFFERNAN. N. T., 2010, (Accepted). How to Construct More Accurate Student Models: Comparing and Optimizing Knowledge Tracing and Performance Factor Analysis. In *International Journal of Artificial Intelligence in Education*. Accepted, 2010.
- GRIGERENKO., E. L., STEINBERG, R. J, 1998, Dynamic Testing. In *Psychological Bulletin* 124 pp. 75-111, 1998.
- JACOBS., R. A., JORDON, M. I., NOWLAN. S. J., HINTON. G.E., 1991, Adaptive Mixture of Local Experts. In *Neural Computation*, Vol 3. No 1, 79-87, 1991.
- LUXBURG U., 2007, A Tutorial on Spectral Clustering, In *Statistics and Computing*, *Kluwer Academic Publishers, Hingham, MA, USA. Vol 17. Issue 4*, 2007.
- MAULL. K. E., SALVIDAR. M. G., SUMNER. T., 2010, Online Curriculum Planning Behavior of Teachers, In *Proceedings of the Third International Conference on Educational Data Mining*, 2010, 121-130.
- MOHAR. B., The Laplacian Spectrum of Graphs, In *Graph Theory, Combinatorics and Applications*, 871-898, 1991.
- NG. A. Y, JORDON. M. I, WAISS. Y., 2001, On Spectral Clustering: Analysis and an Algorithm. In *Advances in Neural Information Processing Systems* 14, 2001.
- NUGENT. R., DEAN. N., AYERS. E., 2010, Skill-Set Profile Clustering: The Empty K-Means Algorithm with Automatic Specification of Starting Cluster Centers. In *Proceedings of the Third International Conference on Educational Data Mining*, 2010, 151-160.
- RAZZAQ. L., FENG M., NUZZO-JONES. G., HEFFERNAN. N. T., KOEDINGER. K. R., JUNKER. B., RITTER S., KNIGHT A., ANISZCZYK. C., CHOKSEY. S., LIVAK. T., MERCADO. E., TURNER. T. E., UPALEKAR.R., WALONOSKI. J.A., MACASEK M. A., AND RASMUSSEN. K. P., 2005, The Assitment Project: Blending Assessment and Assisting. In C. K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds). *Proceedings of the 12th International Conference on Artificial Intelligence in Education*, Amesterdam. ISO Press, pp 555-562.
- RITTER. S., HARRIS. T. K., NIXON. T., DICKISON. D., MURRAY. R.C., TOWLE. B., 2009, Reducing the Knowledge Tracing Space, In *Proceedings of the Second International Conference on Educational Data Mining*, 2009, 151-160.
- SHIH. B., KOEDINGER. K. R., SCHEINES. R., 2010, Unsupervised Discovery of Student Learning Tactics, In *Proceedings of the Third International Conference on Educational Data Mining*, 2010, 201-210.
- TRIVEDI. S., PARDOS. Z. A., HEFFERNAN. N. T, 2011, (submitted) The Utility of Clustering in Prediction Tasks, In *The Seventh ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2011.
- TRIVEDI. S., PARDOS. Z. A., HEFFERNAN. N. T, 2011, (accepted) Clustering Students to Generate an Ensemble to Improve Standard Test Score Predictions, In *The Fifteenth International Conference on Artificial Intelligence in Education*, 2011.

Does Time Matter? Modeling the Effect of Time in Bayesian Knowledge Tracing

YUMENG QIU, YINGMEI QI, HANYUAN LU, ZACHARY A.

PARDOS, NEIL T. HEFFERNAN

Worcester Polytechnic Institute, USA

Intelligent tutoring systems that utilize Bayesian Knowledge Tracing (KT) have the ability to predict student performance well. However, models currently in use do not consider that a student performing on ITS may not be finishing their work in the same day. We looked at KT's predictions on student responses where a day or more had elapsed since the previous response and found that KT consistently over predicted these data points in particular. We made two hypotheses to explain the over prediction behavior: 1) the student forgot since the last time on the tutor and 2) the student made a mistake (or slipped) on that first question of the day. We developed two models; KT-Forget and KT-Slip, modifications on Knowledge Tracing, to represent these two hypotheses. We evaluated and compared the performance of the KT-slip, KT-Forget and regular KT model by calculating prediction residuals and Area Under Curve (AUC) on a Cognitive Tutor and ASSISTments dataset. The results showed that a significant improvement was obtained on the overall prediction by our KT-Forget model, suggesting that forgetting is the more likely cognitive explanation for the data and that there is a place for modeling forgetting, something that has not common practice in student modeling.

Key Words and Phrases: Bayesian Network, Knowledge Tracing, Intelligent Tutoring Systems, Data Mining, Model Evaluation

1. INTRODUCTION

The knowledge tracing model [Corbett & Anderson 1995] has been widely used to model student knowledge and learning over time. It assumes that each skill has two knowledge parameters, *prior* and *learn*; and two performance parameters, *slip* and *guess*. The *learn* parameter represents the probability that a student will transition between the unlearned and the learned state after each question. The *slip* parameter is the probability that a student who understands a skill can make a careless mistake and the *guess* parameter is the probability a student may answer correctly in spite of not knowing the skill. There is also a *forget* parameter; however, in standard knowledge tracing this is fixed at 0, which means that there is no forgetting happen in this model.

When using the standard Knowledge Tracing (KT) model, it is assumed that the students' probability of making the transition from the unlearned to the learned state is constant opportunities (or questions). Many researchers have proposed extensions to Bayesian Knowledge Tracing [Conati, Abigail, Gertner, VanLehn and Druzdzel 1997], however none have tried to incorporate how much time has elapsed between opportunities into the model. They all assume that student performance a minute later is the same as the next day. Nonetheless, ever since Ebbinghaus inaugurated the scientific study of memory [Ebbinghaus 1913], researchers have examined the manner in which memory performance declines with time or intervening events [Pavlik & Anderson 2005].

In the real world coming into class on a new day may result in a student forgetting the material or a higher probability of them slipping. By taking this real world fact into consideration, in this paper we look into how KT performs on each new day's responses. We define a new day's response as a response that occurred on a later calendar date than the student's previous response to a question of the same skill. We found that KT's new day error is far higher than same day error. A residual analysis showed that KT was

largely over predicting student performance on each new day response, the residual analysis is shown in Table I.

Based on the residual result (Table I), we made two hypotheses to explain this phenomenon; 1) that students may forget between days and 2) that students may slip when answering the first question on a new day. The slip hypothesis only affects the model's prediction of new day events while the forget hypothesis could affect prediction of subsequent responses since it hypothesizes a change in the latent of knowledge. We developed two new models based on Knowledge Tracing: a KT-Forget Model and a KT-Slip Model, where a new day variable is taken into account to affect either students' knowledge or performance. To implement this, we introduced a new split-parameter KT model, which allowed us to, for instance, learn a different *forget* parameter for new day opportunities than for same day but learn only a single learn rate parameter for each.

Table I. Knowledge Tracing residual analysis

Problem Set	Residual Same Day	Residual New Day
1	0.039803	-0.363268
2	-0.026765	-0.110578
3	0.088299	-0.076079
4	-0.014643	-0.117302
5	-0.003538	-0.062383
6	0.018866	-0.160024
7	0.009965	-0.109267
8	-0.049156	-0.169034
9	0.023225	0.032221
10	-0.029405	-0.010356
11	0.013791	-0.275969
12	0.082811	-0.054692
Average	0.012771	-0.123060

The structure of the paper is as follows: in section 2 we present the model designs that incorporate the time concept. The evaluations of the proposed models are the focus of section 3 and summarized in section 4. To conclude, we identify and discuss open areas of research for future work in section 5.

2. TIME MODEL DESIGN

When using the Knowledge Tracing model, it is assumed that the student's probability of making the transition from the unlearned to the learned state is not changing across opportunities, while in the real world there might be a time elapse since students' last opportunity on tutor. This fact assumes that there is a great possibility that a student's forgetting rate is not zero. The standard KT model assumes no probability of forgetting. Prior work has modeled forgetting between sessions in a lab but did not allow within-day learning to occur [Pardos, Heffernan, Ruiz and Beck 2008]. Alternatively, poor performance on a new day may also suggest that students may not actually be "forgetting" but instead, they might just be "slipping." We used Bayesian networks [Reye 2004] and Expectation Maximization [Dempster, Laird and Rubin 1977] to detect whether time had any influence on the *forget* parameter and the *slip* parameter of the KT model. The model with the better predictive accuracy will indicate the better cognitive explanation of the data.

2.1 Split-KT Model Design

In order to determine the validity of this method, we represent the above two hypothesis in the Bayesian Knowledge Tracing model by introducing a novel modification to the model that allows us to fit a same day and new day parameter for one parameter in a Conditional Probability Table (CPT) while keeping the other parameter in the CPT constant. In Knowledge tracing; *learn* and *forget* share a CPT and *guess* and *slip* share a CPT. As shown below, the difference between split-KT and the original-KT is the ability to separate the *forget*, *learn*, *guess*, and *slip* parameters individually. The equivalence between these two KT models was confirmed empirically by learning parameters for each model from a shared dataset, without new day data, and confirming that the learned parameters and predictions were the same. We also compared the computational run time of the Split-KT and Regular-KT. We tested on one of the dataset from ASSISTment which contained 527 data points and calculated the EM parameter learning time of the two models. It took approximately 70 seconds for the Regular-KT to learn the parameters and 102 seconds for Split-KT, which equates to a penalty of about 50% to run the Split-KT model. Both models resulted in the same learned parameters.

The individualization of the four parameters were achieved by adding a *forget* node and a *learn* node to the knowledge node, as well as adding a *guess* node and *slip* node to the question node. Therefore, the knowledge nodes and question nodes are conditioned upon the four new nodes. The CPT for knowledge node is given in Table II. The CPT for the question node is also of this form, the only difference is changing the *learn* and *forget* parameters to *guess* and *slip* parameters and changing the previous and current knowledge to previous and current student performance. The question and knowledge CPTs are fixed and essentially serve as logic gates. The *guess*, *slip*, *learn* and *forget* node CPTs contain the continuous probabilities that are familiar to the standard KT model. Take the first row as an example, knowing that the students do not have previous knowledge of the skill (*Knowledge_previous*=F), and they neither learn nor forget (*learn*=F, *forget*=F), then we can infer the probability that students have the current knowledge is 0 ($P(\text{Knowledge_current}=T) = 0$).

Table II. The CPT for Knowledge node

Learn	Forget	Knowledge_previous	$P(\text{Knowledge_current}=T)$
F	F	F	0
T	F	F	1
F	T	F	0
T	T	F	1
F	F	T	1
T	F	T	1
F	T	T	0
T	T	T	0

This model can easily let us set individualized learn rates, forget rates, guess rates and slip rates. By this way we are able to fix the learn parameter and guess parameter in order to investigate how new day instances would affect the *forget* and *slip* parameters.

2.2 The KT-Forget

In this section we focus on one of the hypothesis: how would the new day instance affect the forget parameter. We think that it is highly possible that students could be forgetting the previously learned knowledge when there are several days interval between the practices on the ITS.

The model we used to test our hypotheses is a new model built based on the Split-KT

model discussed in the previous section. By adding a time node to the Split-KT model we are able to easily specify which parameters of the model should be affected by a new day. The new day node is fixed with a prior probability of 0.2, which is the overall proportion of the new day instances in the dataset. The topology of the KT-Forget model is shown in Fig. 1. The forget node is only conditioned on the added new time node, so there is only one new parameter “*forget_n*” introduced in this KT-Forget model and represents the forget rate on a new day. We use “*forget_s*” to denote the forget rate on a same day, which we set to be 0 just as the *forget* parameter in the original Knowledge Tracing model implying that there is no forgetting between opportunities in the same day.

Table III. CPT of the forget node

New Day	P(Forget=T)
F	0
T	<i>forget_n</i>

The CPT for the forget node in this model is shown in Table III. This table says that when a new day response occurs, New Day=T, the probability that student forget knowledge is *forget_n*, $P(\text{Forget}=T|\text{New Day}=T)$ and is 0, otherwise.

2.3 KT-Slip model

An alternate hypothesis is that while students might be performing on the ITS across several days, they are not forgetting the previously learned material. Rather, the students are just making a mistake on the first question of the day (rustiness effect) after which they no longer slip at a higher than usual rate. So the low accuracy on first attempt on a new day might not be captured in the *forget* parameter, it could be that they just slipped and answered wrong. This explanation makes it quite necessary for us to look into the *slip* parameter.

The KT-Slip model is similar to the KT-Forget model and can be represented simply by connecting the time node to the slip node instead of connecting to the forget node as in the Forget model. The Slip model allows us to model the different slip rates of the new days and the same days. The Slip model is shown above in Fig. 1 in the bottom box.

Table IV. CPT of the slip node

New Day	P(slip=T)
F	<i>slip_s</i>
T	<i>slip_n</i>

Since the slip node is only conditioned on the added new time node, there is also one new parameter *slip_n* introduced in this KT-slip model, which represents the slip rate on a new day, and the original *slip* parameter is denoted as *slip_s* here, which is shown in Table IV. This table says that when a new day response occurs, New Day=T, the probability of slipping is *slip_n*, $P(\text{slip}=T|\text{New Day}=T)$ and is *slip_s*, otherwise

2.4 Topology of the models

The Split-KT model's topology is shown together with KT-Slip and KT-Forget in Fig. 1. Boxes in the figure denote the portions of the figure that are used in each model. While all models are shown in this figure so the relationship between them can be seen. When the models are run, they are run separately as a separate topology and not one big model.

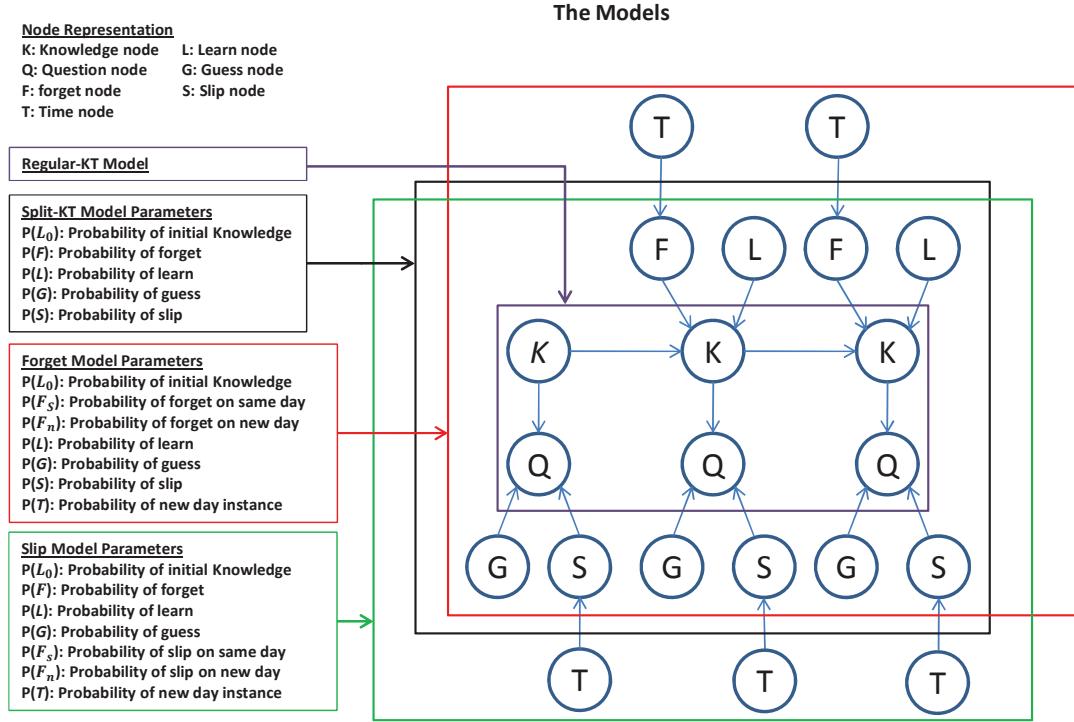


Fig. 1. The topology of the models – Split-KT, KT-Forget, KT-Slip

3 MODEL PERFORMANCE EVALUATIONS

To evaluate the performance of the KT-Forget and the KT-Slip models, we used a Cognitive Tutor dataset and ASSISTments dataset to test the real world utility of these models by comparing their predictive performance with a standard KT model. For each problem set, which represents a certain skill we trained regular KT, KT-Forget and KT-Slip models to make predictions on all the question responses of each student. Then the Residuals and AUC is calculated for predictions and actual responses on same day events, new day events as well as overall events to analyze the three models' performance. Residual is the mean of the actual performance subtracted by the predicted performance. AUC is a robust accuracy measure where a score of 0.50 represents a model that is only as good as chance and 1.0 represents a perfectly predicting model.

The analysis method consisted of two steps: run Expectation Maximization to fit the parameters on the training set for each model, and apply the trained parameters to the test sets to predict the student performance of each question.

3.1 Datasets for Prediction

One of the datasets comes from the Cognitive Tutor System called Bridge to Algebra and is from the 2006-2007 school year. This was one of the smaller, development datasets made public as part of the 2010 Knowledge Discover and Data mining competition [Pardos & Heffernan, In Press]. In this tutor, students answer algebra problems from their

math curriculum which is split into sections. The problems consist of many steps that the students must answer to go to the next problem. A student no longer needs to answer steps of a given skill when the Cognitive Tutor's Knowledge Tracing model believes the student knows the skill with probability 0.95 or greater. When a student has mastered all the skills in their current section they are allowed to move on to the next. The time for students using this system is determined by teachers. Twelve skills were chosen at random from this dataset for analysis (excluding skills such as "press enter" which do not represent math skills). There was an average of 122 student per skill in this dataset.

Another dataset is collected from ASSISTments Platform's Skill Builder problem sets. The ASSISTments Platform is an educational research platform better known for its e-learning [Feng, Heffernan, Mani and Heffernan, 2006] that provides web based math tutoring to 8th-10th grade students. Unlike the Cognitive Tutor System, students are forced to leave the tutor after 10 questions have been finished in one day and will come back to the tutor in a new day. If a student answers three questions correct in a row, they are "graduated" from the problem set. The help the tutorial provided is consist of a series of questions that break a problem in to sub steps. A student can also request a hint, but requesting a hint will mark the student as getting the step wrong in the system. Only answers to the original questions are considered. The largest twelve Skill Builder datasets were selected from the ASSISTments Platform. There was an average of 1,200 students per problem set in this dataset. The highest student count problem sets were selected here because new day events are far sparser in ASSISTments skill problem sets than the Cognitive Tutor skill problem sets.

The twelve datasets from each tutor were randomly divided into two equal parts by student, one part was used as the training set, the other as the testing set.

3.2 Prediction Procedure

Parameters were learned for each skill problem set individually. The parameters were unbounded and initial parameters were set to a *Guess* of 0.14, *Slip* of 0.09, *Prior* of 0.50 and *Learn* of 0.14, these initial values were the average parameter values across all skills in prior modeling work conducted on the ASSISTments tutor [Pardos, Heffernan, Ruiz and Beck, 2008]. Since both tutors, ASSISTments and Cognitive tutor, cover similar domains (algebra) and the initial parameter values are within plausible parameter ranges; we use the above parameters for both datasets.

For parameter learning, the new day observation (0 or 1) was presented as evidence in addition to the student responses. After training, the time and actual response values were given to the model as evidence for our new models to do the prediction (for regular KT, only actual responses were given as evidence) one student at a time. In order to predict every response of each student in the test set, the student data for prediction was presented to the network in the following fashion: for predicting the first question, no evidence was entered; for the second question, the new day information for that question and the actual response of first question were entered as evidence; for the third question, the first two new day information and responses information were entered as evidence. Apply this procedure until the prediction of the last question. This predicting process is shown in Fig. 2. By applying this prediction process, the probability of student answering each question correctly was computed and saved.

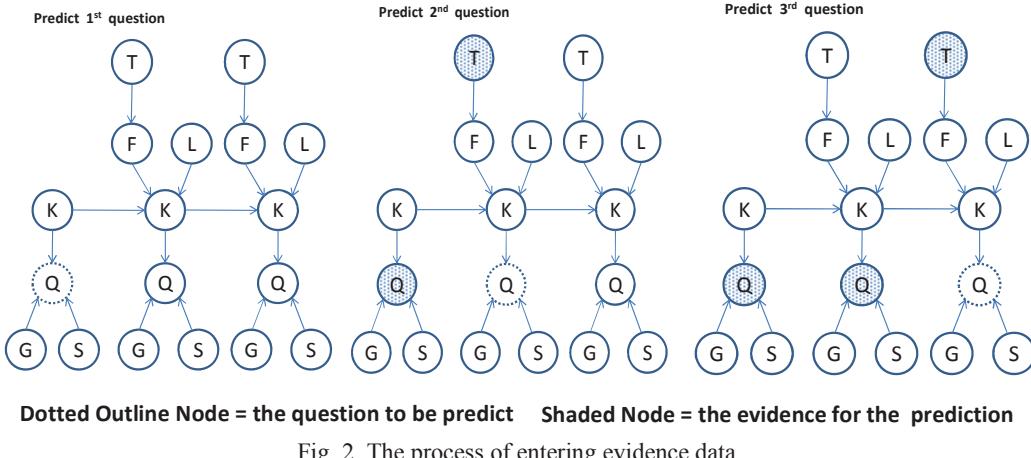


Fig. 2. The process of entering evidence data.

The results summary of all three models across problem sets as well as the results of pairwise t-test is shown in Table VII.

3.3 Prediction Result Analysis

The prediction performance of the three models were calculated in terms of Residuals and AUC values between predictions and actual responses on same day events, new day events as well as overall events of the whole problem set. The model with higher AUC values for a problem set was deemed to be the more accurate predictor of that problem set. In addition, a two-tailed paired t-test was calculated between KT and KT-Forget and KT and KT-Slip. We first applied this to the datasets collected from Cognitive Tutor. The specific results of each problem sets are shown below for regular KT (Table V) and KT-Forget (Table VI).

Table V. Residual and AUC results on Regular KT (Cognitive Tutor)

Regular KT	Residuals			AUC		
	Problem Set	Overall	Same Day	New Day	Overall	Same Day
1	-0.0263	0.0398	-0.3633	0.5952	0.6570	0.4972
2	-0.0390	-0.0268	-0.1106	0.7588	0.7434	0.8669
3	0.0623	0.0883	-0.0761	0.6496	0.6914	0.5656
4	-0.0272	-0.0146	-0.1173	0.7023	0.7324	0.6126
5	-0.0125	-0.0035	-0.0624	0.5822	0.5654	0.6728
6	0.0092	0.0189	-0.1600	0.7892	0.8171	0.6290
7	-0.0063	0.0100	-0.1093	0.6374	0.6446	0.6236
8	-0.0664	-0.0492	-0.1690	0.6936	0.7210	0.6003
9	0.0251	0.0232	0.0322	0.5384	0.5218	0.6278
10	-0.0267	-0.0294	-0.0104	0.6456	0.6204	0.7892
11	-0.0422	0.0138	-0.2760	0.4922	0.5176	0.5055
12	0.0483	0.0828	-0.0547	0.6149	0.6558	0.5129
Average	-0.0085	0.0128	-0.1231	0.6416	0.6573	0.6253

Table VI. Residual and AUC results on KT-Forget (Cognitive Tutor)

KT-forget	Residuals			AUC		
	Problem Set	Overall	Same Day	New Day	Overall	Same Day
1	-0.0121	0.0208	-0.1802	0.7765	0.7771	0.5238
2	-0.0103	-0.0037	-0.0484	0.7373	0.7183	0.8588
3	0.0755	0.0855	0.0223	0.7368	0.7497	0.5528
4	-0.0364	-0.0292	-0.0876	0.7262	0.7433	0.5938
5	-0.0045	-0.0022	-0.0174	0.6681	0.6080	0.7712
6	0.0095	0.0115	-0.0270	0.8331	0.8370	0.6399
7	0.0020	0.0116	-0.0587	0.6834	0.6857	0.6012
8	-0.0549	-0.0435	-0.1230	0.7209	0.7407	0.5805
9	0.0257	0.0165	0.0608	0.6070	0.6301	0.6768
10	-0.0162	-0.0246	0.0331	0.6115	0.6024	0.7746
11	-0.0414	-0.0118	-0.1645	0.6751	0.6376	0.6067
12	0.0445	0.0699	-0.0312	0.6278	0.6525	0.5133
Average	-0.0016	0.0084	-0.0518	0.7003	0.6985	0.6411

Table VII. Summary and T-test on Regular KT, KT-Forget and KT-Slip (Cognitive Tutor)

Model	Residuals (across problem sets)			AUC (across problem sets)		
	Overall	Same Day	New Day	Overall	Same Day	New Day
1. Regular KT	-0.0085	0.0128	-0.1231	0.6416	0.6573	0.6253
2. KT-forget	-0.0016	0.0084	-0.0518	0.7003	0.6985	0.6411
3. KT-slip	-0.0047	-0.0048	0.0017	0.6110	0.5917	0.5175
t-test (1,2)	0.0352	0.2697	0.0004	0.0129	0.0178	0.2445
t-test (1,3)	0.5149	0.0154	0.0017	0.1690	0.0017	0.0033

From the above results, generally, we can see that the new KT-Forget model performed better on both the residuals and AUC compared to the regular KT model. Inversely, the KT-Slip model performed worse than we expected. The specific evaluation of the two new models is shown in Table VII and Table VIII .

For the KT-Forget model, improved results were obtained both on residuals and AUC. Especially for the AUC, although KT-forget did not get significant improvement on new day events in terms of AUC (p value is 0.5175); however, it got significant improvement on same day events prediction and overall prediction (p value is 0.0178 and 0.0129), which means the performance of KT-Forget model is more accurate on predicting of Cognitive Tutor data compared to the regular KT model. Moreover, the better prediction performance also supported our hypothesis that students probably forget knowledge when it comes to a new day.

For the KT-Slip model, the results of overall data's AUC were worse but not

significantly compared to regular KT. However, both same day and new day AUC were significantly worse, which overthrew our assumption that students may slip when it comes to a new day.

Similarly, we applied our models to the ASSISTments datasets. The results of residuals and AUC across all problem sets are as below:

Table VIII. T-test on Regular KT, KT-forget and KT-slip (ASSISTments)

Model	Residuals (across problem sets)			AUC (across problem sets)		
	Overall	Same Day	New Day	Overall	Same Day	New Day
1. Regular KT	0.0019	-0.0019	0.0241	0.6719	0.6704	0.6364
2. KT-forget	-0.0036	-0.0129	0.0488	0.6678	0.6672	0.6366
3. KT-slip	-0.0105	-0.0240	0.0628	0.6486	0.6520	0.5981
t-test (1,2)	0.1449	0.0099	0.0001	0.1640	0.0885	0.9603
t-test (1,3)	0.0133	0.0003	0.0057	0.0085	0.0353	0.0057

From Table VIII, we can observe that the new models, both KT-Forget and KT-Slip lost to the regular KT model, especially on the AUC. We looked into the reason why our new models perform much worse and found that the way the data was collected lead to this result. As we mentioned in the previous section, students are forced to leave the tutor after a certain number of questions have been finished in one day and will come back to the tutor in a new day. Thus, we observed that the datasets collected from ASSISTments have much fewer new day events (average 1 per student) and is not as amenable to a time analysis as the Cognitive Tutor data which has many new days per student and students experience the new day more naturally. Therefore, the results obtained from Cognitive Tutor are more practical for this analysis.

4 CONTRIBUTIONS

This paper makes two contributions. First, we show assumptions made in Knowledge Tracing model, that student don't forget, is false. While this might not be terribly surprising, we identify a particular situation in which the standard KT model has systematic errors in predicting student performance, which is on new day responses.

Secondly, we present a model to account for this phenomenon which does a reliably better job of fitting student data in some datasets. This is significant as KT has proved itself to be a very effective model, difficult to improve upon. It is also noteworthy that KT is easily interpretable and it is beneficial to be able to have a clean model that fits easily into the Bayesian framework and inherits this interpretability. Our contribution is that researchers should pay attention to "time" and we have demonstrated a method that takes this into account and improves modeling performance.

5 DISCUSSIONS AND FUTURE WORK

In this work we attempt to model the time factor to better predict students' learning performance in intelligent tutoring systems. Due to our experiment results that new KT-forget model worked very well on Cognitive Tutor datasets while failed on ASSISTments Tutor datasets, we need to further investigate into the real reasons which caused this. Thus, we would like to know when using KT-forget model is not beneficial.

In this paper we only made two assumptions that the parameters "forget" and "slip" will be affected by time factor. We have not yet looked into the performance of other parameters that might be affected by time, for example: students may have a fresh mind and learn more on a new day, which means a new parameter "learn new day" should be modeled. Also, it is possible that "time" should connect to these two parameters at once.

It is also possible that the model can be improved by taking into account how many days have elapsed since last opportunity, . We will keep on delving into these possibilities to see whether further improvement incorporating time can be obtained. If this is achieved in future, we can build an ensemble model [Caruana, Niculescu-Mizil, Crew and Ksikes 2004] that combines regular KT's results on same day with the new model's results on new day.

Our work only focuses on whether students answer the questions in one day or in a new day, we do not pay attention to the intervals between same day and a new day. Pavlik and Anderson's [Pavlik & Anderson 2005] study showed that longer intervals should have a greater impact more on students' performance while shorter intervals may have very little effect on actual responses. These topics deserve further investigation to figure out how to leverage the valuable time information and build better user models.

ACKNOWLEDGEMENTS

This research was supported by the National Science foundation via grant "Graduates in K-12 Education" (GK-12) Fellowship, award number DGE0742503 and Neil Heffernan's CAREER grant. We would like to thank the organizers of the 2010 KDD Cup at the Pittsburgh Science of Learning Center for the Cognitive Tutor datasets and Matthew Dailey for his data preparation assistance.

We also acknowledge the many additional funders of ASSISTments Platform found here: <http://www.webcitation.org/5ym157Yfr>

REFERENCES

- Caruana, R., Niculescu-Mizil, A., Crew, G., Ksikes, A.: Ensemble selection from libraries of models. Proceedings of the 21st International Conference on Machine Learning, (2004)
- Conati, C., Abigail S. Gertner, VanLehn, K., Druzdzel, M.: On-Line Student Modeling for Coached Problem Solving Using Bayesian Networks (1997)
- Corbett, A.T. and Anderson, J.R.: Knowledge tracing: modeling the acquisition of procedural knowledge, pp. 253–278. User Modeling and User-Adapted Interaction 4, (1995)
- Corbett, A.T.: Cognitive computer tutors: Solving the two-sigma problem, pp 137-147. User Modeling: Proceedings of the Eighth International Conference, Sonthofen, Germany, UM (2001)
- Dempster, A.P., Laird, N.M., Rubin, D.B.: "Maximum Likelihood from Incomplete Data via the EM Algorithm". Journal of the Royal Statistical Society. Series B (Methodological) 39 (1): 1–38 (1977)
- Ebbinghaus, H. Memory: A Contribution to Experimental Psychology, translated in English, (1913)
- Feng, M., Heffernan, N. T., Mani, M., & Heffernan, C.: Using Mixed-Effects Modeling to Compare Different Grain-Sized Skill Models. In Beck, J., Aimeur, E., & Barnes, T. (Eds). Educational Data Mining: Papers from the AAAI Workshop. Menlo Park, CA: AAAI Press. pp. 57-66. Technical Report WS-06-05. ISBN 978-1-57735-287-7 (2006)
- Pardos, Z. A., Heffernan, N. T., Ruiz, C. & Beck, J.E.: Effective Skill Assessment Using Expectation Maximization in a Multi Network Temporal Bayesian Network. The Young Researchers Track at the 20th International Conference on Intelligent Tutoring Systems. Montreal, Canada, (2008)
- Pardos, Z.A., Heffernan, N. T.: Using HMMs and bagged decision trees to leverage rich features of user and skill from an intelligent tutoring system dataset. To appear in Journal of Machine Learning Research W & CP, (In Press)
- Pavlik, P. Jr., Anderson, J. R.: Practice and Forgetting Effects on Vocabulary Memory: An Activation-Based Model of the Spacing Effect, pp. 559–586. Cognitive Science 29 (2005)
- Reye, J.: Student modeling based on belief networks. International Journal of Artificial Intelligence in Education 14, 1-33. (2004)

Learning Classifiers from a Relational Database of Tutor Logs

JACK MOSTOW, JOSÉ GONZÁLEZ-BRENES, BAO HONG TAN
Carnegie Mellon University, United States

A bottleneck in mining tutor data is mapping heterogeneous event streams to feature vectors with which to train and test classifiers. To bypass the labor-intensive process of feature engineering, AutoCord learns classifiers directly from a relational database of events logged by a tutor. It searches through a space of classifiers represented as database queries, using a small set of heuristic operators. We show how AutoCord learns a classifier to predict whether a child will finish reading a story in Project LISTEN’s Reading Tutor. We compare it to a previously reported classifier that uses hand-engineered features. AutoCord has the potential to learn classifiers with less effort and greater accuracy.

1. INTRODUCTION

Intelligent tutors’ interactions with students consist of streams of tutorial events. Mining such data typically involves translating it into tables of feature vectors amenable to statistical analysis and classifier learning [Mostow and Beck, 2006]. The process of devising suitable features for this purpose is called feature engineering. Designing good features can require considerable knowledge of the domain, familiarity with the tutor, and effort. For example, manual feature engineering for a previous classification task [González-Brenes and Mostow, 2010] took approximately two months.

This paper presents AutoCord (Automatic Classifier Of Relational Data), an implemented system that bypasses the labor-intensive process of feature engineering by training classifiers directly on a relational database of events logged by a tutor. We illustrate AutoCord on data logged by Project LISTEN’s Reading Tutor, which listens to children read stories aloud, responds with spoken and graphical feedback [Mostow and Aist, 1999], and helps them learn to read [see, e.g., Mostow et al., 2003]. To illustrate AutoCord, we train a classifier to perform a previously published task [González-Brenes and Mostow, 2010]: predict whether a child who is reading a story will finish it.

The rest of the paper is organized as follows. Section 2 describes how we represent event patterns. Section 3 explains how AutoCord discovers classifiers. Section 4 evaluates AutoCord. Section 5 relates AutoCord to prior work. Section 6 concludes.

2. REPRESENTATION OF EVENTS, CONTEXTS, AND PATTERNS

We now summarize how we log, display, generalize, and constrain Reading Tutor events.

2.1 *The structure of data logged by the Reading Tutor*

The events logged by the Reading Tutor vary in grain size. As Figure 1 illustrates, logged events range all the way from an entire run of the program, to a student session, to reading a story, to encountering a sentence, to producing an utterance, down to individual spoken words and mouse clicks. Figure 1 shows a screenshot of the Session Browser

Authors’ addresses: J. Mostow, RI-NSH 4103, 5000 Forbes Avenue, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213-3890, United States; email: mostow@cs.cmu.edu; J. González-Brenes; email: joseg@cs.cmu.edu; B.H. Tan; email: btan@andrew.cmu.edu. This work was supported in part 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 the U.S. Department of Education. We thank the educators, students, and LISTENers who generated our data.

[Mostow et al., 2010], which displays logged Reading Tutor data in human-readable, interactively expandable form.

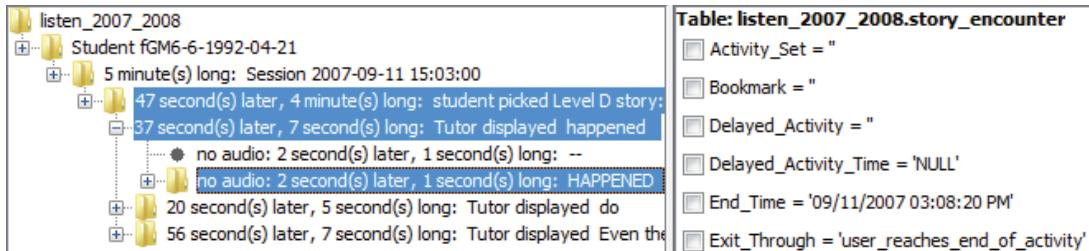


Figure 1: Session Browser’s partially expanded event tree (left); partial record for the highlighted story_encounter event (right).

Figure 1 displays a story encounter in the temporal hierarchy of the session in which it occurred. Each line summarizes the database record for an event. The highlighted story encounter “...student picked Level D story...” is represented as a row in the story_encounter table, with the field names and values listed on the right side of Figure 1. For example, the Exit_through field is a label that shows how the story encounter ended, and its value user_reaches_end_of_activity indicates that the student finished the story, so the story encounter is a positive example of story completion. All other values indicate different outcomes, such as clicking *Back* or *Goodbye*, timing out, or crashing.

The fields User_ID, Machine_Name, Start_time, and Sms are common to all types of events, including story encounters and sentence encounters. As their names suggest, they respectively identify the student and computer involved in the event, and when it started, with the milliseconds portion in its own field. Events with non-zero durations also have corresponding End_time and Ems fields.

Here the user has partially expanded the tree of events by clicking on some “+” icons. The structure of the tree indicates parental and fraternal temporal relations among events. A child event is defined as starting during its parent event; siblings share the same parent. The indentation level of each event reflects these relations. For instance, the highlighted story encounter is a child of the session summarized on the preceding line, and is therefore indented further. The story encounter’s children are the sentence encounters shown below it, displayed at the same indentation level because they are siblings.

2.2 Inferring a pattern from a set of related events

In Figure 1, the user has selected the highlighted events by clicking on them with the CTRL key down. Given such a constellation of related events, the Session Browser’s AutoJoin operator [Mostow and Tan, 2010] generalizes it into a pattern of which it is an instance. To infer a pattern from a single instance, AutoJoin heuristically assumes that repetition of a constant unlikely to recur by coincidence, such as a user ID, is a requirement of the pattern. AutoJoin represents the inferred pattern as a MySQL query [MySQL, 2004] that can retrieve instances of the pattern. An example of such a query is:

```
SELECT * FROM
    utterance u,
    story_encounter st,
    sentence_encounter se
WHERE
    (st.Machine_Name = se.Machine_Name) AND
    (st.Start_Time = se.Story_Encounter_Start_Time) AND
    (st.User_ID = se.User_ID) AND
    (st.Start_Time = se.Start_Time) AND
```

Identify sentence
encounter as part of
the story encounter.
Ensure it is the first
sentence encounter.

(st.Machine_Name = u.Machine_Name) AND
 (st.User_ID = u.User_ID) AND
 (se.sms = u.Sentence_Encounter_sms) AND
 (st.Start_Time = u.Sentence_Encounter_Start_Time) AND
 (se.End_Time = u.End_Time)

Identify utterance as part of the sentence encounter.
 Ensure it is the last utterance of the sentence encounter.

2.3 Operability criteria for learned queries

Given a target concept such as “stories the student will finish reading,” AutoCord searches for queries that maximize the number of positive instances retrieved and minimize the number of negative instances. In addition, the query must satisfy operability criteria [Mostow, 1983] that constrain the information used in the query. These constraints vary in form and purpose.

One type of operability constraint limits the query to information available at the point in time where the classifier will be used. For instance, a story encounter’s End_Time field tells us when the encounter ends, but obviously the Reading Tutor can only log this information once the encounter actually ends, so the trained classifier cannot use it to help predict whether a child will finish a story. Similarly, we use the Exit_through field of a story encounter to label it as a positive or negative example of story completion, but the trained classifier cannot use it to make predictions, since that information is only available once the encounter ends. As Yogi Berra famously said, “It’s hard to make predictions, especially about the future.” More subtly, if we want to use the trained classifier a specified time interval after a story encounter starts, we should train and test it on data representative of what will be available then. To simulate such data, we restrict the training and test sets to story encounters lasting at least this long, and we exclude events logged after this amount of time elapsed since the story encounter started. We implement these constraints by adding the following two clauses to a query:

... AND (UNIX_TIMESTAMP(st.End_Time) – UNIX_TIMESTAMP(st.Start_Time) >= [*limit*])

AND (se.Start_Time <= DATE_ADD(st.Start_Time, INTERVAL [*limit*] SECOND))

Here, [*limit*] is the time limit in seconds, say 10. Then the training and test sets include only story encounters that lasted at least 10 seconds, and the training and test procedures can only consider events that occurred within these story encounters’ first 10 seconds.

Operability criteria may also restrict what sort of classifier is useful to learn. For instance, to apply to future data, we may not want the trained classifier to be specific to any particular student or computer. We enforce this constraint by excluding user IDs and machine names from the query. Similarly, if we want the classifier to predict story completion based solely on the student’s observed behavior rather than traits such as age or gender, we exclude those fields from the query.

Finally, operability criteria may pertain to the protocol for training and testing the classifier. Even if we preclude the trained classifier from mentioning specific students, it may still implicitly exploit information about them, improving classification performance on the training set – and inflating performance on a test set that includes the same students. To ensure that the training and test sets have no students in common, the queries that generate them include mutually exclusive constraints on the user_id, e.g.:

(st.User_ID <= 'mDS8-8-1998-09-22') /* Use training set */
 or

(st.User_ID > 'mDS8-8-1998-09-22') /* Use test set */

Although these clauses mention a specific user_ID, despite the constraint against doing so, we do not consider them part of the learned classifier itself, just a way to split the data

into training and test sets. We could use a more complex constraint to implement a more sophisticated split, e.g. to stratify by gender, encoded by the first letter of the user_ID.

3. APPROACH

We formulate AutoCord as a heuristic search through a space of classifiers represented as database queries. Section 3.1 outlines the overall search algorithm. Section 3.2 describes the search operators.

3.1 Search Algorithm

AutoCord searches through a space of classifiers by hill climbing on their accuracy. In the pseudo-code below, step 1 starts with a query to retrieve the entire training or test set.

Pseudo-code for AutoCord(initial query)

1. $Q \leftarrow$ initial query
2. $S \leftarrow$ empty set; $K_{best} \leftarrow 0$
3. $R \leftarrow$ table of results (examples) retrieved by executing query Q
4. For each operator Op:
5. $Q' \leftarrow Op(Q, R)$
6. $R' \leftarrow$ table of results retrieved by Q'
7. $K \leftarrow Score(R')$
8. Add tuple (Q', K) to S
- End For
9. Pick (Q_{high}, K_{high}) from S that maximizes K_{high}
10. If $K_{high} < K_{best} + \text{epsilon}$, return Q_{high}
11. $Q \leftarrow Q_{high}$; $K_{best} \leftarrow \text{Max}(K_{best}, K_{high})$
12. Go to step 3.

For the task of predicting story completion, we start with this initial query:

```
SELECT * FROM story_encounter st
WHERE (st.User_ID <= 'mDS8-8-1998-09-22') /* Use training set */
AND (UNIX_TIMESTAMP(st.End_Time) - UNIX_TIMESTAMP(st.Start_Time) >= [limit])
```

Other classification problems would require a different initial query.

Steps 3 through 13 specify an iterative process. Step 3 retrieves a table of results R from the database by executing the current query Q . Next, the loop starting at step 4 applies each of AutoCord's operators. Based on the result set R , each operator adds one or more constraints to the input query Q to generate a new query Q' . Step 6 executes the new query Q' to get a new table of results R' . Step 7 scores classification accuracy as the number of positive examples in R minus the number of negative examples. Step 8 records query Q' and its score K . Step 9 chooses the highest-scoring query so far for the next iteration of the iterative step. The higher this score, the larger the number of positive examples the query retrieves, and the smaller the number of negative examples. Recall that the Exit_through field provides the label for a retrieved example. We score queries by the difference of these numbers rather than their ratio in order to reward recall as well as precision. Unless the query enlarges this difference by more than epsilon (currently 2), step 10 stops and returns it. Otherwise search continues from the best query found so far.

The query can be applied as a classifier when a child is reading a story. Events that occurred up to the point in time the query is applied form a partial event tree which could be used to check against the constraints specified in the query. If all of the constraints are satisfied, then the label for the current story will be positive; otherwise, the label will be negative. To train a query representative of negative examples, we can re-interpret the

value of the Exit_through field of the story encounters in the training set. When we consider story completion as being positive, we interpret a value of user_reaches_end_of_activity as a positive label, and all other values as negative labels. If we now consider quitting as being positive instead, we could interpret the value user_reaches_end_of_activity as a negative label. In this way, we could train queries representative of quitting. It is possible for the same example to have multiple labels if it is checked against more than one classifier, each of which represents a different category. An evaluation metric for accuracy would need to penalize such cases appropriately.

Next, we describe AutoCord's operators. To illustrate them, we do a walkthrough of the search algorithm, starting from the following initial query:

```
SELECT * FROM story_encounter st /* st is alias for story_encounter */
WHERE (st.User_ID <= 'mDS8-8-1998-09-22') /* Use training set */
```

3.2 Contrast Operator

The Contrast operator adds a single constraint that best distinguishes positive from negative examples. It generates this constraint based on a split in the distribution of values for a field. For example, if all positive examples have values below 5 for a particular field, and if all negative examples have values above 5 for that field, then the split value 5 perfectly separates positive from negative examples. To find the field that can provide the best split, AutoCord calculates the frequencies of values for each column of the results table retrieved by the initial query. It computes two sets of frequencies – one for positive examples and another for negative examples. To illustrate, consider the following results table:

Row #	New_Word_Count	Initiative	Student_Level	...
1	4	Student	A	
2	6	Student	C	
3	7	Student	C	

Figure 2: An example of a table of results retrieved

All the fields come from the story_encounter table, and each row represents a story encounter. The rest of the fields are omitted for brevity. Assume the first two rows are positive examples, while the third is a negative example. The calculated frequencies are:

	New_Word_Count	Initiative	Student_Level
Positive	4: once, 6: once	Student: twice	A: once, C: once	
Negative	7: once	Student: once	C: once	

In this case, the Contrast operator finds that the best split occurs in the New_Word_Count field, with a split value of 6. Thus it adds the new constraint New_Word_Count <= '6' since only the positive examples satisfy this constraint. However, in general, when it is not possible to find a perfect split, the operator will choose one that separates as many positive examples as possible from the negative examples. The Contrast operator considers the mathematical relations =, !=, <, <=, >, and >=.

3.3 Extend Operator

The Extend operator essentially captures the relational structure of positive examples. To do so, it first picks a random positive example (which is a row) from the results table. Recall that a row in the results table represents a collection of events. Next it randomly

picks an event in the chosen row. For that event, it will then either pick a random child, sibling, or parent event. With the existing events in the input query and the newly picked event, the Extend operator then applies AutoJoin and adds the resulting constraints to the input query.

We illustrate the Extend operator on the initial query shown at the end of Section 3.1. This query only represents story encounters, so it retrieves a table of results where each row represents a single story encounter. Suppose the Extend operator randomly picks one such `story_encounter` and then one of its children, namely a `sentence_encounter`. Applying AutoJoin to these two events might yield this query:

```
SELECT * FROM story_encounter st, sentence_encounter se
WHERE
    (st.User_ID <= 'mDS8-8-1998-09-22') /* Use training set */
AND  (st.Machine_Name = se.Machine_Name) /* Added by Extend operator */
AND  (st.User_ID = se.User_ID)
AND  (st.Start_Time = se.Story_Encounter_Start_Time)
```

AutoJoin adds the last three constraints because both events have the same values for the fields `Machine_Name`, `User_ID`, and `Start_Time`.

3.4 Aggregate Operator

The Aggregate operator generates additional pseudo-fields for the Contrast operator to work on. The pseudo-fields of an event refer to the aggregated fields of the event's children. We shall illustrate the idea of pseudo-fields using the figure below.

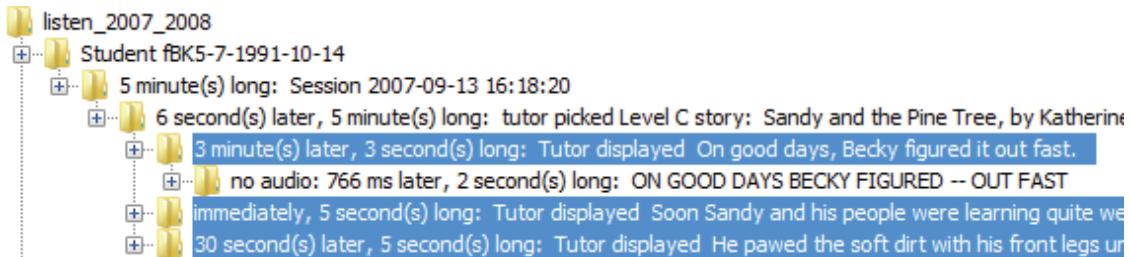


Figure 3: The highlighted sentence_encounter events of the story_encounter event.

Figure 3 highlights the three children of the `story_encounter` event “6 second(s) later, 5 minute(s) long: tutor ...”. These children are `sentence_encounter` events. As its name suggests, the Aggregate operator aggregates the values of each `sentence_encounter` field over these children and adds them as pseudo-fields of the `story_encounter` event to provide additional information about it. For instance, the aggregated field `AVG(se.Word_Count)`, where “`se`” refers to each `sentence_encounter`, represents the average word count of the sentences in a `story_encounter`, reflecting its reading level.

For efficiency reasons, AutoCord precomputes the aggregated fields for all events in the training set before the search starts and stores them in a separate temporary table for each parent-child relation and specified time limit. The following example query calculates the table for the `story_encounter/sentence_encounter` relation:

```
CREATE TEMPORARY TABLE `story_encounter-sentence_encounter_agg` AS
SELECT st.*, AVG(se.Word_Count) AS _AVG_Word_Count,
[other aggregated fields...]
FROM
    story_encounter st,
    sentence_encounter se
WHERE st.Machine_Name = se.Machine_Name
```

```

AND st.User_ID = se.User_ID
AND st.Start_Time = se.Story_Encounter_Start_Time
AND se.Start_Time <= DATE_ADD(st.Start_Time, INTERVAL [limit] SECOND)
AND UNIX_TIMESTAMP(st.End_Time) - UNIX_TIMESTAMP(st.Start_Time) >= [limit]
GROUP BY st.Machine_Name, st.User_ID, st.Start_Time, st.sms

```

Recall that the last two constraints impose the time limit operability criterion. Also note that the GROUP BY clause is necessary for the aggregation to work correctly. Currently, AutoCord supports only the MIN, MAX, AVG, SUM, COUNT, and STDDEV aggregator functions, and only on numeric-valued fields, except for COUNT, which simply counts the number of rows it aggregates over. It applies each aggregator function to every field of the child event, as indicated by *[other aggregated fields...]*.

Using an aggregated field in a constraint requires a join to the temporary table, e.g.:

```

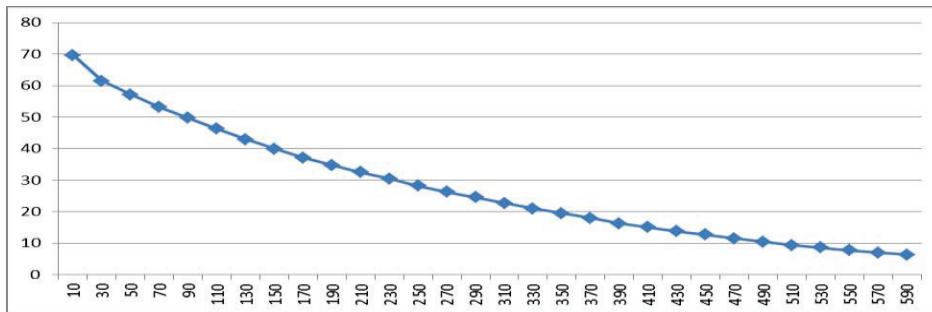
... FROM
    story_encounter st,
    `story_encounter-sentence_encounter_agg` `st-sentence_encounter_agg`
WHERE (st.User_ID <= 'mDS8-8-1998-09-22') /* Use training set */
AND (UNIX_TIMESTAMP (st.End_Time) – UNIX_TIMESTAMP(st.Start_Time) >= [limit])
AND (st.Machine_Name = `st-sentence_encounter_agg`.Machine_Name)
AND (st.User_ID = `st-sentence_encounter_agg`.User_ID)
AND (st.Start_Time = `st-sentence_encounter_agg`.Start_Time)
AND (st.sms = `st-sentence_encounter_agg`.sms)
AND (`st-sentence_encounter_agg`._STDDEV_Word_Count` >= '0.4')

```

The last constraint, added by the Contrast operator, selects story encounters whose sentence lengths vary enough to have standard deviation of at least 0.4. Such variation might make stories more interesting, or simply reflect harder stories read by better readers likelier to complete them.

4. EVALUATION

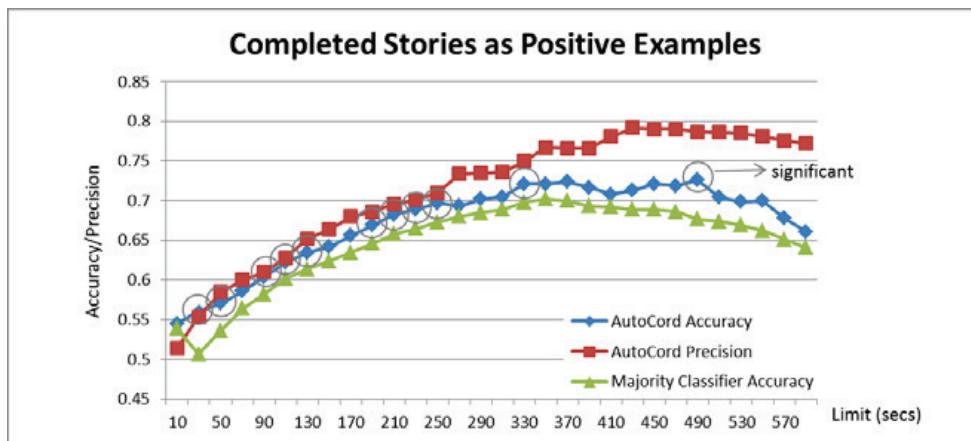
Section 2.3 discussed how to restrict the amount of information the search algorithm can look at for each story encounter in the labeled training set. In this way, the algorithm can only learn from events available from the start of the story encounter up to the specified time limit. In other words, the algorithm cannot “peek into the future” of a story encounter. Imposing a time limit also provides a means to test the classifier’s ability to predict the outcome of a story encounter at various points in time before the story ends.



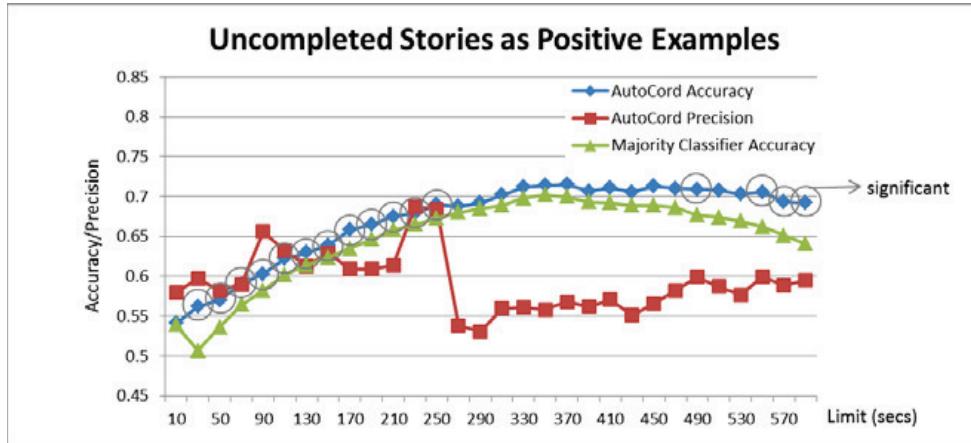
We modified the search algorithm slightly to restricting the information available. More specifically, we modified the Extend operator so that whenever it adds a new event to the current query, the new event start time starts before the specified time limit. We similarly constrained the Aggregate operator to include only such child events too. We ran the modified search algorithm for various time limits ranging from 10 to 590 seconds

in increments of 20 seconds, which corresponds roughly to the average duration of a sentence encounter. After each run, we got a query suitable for the specified time limit. The graph above shows the percentage of story encounters in the test set (shown on the y-axis) that lasted at least a certain number of seconds (shown on the x-axis).

We executed the queries generated by the algorithm, one by one, and for each one, calculated its accuracy and precision. It is not meaningful to compare the values of these two metrics, but to save space we plot them both on the y-axis of the same graph below against the time limit in seconds on the x-axis. We include majority class accuracy as a baseline for comparison. The majority classifier always outputs the label assigned to the majority of the story encounters in the training set, so its accuracy for a specified time limit is simply the percentage of story encounters with that label in the test set for that limit. We circle the points where the difference in classification accuracy between the trained query and majority class is statistically significant at $p < .05$. To account conservatively for statistical dependencies among data points from the same student, we test whether this difference exceeds zero by more than a 95% confidence interval defined as twice the weighted standard error of the per-student difference, weighting by the number of data points per student.



For comparison, we trained two types of queries, one with completed stories as positive examples and the other with uncompleted stories as positive examples. The graph below shows the corresponding accuracy and precision when uncompleted stories are treated as positive examples. Accuracy is similar, but precision is less consistent.



González-Brenes and Mostow [2010] applied ℓ_1 -regularized logistic regression to the same classification task, but their results are not directly comparable because they framed

it differently. They expressed their time limit as a number of sentence encounters before a story encounter ended, rather than as a number of seconds after it started.

5. RELATION TO PRIOR WORK

Mining relational data sits at the intersection of Machine Learning with classical Artificial Intelligence methods that rely on formal logic, an area called Inductive Logic Programming (ILP). Notable examples of ILP algorithms that learn from data expressed as relations using formal logic representations include FOIL [Quinlan, 1990] and Progol [Muggleton, 1995]. Like FOIL, AutoCord inputs positive and negative examples in relational format, and hill-climbs to distinguish between classes. FOIL uses negation and conjunction operators and outputs Horn clauses, whereas AutoCord uses the logical conjunction AND to combine all constraints. It uses negation only to negate the equality relation in the Contrast operator, not for an entire constraint. Also, AutoCord assumes that relations describe events, works on SQL queries directly, and outputs SQL queries.

ILP methods can sometimes achieve high classification accuracy [Cohen, 1995], but are sensitive to noise [Brunk and Pazzani, 1991], and fail to scale to real-life database systems with many relations [Yin et al., 2006]. In contrast, AutoCord’s direct use of SQL queries enables it to operate directly on large event databases thanks to efficient retrieval from suitably indexed tables of events.

Provost and Kolluri [1999] reviewed literature on how to scale ILP approaches. They suggested that integrating data mining with relational databases might take advantage of the storage efficiencies of relational representations and indices. We believe AutoCord is the first ILP system to learn a classifier from databases by operating directly in SQL.

Other approaches to scale relational learning include CrossMine [Yin et al., 2006], which reduces the number of relations by using a “virtual join” in which the tuple IDs of the target relation are attached to the tuples of a non-target relation. CrossMine employs selective sampling to achieve high efficiency on databases with complex schemas. In contrast, AutoCord operates on all training data available to eliminate sampling bias.

A more recent perspective on ILP, Relational Mining, focuses on modeling relational dependencies. For example, it has been used to classify and cluster hypertexts, taking advantage of their relational links between instances [Slattery and Craven, 1998]. AutoCord’s Extend operator also exploits relational links between events.

Modeling the database without an explicit feature vector contrasts with work that uses feature induction. For example, a feature vector can be expanded using conjunction operators to improve accuracy [McCallum, 2003]. Alternatively, Popescul and Ungar [Popescul, 2004] proposed modifying SQL queries systematically, which is similar to what AutoCord does, but their method involved generating cluster IDs that can be used as features in logistic regression.

6. CONCLUSION

This paper proposes, implements, and tests an automated process for training classifiers on relational data logged by an intelligent tutor. Unlike many machine learning techniques, it does not require defining a feature vector first. Future work includes:

Evaluating on more tasks: So far we have applied AutoCord only to predicting story completion. We need to evaluate it on other classification tasks, such as characterizing children’s behavior according to whether they or the Reading Tutor picked the story [González-Brenes and Mostow, 2010], or what events tend to precede a software crash.

Adding more operators: For example, event duration is useful for predicting story completion [González-Brenes and Mostow, 2010], but is not an explicit database field. To address this limitation, a Derive operator would compute simple combinations of existing fields, e.g., end_time – start_time, to use as additional fields.

Combining queries: Due to the Extend operator's nondeterministic nature, different runs of AutoCord can generate different queries, varying in the information they use and the classification accuracy they achieve. Picking the best one or combining them into an ensemble of classifiers could improve accuracy.

Operationality criteria: AutoCord enforces specific operationality criteria *ad hoc* by adding clauses to the query or by excluding particular fields or constants from it. Future work might invent a general way to express operationality criteria in machine-understandable form and translate them into enforcement mechanisms automatically.

Generalizing to other tutors: AutoCord relies on the schema of the Reading Tutor database for reasons of efficiency and expedience rather than due to intrinsic limitations. Moreover, although its implementation uses MySQL, its method should apply to any relational database system. Generalizing AutoCord to apply to similarly structured data from other tutors would multiply its potential impact.

REFERENCES

- BRUNK, C.A. and PAZZANI, M.J. 1991. An investigation of noise-tolerant relational concept learning algorithms. In *Proceedings of the Eighth International Workshop on Machine Learning*, Evanston, IL, 1991, 389–393.
- COHEN, W. 1995. Learning to classify English text with ILP methods. *Advances in inductive logic programming*, 124–143.
- GONZÁLEZ-BRENES, J.P. and MOSTOW, J. 2010. Predicting Task Completion from Rich but Scarce Data. In *Proceedings of the 3rd International Conference on Educational Data Mining*, Pittsburgh, PA, June 11-13, 2010, R.S.J.D. BAKER, A. MERCERON and P.I.J. PAVLIK, Eds., 291-292.
- MOSTOW, D.J. 1983. Machine transformation of advice into a heuristic search procedure. In *Machine Learning*, R.S. MICHALSKI, J.G. CARBONELL and T.M. MITCHELL, Eds. Tioga, Palo Alto, CA, 367-403.
- MOSTOW, J. and AIST, G. 1999. Giving help and praise in a reading tutor with imperfect listening -- because automated speech recognition means never being able to say you're certain. *CALICO Journal* 16(3), 407-424.
- MOSTOW, J., AIST, G., BURKHEAD, P., CORBETT, A., CUNEO, A., EITELMAN, S., HUANG, C., JUNKER, B., SKLAR, M.B. and TOBIN, B. 2003. Evaluation of an automated Reading Tutor that listens: Comparison to human tutoring and classroom instruction. *Journal of Educational Computing Research* 29(1), 61-117.
- MOSTOW, J. and BECK, J.E. 2006. Some useful tactics to modify, map, and mine data from intelligent tutors. *Natural Language Engineering (Special Issue on Educational Applications)* 12(2), 195-208.
- MOSTOW, J., BECK, J.E., CUNEO, A., GOUVEA, E., HEINER, C. and JUAREZ, O. 2010. Lessons from Project LISTEN's Session Browser. In *Handbook of Educational Data Mining*, C. ROMERO, S. VENTURA, S.R. VIOLA, M. PECHENIZKIY and R.S.J.D. BAKER, Eds. CRC Press, Taylor & Francis Group, New York, 389-416.
- MOSTOW, J. and TAN, B.H.L. 2010. AutoJoin: Generalizing an Example into an EDM query. In *Proceedings of the 3rd International Conference on Educational Data Mining*, Pittsburgh, PA, June 11-13, 2010, R.S.J.D. BAKER, A. MERCERON and P.I.J. PAVLIK, Eds., 307-308.
- MUGGLETON, S. 1995. Inverse entailment and proglol. *New Generation Computing* 13(3), 245-286.
- MYSQL 2004. Online MySQL Documentation at <http://dev.mysql.com/doc/mysql>.
- PROVOST, F. and KOLLURI, V. 1999. A Survey of Methods for Scaling Up Inductive Algorithms. *Data Mining and Knowledge Discovery* 3(2), 131-169.
- QUINLAN, J.R. 1990. Learning logical definitions from relations. *Machine Learning* 5(3), 239-266.
- SLATTERY, S. and CRAVEN, M. 1998. Combining statistical and relational methods for learning in hypertext domains. *Inductive Logic Programming*, 38-52.
- YIN, X., HAN, J., YANG, J. and YU, P. 2006. CrossMine: Efficient Classification Across Multiple Database Relations. In *Constraint-Based Mining and Inductive Databases*. Lecture Notes in Computer Science, J.-F. BOULICAUT, L. DE RAEDT and H. MANNILA, Eds. Springer Berlin / Heidelberg, 172-195.

A Framework for Capturing Distinguishing User Interaction Behaviours in Novel Interfaces

S. KARDAN, C. CONATI
University of British Columbia, Canada

As novel forms of educational software continue to be created, it is often difficult to understand a priori which ensemble of interaction behaviours is conducive to learning. In this paper, we describe a user modeling framework that relies on interaction logs to identify different types of learners, as well as their characteristic interaction behaviours and how these behaviours relate to learning. This information is then used to classify new learners, with the long term goal of providing adaptive interaction support when behaviours detrimental to learning are detected. In previous research, we described a proof-of-concept version of this user modeling approach, based on unsupervised clustering and class association rules. In this paper, we describe and evaluate an improved version, implemented in a comprehensive user-modeling framework that streamlines the application of the various phases of the modeling process.

Key Words and Phrases: Student Modeling, Clustering, Associative Rule Mining

1. INTRODUCTION

Advances in HCI continuously aid the creation of novel interfaces to support education and training. Because of the novelty of these interfaces, it can be difficult to judge a priori which ensemble of user interaction behaviours are conducive to learning. Our long-term goal is to devise automatic techniques to analyze logs of the interactions with a novel application and identify classes of user types, their identifying behaviours and how these behaviours relate to learning. In addition, we want to use this information to create a user model, i.e., to automatically identify the behaviours of new users, and enable the application to provide adaptive support during interaction if the behaviours are associated with suboptimal task performance.

In previous work, we described a proof-of-concept user modeling approach that uses unsupervised clustering and class association rules to identify relevant user types/behaviours from an existing dataset, and relies on these to classify new users. In this paper, we refine that proof-of-concept into a comprehensive user-modeling framework that streamlines the phases necessary to generate a user classifier from an initial dataset of raw interaction logs. In [1] the initial approach was evaluated on an environment to support learning of AI algorithms via the exploration of interactive simulations. Here, we evaluate the new user modeling framework on the same environment but on a larger dataset (65 students vs. 24), thus providing more convincing evidence on the approach effectiveness.

After discussing related work, we illustrate the general user modeling approach, including improvements from previous versions. Next, we discuss an empirical evaluation of the framework and conclude with a discussion of future work.

2. RELATED WORK

Association rules have been widely used for off-line analysis of learners' interaction patterns with educational software. e.g., to discover (*i*) error patterns that can help improve the teaching of SQL [14]; (*ii*) similarities among exercises for algebra problem solving in terms of solution difficulty [6]; (*iii*) usage patterns relevant for revising a web based educational system spanning a complete university course [7].

Most work on using association rules for on-line adaptation has been done within research on recommender systems. In [4], for instance, association rule mining is used to match the user type with appropriate products. The main difference with our work is that in [4] there is no on-line classification. Users are “labelled” based on clusters built off-line and the labels are used to guide recommendations when these users utilize the system. In contrast, we perform online classification of new users, with the goal of eventually providing real-time adaptation. Similarly, associative classification is used in [20] to classify user requirements and generate personalized item recommendation in an e-commerce application. The main difference with our work is that the approach in [20] needs labelled data, while ours can work with unlabelled datasets.

The work by Romero et al ([16]) is the most similar to the research described here, in that the authors aim to use clustering and sequential pattern mining to recognize how students navigate through a web-based learning environment, classify them and use some teacher tuned rules for recommending further navigation links accordingly. The evaluation of this work focused on analyzing the quality of the rules generated by different algorithms, but no results have yet been presented on the classification accuracy of the proposed approach.

3. GENERAL USER MODELING FRAMEWORK

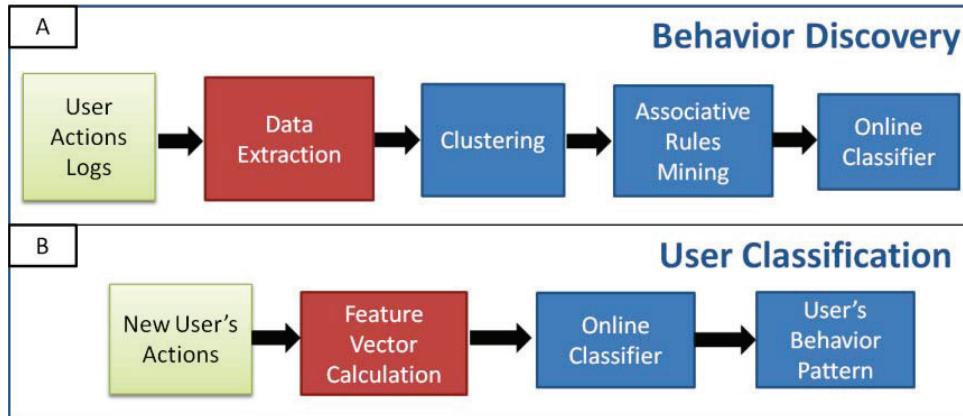


Figure 1: general User Modeling Approach.

Our user modeling approach consists of major phases: *Behaviour Discovery* (Figure 1A) and *User Classification* (Figure 1B). In Behaviour Discovery, raw unlabeled data from interaction logs is preprocessed into feature vectors representing individual users in terms of their interface usage. These vectors are the input to an unsupervised clustering algorithm that groups them according to their similarity. The resulting clusters represent users who interact similarly with the interface. These clusters are then analyzed to (*i*) identify if/how they relate to learning and then (*ii*) isolate in each cluster those behaviours that are responsible for this performance. In [3] we introduced the use of Class Association Rules [18] to identify the interaction behaviour characteristics of each cluster.

Understanding the effectiveness of a user’s interaction behaviours is useful in itself for revealing to developers how the application can be improved e.g. [10]. However, we also want to use these behaviours to guide automated adaptive support during interaction. Thus, the clusters and behaviours identified in the Behaviour Discovery phase are used to build an on-line classifier user model. In the User Classification phase (Figure 1B), this classifier is used to assess the performance of a new user based on her interaction behaviours. This assessment will eventually guide adaptive interventions that encourage effective interaction behaviours and prevent detrimental ones.

To test this approach, we generated a proof-of-concept version based on off-the shelf components and simplistic parameter settings. Following the encouraging results we obtained with this initial version [1], we have refined all framework components and implemented them in a Python-based unifying framework that streamlines the application of the various phases of the user modeling process. In the next few sections, we describe the most salient improvements we have made to the framework.

3.1 DATA EXTRACTION

The first step in *behaviour discovery* phase is to create a set of data-points from user interaction logs. Currently, our data-points are vectors of features consisting of statistical measures that summarize the user's actions in the interfaces (e.g. action frequencies; time interval between actions). Another approach is to create data-points from sequence mining. This approach is useful when actions order is important to identify relevant behaviours, and has been successfully applied when there are few high-level types of actions (e.g. a successful attempt on the first step of a problem, asking for hints, etc.) e.g. in [12,17]. These conditions do not apply to the test-bed educational environments we have used so far (described later), i.e. interactive simulations with many fine-grained interface actions that can be done in any order, which makes looking for recurring sequences in user actions computationally expensive without much added value.

3.2 USER CLUSTERING

In the initial version of our user-modeling approach, for clustering we used a standard implementation of the k-means algorithm [5] available in the Weka data mining package [9]. To refine the clustering step, we first experimented with other clustering algorithms available in Weka, including Hierarchical Clustering and Expectation Maximization [5]. None of these alternatives, however, substantially outperformed k-means. We thus decided to retain k-means as the clustering algorithm for our approach, but devised a method to ensure faster convergence to a good set of clusters.

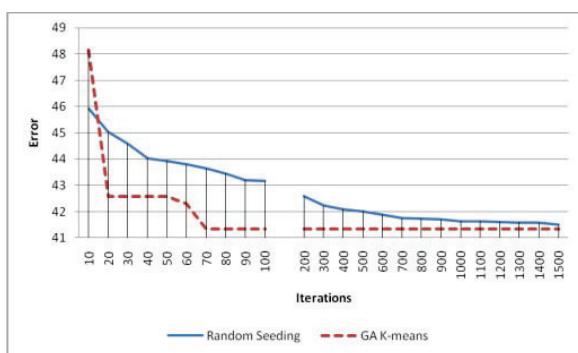


Figure 2 - Convergence of GA K-means compared to Random Seeding (K=2)

One of the issues when using the k-means is setting good initial centroids, so that the algorithm can quickly converge to a stable set of clusters with small inter-cluster error. The implementation available in Weka tended to converge slowly on the dataset we used as a test-bed for this research (described in a later section). We thus experimented with Genetic Algorithms (GA) to initialize the centroids in k-means, based on an approach suggested in [11]. This approach relies on using "chromosomes" to mold initial cluster centroids as needed. These chromosomes represent different initial values for each feature and of the initial centroids. Through mutation and crossover, in each iteration, new initial centroids are generated and the ones with lower corresponding inter-cluster error for the resultant clusters are retained for next iteration.

In our user modeling tasks, we have 21 continuous features, so the method proposed in [11] is inefficient because it requires chromosomes with too many extra bits to discretize the features without major loss of information. We thus changed the approach in [11] as follows. We generate a random population of 100 initial chromosomes, each used to generate a set of centroids that initialize a different run of k-means. We then select the half of the chromosomes that led to clusters with the lowest inter-cluster error and use

these to generate the next generation using crossover (i.e. selecting two chromosomes and choosing the upper half bits of one chromosome and the lower half of the other chromosome to form a new one) and mutation (i.e. selecting a chromosome and randomly changing one of its bits). We repeat the process until there is no improvement for a certain number of generations or we reach the maximum number of iteration limit. Our experimental results show that, although this approach does not guarantee finding the global minimum for the inter-cluster error, it converges faster than the standard random seeding method. Figure 2 for instance, compares the performances of GA k-means and the k-means from Weka on the dataset that is the test-bed for this research (averaged over 30 different runs). GA k-means converges after 100 iterations, while the standard seeding method does not reach that same error level even after 1500 iterations (here, iterations are the number of times that basic k-means is used for both cases).

3.3 ASSOCIATION RULE MINING TO DESCRIBE USER BEHAVIOURS

In our user modeling framework, association rule mining is used to identify the interaction behaviours that characterize each of the clusters found in the clustering phase. We use the Hotspot algorithm [9] to perform association rule mining on our clusters. Hotspot inspects the training data and generates the association rules corresponding to a class label (a specific cluster, in our case) in the form of a tree. For instance, two sample generic rules derived from the same tree branching could be as follows:

If <i>Action A frequency = High</i> → Cluster X	If <i>Action A frequency = High and Action B frequency = Low</i> → Cluster X
---	--

The algorithm has three parameters that influence the type and number of rules generated: the minimum level of support requested for a rule to be considered relevant (where support for rule $X \rightarrow Y$ is defined as the percentage of data points satisfying both X and Y in the dataset); the tree branching factor, influencing how many new rules can be generated from an existing one by adding a new condition; the minimum improvement in confidence needed for creating a new tree branch (where confidence for rule $X \rightarrow Y$ is the probability that Y occurs when X does). Essentially, the goal is to find a few rules that characterize as many elements in the cluster as possible and provide an easily understandable explanation of users' behaviours for each cluster.

Improvements on Rule Mining: rule generation. In the original version of the approach [1], we kept the Hotspot's default values for minimum improvement (0.01) and branching factor (2), and experimented with level of support within each cluster as a criterion to filter out rules [13]. In the new framework, we added a functionality to experiment with a variety of parameter settings. We also modified the criterion for filtering out rules so that, when there is a set of rules derived from the same tree branching, rules closer to the root and with low confidence are discarded. The rationale behind this choice is that rules with low confidence include interaction behaviours that are not representative of a specific cluster (i.e., these behaviours are observed in more than one cluster), and thus they tend to weaken the classification ability of the rule set as a whole (more detail on this point is provided in the section on user classification).

Improvements on Rule Mining: features discretization: Class association rules mining algorithms generally work with both discrete and continuous values. The attributes that describe the user interaction behaviours in our user modeling tasks are continuous, but they need to be discretized, otherwise they would produce a large number of very fine-grained rules, unsuitable for classification. Choosing the appropriate number of bins for feature discretization involves a trade-off between information loss (having too few bins)

and generating overly specific rules too detailed to capture meaningful patterns (having too many bins). While in [3] we chose a simple binary discretization, here we experimented with higher number of bins and empirically set the maximum number of bins to 7. In online user classification, as explained in the next section, the number of user actions observed is limited and it is possible that the feature values calculated for a user fall in different adjacent bins overtime, higher number of bins makes the classifier more tolerant to these fluctuations (i.e. a minor change in a feature value does not trigger a changing the label assigned to the user).

3.4 USER CLASSIFICATION

In the *user classification* phase, as new users interact with the system they are classified in real-time into one of the clusters generated by the behaviour discovery phase, based on which association rules match their behaviours. The use of association rules to construct a classifier is called Associative Classification Mining or Associative Classification [18]. Algorithms for Associative Classification usually, generate a complete set of class association rules (CARs) from training data, and then prune this initial set to obtain a subset of rules that constitute the classifier. When a new unknown object (a user in our case) is presented to the classifier, it is compared to a number of CARs and its class is predicted based on a measure that summarizes how well the user matches the CARs for each class. In the first version of our approach, the classification measure was simply the number of CARs satisfied for each cluster. This means that all rules were considered equally important for classification, failing to account for the fact that some rules with limited class support (i.e., applicable to fewer members of the class compared to others) should be considered with caution when deciding the class label of a user. In the current version, we improved the classification measure based on an approach that assigns a value to each rule, and calculates class membership scores based on the values of the satisfied rules that apply to a class (e.g. [19]). We used a variant of this approach where, instead of calculating membership scores based only on the satisfied rules, all of the CARs that represent a cluster are used. The rationale behind this choice is that, in our user modeling task the rules that do not apply to the new instance are also important for determining the final label. For instance, it is important to penalize the score of a class c when a major rule (which applies to most of the c 's members) is not satisfied by a new instance, even if a less distinctive rule for c applies to it. Accordingly, the membership function we adopted returns a score S_A for a given class A as follows:

$$(1) \quad S_A = \frac{\sum_{i=1}^m Test(r_i) \times W_{r_i}}{\sum_{i=1}^m W_{r_i}}, Test(r_i) = \begin{cases} 1 & \text{if } r_i \text{ is satisfied} \\ 0 & \text{otherwise} \end{cases}$$

Where r_i 's are the m rules selected as representative for class A , W_{r_i} is the corresponding rule weight (based on a measure explained below), and $Test(r_i)$ is the function that tests the applicability of a rule to a given instance. We tried different measures from the literature to define W_{r_i} [8] (including confidence, support, conviction and leverage) and found confidence to be the measure that generates the best classification accuracy.

4. FRAMEWORK IMPLEMENTATION

We implemented the user modeling framework as a toolset of modules that automate most of the process of going from interaction logs to generating the rule-based classifier and the adaptation rules. Most modules are implemented in Python, with some external calls to Weka through a command line interface (please note that the used functionalities from Weka are standard algorithms such as association rule mining, and can be replaced by any other standard tool or implemented internally and are transparent to the final user of the framework).

First, the **preprocessing module** reads the time stamped action logs and calculates the feature vectors. Next, the GA k-means **clustering module** generates the clusters and assigns labels to each user. The **discretization module** finds the optimal number of bins and discretizes the feature vectors (this module uses Weka to run the rule-based classifiers for finding the best number of bins). The discretized dataset is passed, along with the generated clusters, to the **rule generation module** for association rule mining and rule pruning. This module uses the Hotspot algorithm from Weka and, for each cluster, looks for the optimal settings from a set of predefined values for each of the three Hotspot parameters (i.e. minimum support, confidence improvement threshold and branching factor). The last module (**classifier**) parses the generated rules and builds a classifier that gets a new feature vector and returns the computed label. We implemented a **classifier evaluation module** that uses LOOCV and all the aforementioned modules to evaluate the classifier on available datasets, as follows.

For each fold of the LOOCV, a sub-module of **classifier evaluation** feeds the test user's data into the classifier trained on the reduced dataset, by incrementally updating the feature vector representing the interaction behaviours of this user. Predictions are then made for the incoming vector as described earlier. A second sub-module computes the accuracy of the classifier by checking (after each action in the user's log) whether the test user is correctly classified into its original cluster.

5. EVALUATION

We validated the current user-modeling framework on the Alspace CSP applet, the same interactive system we used to test previous versions. However, a larger dataset was generated for testing, which is described after illustrating the CSP applet.

5.1 THE AISPACE CSP APPLET

The Constraint Satisfaction Problem (CSP) Applet is part of a collection of interactive visualizations for learning common Artificial Intelligence algorithms, called Alspace [2]. Algorithm dynamics are demonstrated on graphs by using color and highlighting, and state changes are reinforced through textual messages (see Figure 3 for an example).

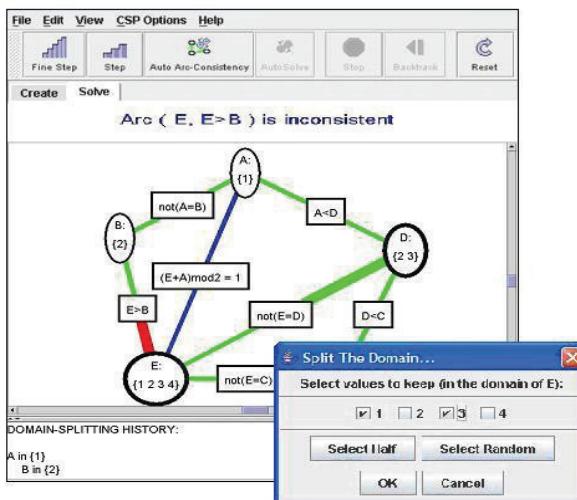


Figure 3 - CSP applet with example CSP problem

A CSP consists of a set of variables, their domains and a set of constraints on legal variable-value assignments. The goal is to find an assignment that satisfies all constraints. The CSP applet illustrates the Arc Consistency 3 (AC-3) algorithm for solving CSPs represented as networks of variable nodes and constraint arcs. AC-3 iteratively makes

individual arcs consistent by removing domain values inconsistent with a given constraint until all arcs have been considered and the network is consistent. Then, if there is still a variable with more than one value, a procedure called domain splitting is applied to that variable to split the CSP into disjoint cases so that AC-3 can recursively solve each case. The CSP applet provides mechanisms for interactive execution of the AC-3 algorithm, accessible through the toolbar shown at the top of Figure 3 or through direct manipulation of graph elements. Here we provide a brief description of these mechanisms necessary to understand the results of applying our student modeling approach to this environment:

- *Fine Stepping*. Cycles through three detailed algorithm steps: selecting an arc, testing it for consistency, and removing variable domain values when necessary.
- *Direct Arc Clicking*. Allows the user to decide which arc to test, and then performs three *Fine Steps* on that arc to make it consistent.
- Auto Arc Consistency (Auto AC). Automatically Fine Steps through the network.
- Stop. Stops Auto AC.
- *Domain Splitting (DS)*. Allows the user to select a variable domain to split, and specify a sub-network for further application of AC-3.
- *Backtracking*. Recovers the alternative sub-network set aside by *DS*.
- *Resetting*. Resets the CSP network to its initial state.

In the following sections, we describe the performance of our user-modeling framework to create a classifier user model for the CSP applet. This model will eventually be used to provide adaptive interventions for students who do not learn well while using this environment. We evaluated our framework along several dimensions.

5.2 DATA COLLECTION

Data for our evaluation was collected via user studies based on the experimental protocol for the CSP applet described in [1]. University students, who were familiar with basic graph theory but had not taken an AI course, were introduced to the AC3 algorithm and then took a pretest on the topic. Next, each participant used the applet on two CSP problems and wrote a posttest. The resulting dataset includes action logs for 65 users (compared to 24 users in [3]), totalling 13,078 actions over 62,752 seconds of interaction. From these logs, we calculated: (*i*) usage frequency of each interface action (*ii*) mean and standard deviation of latency between actions. Average latency is an indicator of the time spent reflecting after an action and planning for the next one, while standard deviation of latency tells if the user was consistent or selective in amount of pausing after each action. Since we have 7 interface actions the calculated feature vectors are 21-dimentional.

5.3 RESULTS: RELATION OF CLUSTERS TO LEARNING

First, we want to see if/how the discovered clusters relate to user learning. From the study test scores we computed the proportional learning gains for each student (in percentage), and then analyzed the clusters detected for K=2 (we used C-index as described in [15] to determine the optimal number of clusters for the data), to see if there is any significant difference with regard to learning gains. An independent samples t-test revealed a significant difference in the learning gain between the two clusters ($p = .03 < .05$), with a medium effect size (Cohen $d = .47$). We refer to these clusters as High ($n = 18$, $M = 61.32$, $SD = 27.38$) and Low ($n = 47$, $M = 39.28$, $SD = 62.06$) Learners (HL and LL). There is no significant difference between the average pretest scores of LL and HL ($p = .19$), indicating that behaviour patterns of the HL group have an impact on their learning.

5.4 RESULTS: USEFULNESS OF ASSOCIATION RULES FOR ADAPTATION

We also want to verify whether the rules generated by the framework can be used to define adaptive interventions. Table I shows a subset of the representative rules for the

HL and LL clusters in our experiment, where we report the preconditions for each rule but leave out the consequence. The table also shows, for each rule, its level of confidence (*conf*), and support within its cluster (*class cov*). These rules were generated by discretizing the feature vectors in our dataset into seven mutually exclusive ranges (bins), as explained earlier.

Direct Arc Click frequency appears in Rule1 for the HL cluster, with value in the highest bin, while it appears in Rule 3 for LL with the lowest value, indicating that LL members use *Direct Arc Click* much less than HL members. The high class coverage of Rule1 for HL (100%) indicates that high frequency of *Arc Click* pertains to all high learners, and thus it would be beneficial to trigger this behaviour for students who otherwise would not engage in it. Low values of *Direct Arc Click Pause* average and standard deviation in Rule1 and Rule2 for LL suggest that, even when they do select arcs proactively, LL students consistently spend little time thinking about this action's outcome. Finally, the high level of confidence of Rule1 for HL (100 %) indicates that, this rule will have high impact in classifying new users as per equation (1).

The above observations suggest, for instance, the following adaptation rules for the CSP applet. “**IF** user is classified as a LL and is using *Direct Arc Click* very infrequently **Then** give a hint to prompt this action”; “**IF** user is classified as a LL and pauses very briefly after a *Direct Arc Click* **Then** intervene to slow down the student”

Table I. The representative rules for HL and LL clusters

Rules for HL cluster[*]: Rule1: Direct Arc Click frequency = Highest (Conf=100%, Class Cov = 100%) Rule5: Domain Split frequency = Highest and Auto AC frequency = Lowest and Fine Step Pause Avg = Highest (Conf = 50%, Class Cov = 50%) └ Rule8: Domain Split frequency = Highest and Auto AC frequency = Lowest and Fine Step Pause Avg = Highest and Reset frequency = Lowest (Conf = 65%, Class Cov = 76.47%)
Rules for LL cluster[*]: Rule1: Direct Arc Click Pause Avg = Lowest (Conf=100%, Class Cov = 100%) Rule2: Direct Arc Click Pause STD =Lowest (Conf = 95.83%, Class Cov = 95.8%) Rule3: Direct Arc Click frequency = Lowest (Conf = 93.48%, Class Cov=93.5%) └ Rule4: Direct Arc Click frequency = Lowest and Direct Arc Click Pause Avg = Lowest (Conf=100%, Class Cov=100%)

* Conf= Confidence; Avg= Average; Class Cov= Class Coverage, STD = Standard Deviation

As another example, consider Rule8 for the HL cluster. This rule indicates that HL members (*i*) use *Auto AC* action sparsely. Recall that this action quickly runs AC3 to completion, and thus is not very useful for learning how the algorithm works; (*ii*) perform *Domain Split* (the most advanced step in the algorithm) frequently (*iii*) spend the highest average time thinking about each *Fine Step* they take. This is an ensemble of effective behaviours that should be encouraged in an adaptive version of the CSP applet. Taken individually, these behaviours don't show a statistically significant difference between LL and HL and thus would not be identified as relevant by a pair wise analysis of features (as performed, for instance, in [1]).

In summary, this section illustrates that the rules generated by our framework are informative and can be used for generating real-time adaptive interventions. These interventions, however, are appropriate only if the classifier user model can recognize which users need them. Thus, the next section discusses classifier's performance.

5.5 RESULTS: PERFORMANCE ON USER CLASSIFICATION

We used LOOCV, as explained earlier, to evaluate the accuracy of our rule-based

classifier and compare it against: (i) a baseline that always predicts the most likely label (LL in our dataset); (ii) the best achieving classifier among various complex classifiers available in Weka, i.e., the Random Subspace meta-classifier using C4.5 as the base classifier; (iii) the classifier obtained with the earlier version of the framework [1] (old rule-based classifier in Figure 4). Note that all these four classifiers use the categories learned via the unsupervised process described in sections 3.1 though 3.4. We also want to compare our approach against a fully supervised approach that starts from categories defined based on the available learning gains. For this, we calculated the median of the learning gains and labelled the students above the median as high learners and others as low learners. We then trained and tested a C4.5 classifier with these new labels.

Figure 4 shows the overtime average accuracy of these five classifiers, both in terms of percentage of correct classifications for the individual clusters (LL and HL), and overall. The new rule-based classifier has the highest overall accuracy, and the differences with the other classifiers are statistically significant ($p < .001$), with a large effect size ($d > 3$). For each cluster, the accuracy of new classifier is comparable with the best competitor, but no other classifier achieves the same accuracy in both clusters.

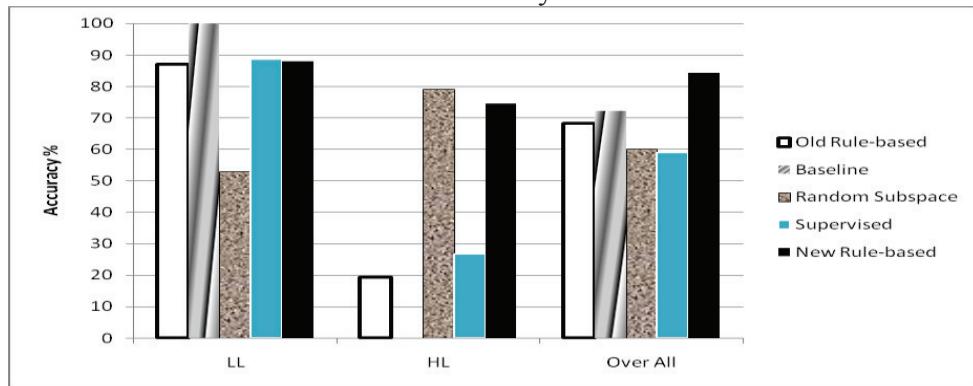


Figure 4 - The overtime average accuracy of different classifiers compared to the new rule-based classifier

Figure 5, shows accuracy of the new classifier as a function of the percentage of observed actions, both overall and for the individual clusters.

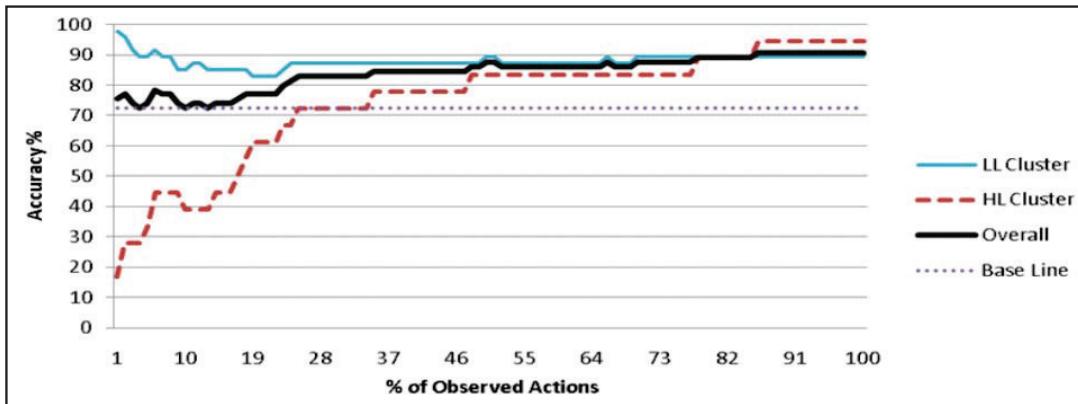


Figure 5 – Accuracy of the new rule-based classifier as a function of the percentage of observed actions

For comparison, we include the overall accuracy of the baseline, which is the best performing classifier after ours. The new rule based classifier reaches a relatively high accuracy in early stages of the interaction which is very important when the goal is to provide adaptive interventions to improve the user experience with the educational software. The overall accuracy of the new classifier becomes consistently higher than all the other classifiers before observing 20% of user actions, and accuracy on each cluster goes above 80% after seeing about 50% of the actions, while the baseline consistently misclassifies high learners throughout.

6. CONCLUSION AND FUTURE WORK

In this paper, we describe a user modeling framework that uses unsupervised clustering and Class Associating Mining to discover and recognizes relevant interaction patterns during student interaction with educational software. The framework improves a previous proof-of-concept approach by adding functionalities for more efficient clustering and more principled selection of some of the required parameters. An empirical evaluation of the framework provides evidence that it can both cluster users into meaningful groups, as well as classifying new users accurately. More importantly, the framework generates rules that provide a fine grained description of common behaviours for users in different clusters. These rules appear to be suitable to guide adaptive interventions targeted at improving interaction effectiveness. The next step of this work will be to add these adaptive interventions to the educational software we have been using as a test-bed for this research, an interactive simulation to help students understand an algorithm for constraint satisfaction. We also plan to use the framework for generating classifier user models for other educational software developed in our lab, including interactive simulations for other AI algorithms and an educational game for mathematical skills.

REFERENCES

1. AMERSHI, S. AND CONATI, C. 2009. Combining Unsupervised and Supervised Machine Learning to Build User Models for Exploratory Learning Environments. *J. of Educational Data Mining* 1, 1, 2.
2. AMERSHI, S., CARENINI, G., CONATI, C., MACKWORTH, A. AND POOLE, D. 2008. Pedagogy and Usability in Interactive Algorithm Visualizations - Designing and Evaluating ClSpace. *Interact Comput.* 20, 1, 64-96.
3. BERNARDINI, A., AND CONATI, C. 2010. Discovering and Recognizing Student Interaction Patterns in Exploratory Learning Environments. *Proc. ITS2010*, Springer, 125-134.
4. CHANGCHIEN, S.W., AND LU, T. 2001. Mining association rules procedure to support on-line recommendation by customers and products fragmentation. *Expert Systems with Applications* 20, 325-335
5. DUDA, R. O., HART, P. E., AND STORK, D., G. 2001. *Pattern Classification*. New York: Wiley- Interscience.
6. FREYBERGER, J., HEFFERNAN, N., AND RUIZ, C. 2004. Using association rules to guide a search for best fitting transfer models of student learning. Workshop on analyzing student-tutor interactions logs to improve educational outcomes at ITS conference, 1-4.
7. GARCIA, E., ROMERO, C., VENTURA, S., AND CASTRO, C. D. 2009. An architecture for making recommendations to courseware authors using association rule mining and collaborative filtering. *UMUAI* 19, 99-132.
8. GENG, L., AND HAMILTON, H. J. 2006. Interestingness measures for data mining: A survey. *ACM Comput. Surv.* 38, 3, 9.
9. HALL, M., EIBE, F., HOLMES, G., PFAHRINGER, B., REUTEMANN, P., WITTEN, I.H. 2009. The WEKA data mining software: an update. *SIGKDD Explor.* 11(1), 10–18.
10. HUNT, E., AND MADHYASTHA, T. 2005. Data Mining Patterns of Thought. In *Proc. the AAAI Workshop on Educational Data Mining*.
11. KIM, K. AND AHN, H. 2008. A recommender system using GA k-means clustering in an online shopping market. *Expert Syst. Appl.* 34, 2, 1200-1209.
12. KOCK, M., PARAMYTHIS, A. 2011. Activity sequence modelling and dynamic clustering for personalized e-learning. *User Model User-Adap Inter* 21:51-97.
13. LIU, B., HSU, W., AND MA, Y. 1999. Mining association rules with multiple minimum supports. In *Proc. KDD '99*. ACM, 337-341
14. MERCERON A., AND YACEF K. 2003. A web-based tutoring tool with mining facilities to Improve Teaching and Learning, *AIED 2003*, 201-208.
15. MILLIGAN G. W., AND COOPER, M. C. 1985. An examination of procedures for determining the number of clusters in a data set, *Psychometrika* , 50, 2, 159-179.
16. ROMERO, C., VENTURA, S., ZAFRA, A., AND DE BRA, P. 2009. Applying Web usage mining for personalizing hyperlinks in Web-based adaptive educational systems. *Comput. Educ.* 53, 3, 828-840.
17. SHANABROOK, D. H., COOPER, D. G., WOOLF, B. P., ARROYO, I. 2010. Identifying High-Level Student Behaviour Using Sequence-based Motif Discovery. In *Proceedings of EDM'2010*.191-200.
18. THABTAH, F. 2007. A review of associative classification mining. *Knowl. Eng. Rev.* 22, 1, 37-65.
19. YIN, X., AND HAN, J. 2003. CPAR: Classification based on predictive association rules. In *Proc. of the Third SIAM International Conference on Data Mining*, SIAM, 208-217.
20. ZHANGA, Y., AND JIAO, J. 2007. An associative classification-based recommendation system for personalization in B2C e-commerce applications, *Expert Syst. Appl.* 33, 2, 357-367.

How to Classify Tutorial Dialogue? Comparing Feature Vectors vs. Sequences

JOSÉ P. GONZÁLEZ-BRENES, WEISI DUAN, AND JACK MOSTOW
Language Technologies Institute, Carnegie Mellon University, USA

A key issue in using machine learning to classify tutorial dialogues is how to represent time-varying data. Standard classifiers take as input a feature vector and output its predicted label. It is possible to formulate tutorial dialogue classification problems in this way. However, a feature vector representation requires mapping a dialogue onto a fixed number of features, and does not innately exploit its sequential nature. In contrast, this paper explores a recent method that classifies sequences, using a technique new to the Educational Data Mining community – Hidden Conditional Random Fields [Quattoni et al., 2007]. We illustrate its application to a data set from Project LISTEN’s Reading Tutor, and compare it to three baselines using the same data, cross-validation splits, and feature set. Our technique produces state-of-the-art classification accuracy in predicting reading task completion. We consider the contributions of this paper to be (i) introducing HCRFs to the EDM community, (ii) formulating tutorial dialogue classification as a sequence classification problem, and (iii) evaluating and comparing dialogue classification.

Key Words and Phrases: Project LISTEN, Feature Vectors, Sequence Classification, Reading Task Completion

1. INTRODUCTION

Researchers in education have long distinguished a *student trait*, a characteristic that is relatively constant, from a *student state*, a characteristic that changes thorough time [Reigeluth, 1983]. In this paper, we discuss how to train a classifier to represent time-varying characteristics of student states.

We illustrate our discussion with an example. Suppose we are classifying computer-student dialogues using the single feature “turn duration”. Figure 1 shows the duration of each of the turns in a dialogue (9s, 8s, 5s, 7s, and 6s respectively). Conventional classifiers, like logistic regression or decision trees, rely on a fixed-size feature vector as an input; hence, we have to decide *a priori* how many features we are going to include. But, how to map into a fixed-size feature vector a dialogue that may vary in number of turns? One approach is to extract features from a window, either from the beginning or the end of the dialogue [González-Brenes and Mostow, 2011]. There are (at least) two alternative approaches: (i) averaging the value of the features in the window – in our example, it would be a single feature with value 6.0; or (ii) having a feature for every turn – in our example, three features with values 5, 7 and 6. Once we transform dialogues into feature vectors, we can train conventional classifiers on them.

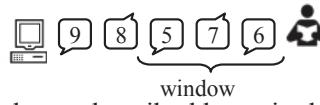


Figure 1: Dialogue described by a single feature

Mapping dialogues into feature vectors does not innately capture or exploit the sequential nature of dialogue. Furthermore, it is not clear how appropriate the window strategy is, since short windows may exclude important information, whereas long windows may have too many missing values. In this paper, we consider the alternative approach of classifying over the entire dialogue using sequences, by applying Hidden Markov Models, and we introduce a recent technique, Hidden Conditional Random Field (HCRF) [Quattoni et al., 2007].

Authors’ addresses: J.P. González-Brenes, e-mail: joseg@cs.cmu.edu; W. Duan, e-mail: wduan@cs.cmu.edu; J. Mostow, e-mail mostow@cs.cmu.edu. , Project LISTEN, RI-NSH 4103, 5000 Forbes Avenue, Carnegie Mellon University, Pittsburgh, PA 15213-3890, USA.

The rest of this paper is organized as follows. Section 2 discusses relation to prior work. Section 3 describes the different feature vector and sequence classifiers we consider to classify dialogues. Section 4 presents empirical results on a classification task to predict whether a student will complete a reading task. Section 5 concludes.

2. RELATION TO PRIOR WORK

Previous work on representations of data in language technologies has relied on feature vectors using bag of word representations, n -grams, or their projections into latent space [Wallach, 2006]. Alternatively, kernels have allowed richer representations. For example, for text classification, the String Kernel [Lodhi et al., 2002], represents documents in a feature space of all of the substrings of length k . A similar feature vector representation would involve a prohibitive amount of computation, since the size of the feature vector space grows exponentially with k . Sequence Kernels have been used for speaker verification to map the audio signal sequence into a single feature vector using polynomial expansions [Louradour et al., 2006]. We are unaware of alternative classification approaches for dialogue other than using feature vectors.

Classification of sequences can be categorized in three different ways [Xing et al., 2010]: feature vector based classification, model based classification, and distance based classification. In the rest of this section, we discuss previous approaches to dialogue classification in these categories.

2.1 Feature Vector Based Dialogue Classification

As discussed earlier, sequences can be mapped into fixed-size feature vectors. As far as we know, all of the previous approaches in classification of tutorial dialogue have ignored the sequential nature of dialogue, constraining dialogue into a fixed-size representation. For example, predicting dialogue completion has been studied extensively in the literature, relying on a feature vector representation [González-Brenes et al., 2009; González-Brenes and Mostow, 2010; González-Brenes and Mostow, 2011; Hajdinjak and Mihelic, 2006; Möller et al., 2008; Möller et al., 2007; Walker et al., 2001].

2.2 Model Based Dialogue Classification

Model based classification models sequences directly, for example using Hidden Markov Models (HMMs). In this paper, we advocate for model based approaches over using feature vectors.

HMMs have been used extensively in language technologies, for example in topic segmentation [Eisenstein et al., 2008]. In the dialogue community, to our knowledge, HMMs have been used only to segment dialogue [Stolcke et al., 2000], but not to classify it as we do here. A growing body of work has investigated how to use policy learning to improve tutorial effectiveness [Ai et al., 2007; Beck, 2004; Beck and Woolf, 2000; Boyer et al., 2010; Chi et al., 2008; Chi et al., 2010]. Policy learning often relies on Markov Decision Processes (MDPs) [Singh et al., 1999] to learn a strategy that maximizes the expected value of a specified reward function. MDPs are very similar to HMMs in that the input is a sequence. However, learning a strategy for what to do at each point in a dialogue is a different problem than learning a classifier. Although speech is traditionally modeled as a sequence of phonemes [Gunawardana et al., 2005], we believe we are the first to model dialogues without using feature vectors. We do not know of any previous use of HCRFs in the Educational Data Mining community.

2.3 Distance Based Classification

Distance-based methods for sequence analysis rely on a distance function to measure the similarity between two sequences. Dialogue System Difference Finder [González-Brenes

et al., 2009] defines a distance function between dialogues described by feature vectors. We are unaware of distance functions between dialogues that model dialogues as sequences.

3. DIALOGUE CLASSIFICATION

In this section, we discuss the classification algorithms we considered to model tutorial dialogue behavior using either feature vectors or sequences. For feature vector classification we considered Maximum Entropy Classification [Berger et al., 1996] and Random Forest [Breiman, 2004]. We used Maximum Entropy Classification, often called Logistic Regression, as a baseline because of its recent success in classifying tutorial dialogue [González-Brenes and Mostow, 2011]. Random Forest, often called Ensemble of Decision Trees, has provided good empirical results in the EDM community, having been used in the winning submission of the Educational Data Mining Challenge at SIGKDD 2010.

Alternatively, for classifying sequences, we use the popular Hidden Markov Model (HMM) approach [Rabiner, 1989]. We also introduce to the EDM community a recent technique called Hidden Conditional Random Fields (HCRFs), which have been applied to other domains [Gunawardana et al., 2005; Sy Bor, 2006]; for details of their implementation, see [Quattoni et al., 2007].

Maximum Entropy, and HCRF can be formulated under an approach called risk minimization [Obozinski et al., 2007], where the parameters are estimated by maximizing the fit to the training data while penalizing model complexity (number of features). Better fit to the training data favors classification accuracy in the training set, but risks over-fitting the model to the data. Conversely, low model complexity sacrifices classification accuracy on the training set in hopes of generalizing better to unseen data. Both Maximum Entropy and HCRF are log-linear and discriminative – they model the differences between class labels without inferring generative models of the training data. However, they differ in the way they calculate the fit to the training data: HCRFs use a latent variable (a hidden state) to model input sequences, while logistic regression uses feature vectors. To penalize complexity, they both rely on regularization penalties. The two most popular regularization penalties are the L_1 norm and the L_2 norm of the feature vector [Ng, 2004]. The L_1 norm selects fewer features than the L_2 norm, and hence it is used when interpretability of the model is desired, or when the number of features exceeds the number of data points. Conversely, when the number of features is small compared to the training data, the L_2 norm offers better predictive power [Zou and Hastie, 2005]. The trade-off between fit to the training data and model complexity is controlled by a so-called regularization hyper-parameter, often optimized during cross-validation using a held-out set of development data.

Random Forest is an ensemble of decision trees. To avoid over-fitting, each tree is grown using only a random subset of the features and a random subset of the training data. The training procedure grows each tree greedily, selecting the best decision split at each node, and stopping when each leaf has five data points, with no pruning. During testing, Random Forest returns the class predicted by the largest number of decision trees. Random Forest does not assume that the data belongs to any particular distribution, and hence it is considered a non-parametric approach.

An HMM is a generative classifier. Thus to distinguish between two classes, it requires two models: one for the positive class, and one for the negative class. Like an HCRF, an HMM models its input as a sequence, and uses a latent variable to model hidden state. Using hidden variables in HCRFs and HMMs converts the learning problem into non-convex optimization. Consequently, the parameters used to initialize the model in the training procedure affect its final performance.

In Table II, we show a summary of the differences between the classifiers we described. The plate diagrams of Maximum Entropy, HCRFs, and HMMs follow the convention of coloring the circles of the variables that are observable during training. The outcome variable y is the class label we want to learn. For example, we may label dialogues with $y=+1$ if they were completed successfully, and $y=-1$ otherwise. For feature vector classification, x is a feature vector describing an entire dialogue. In contrast, for sequence classification, x_t is the value of the features at time t . For example, x can represent the duration of each turn in the dialogue. The hidden discrete variable s_t is not directly observed. Figure 2 expands the plate notation of Table II for HCRFs. The undirected graphical model notation indicates that a variable is independent of all the other variables given its neighbors. For example, the hidden state s_T is independent of all other variables given y , x_T , and s_{T-1} . Instead of drawing each repeated variable, a plate is used to group repeated variables. The class label y depends only on the hidden states. The directed graph notation to represent HMMs uses a conventional Bayesian Network representation.

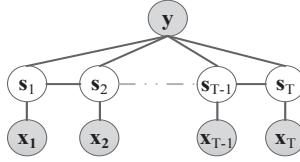


Figure 2: Graphical representation of a Hidden Conditional Random Field

Table II: Bird's-eye View of the Classifiers Considered

	Random Forest	Max. Ent.	HCRF	HMM
Input	Feature vector	Feature vector	Sequences	Sequences
Classifier type	Non-parametric	Parametric / discriminative	Parametric / discriminative	Parametric / generative
Convex?	No explicit objective function	Yes	No	No
Random?	Subset of training and features	No	Initialization	Initialization
Distribution	Non-linear	Log-linear	Log-Linear (in latent space)	Gaussian
Overfitting protection	Randomization	Regularization penalty	Regularization penalty	Prior on emission distribution

4. EXPERIMENTAL EVALUATION

We compared the methods on data logged by Project LISTEN’s Reading Tutor, which listens to a child read aloud, and takes turns picking stories to read [Mostow and Aist, 2001]. The Reading Tutor adapts the Sphinx-II speech recognizer [Huang et al., 1993] to analyze the students’ oral reading, and intervenes when it notices the reader makes a mistake, get stuck, click for help, or encounter difficulty. Figure 3 shows a screenshot of the 2005 Reading Tutor. The current sentence is in boldface, and the tutor is giving help on the highlighted word *teach*. The Reading Tutor is an atypical dialogue system, in the sense that it is not designed to answer questions by a user. However, it addresses such

dialogue phenomena as turn-taking, backchanneling, mixed initiative, and multimodal interactions (speech and mouse).



Figure 3: Project LISTEN’s Reading Tutor screenshot

We demonstrate our approach using a previously studied prediction problem [González-Brenes and Mostow, 2010; González-Brenes and Mostow, 2011]. We count the interaction of a student reading a story with the tutor as one dialogue. We consider dialogues to be composed of one or more *sentence encounters* in which the Reading Tutor displays a sentence for the student to read. We want to classify each dialogue at runtime, based on the sentence encounters so far, according to whether the student will finish reading the story or is about to stop reading.

We calculate all features using only information available at prediction time. Thus for positive training examples, we truncate each finished story to a random number of sentence encounters before calculating features. For negative training examples, we use unfinished stories, but we do not truncate them. Table II summarizes the sorts of dialogue features used. We extract features only from the student’s sentence encounters, not from the tutor’s utterances, because we want to base our predictions on the student’s behavior.

Table II: Features considered

Prosodic Features: Various duration, pitch and intensity features, as described in [Duong and Mostow, 2010].
Sentence Features: Properties of a sentence to be read, such as percentage of story read so far, number of word types and tokens, number of clicks for help, and statistics on word length and frequency

4.1 Dataset

The data set we used was logged by Project LISTEN’s Reading Tutor while used regularly at elementary schools during the 2005-2006 school year. To obtain a balanced data set of 2,112 dialogues with 162 children, we randomly selected half of them from dialogues where the students completed the reading, and the other half where they did not. We only included dialogues with at least four sentence encounters, to provide some basis for prediction. The selected dialogues average 18 sentences encounters.

Training and testing on the same students can risk relying on peculiarities of individual students. Hence, we separated the data such that the development and testing sets had no students from the training set. We report all results using 10-fold cross-validation across students. Because the folds can vary in size, we report an average weighted by the number of data points in each individual fold. We take this variation into consideration when we report significance in our statistical tests, weighting each fold accordingly, as described previously [Bland and Kerry, 1998]. We split each fold into four non-overlapping sets: training set (70% of the students), two development sets (each with 10% of the students), and test set (with 10% of the students).

4.2 Data preparation

When using HMMs with continuous feature values, the initialization of the parameters is very important. One of the parameters of a continuous HMM, the covariance matrix of the emission probability, cannot be initialized with just any random numbers. In fact, at every point of training, it should adhere to some rules: it must be non-singular (invertible) and positive semi-definite. On preliminary experiments with HMMs, the large size of the feature set relative to the amount of training data resulted in non-singular covariance matrices that assign infinite likelihood to sequences.

To avoid such problems in training HMMs, we reduced the feature space by eliminating features that are “usually the same value.” For this purpose we used a statistical property of distributions called kurtosis, also referred as the fourth standardized moment. Gaussian distributions have zero kurtosis, “peakier” distributions than the Gaussian have positive kurtosis, and conversely, flatter distributions have negative kurtosis. To make our results comparable across classifiers, we want all classifiers to have access to exactly the same set of features. Hence, we use the training data in each cross-validation fold to remove features that have kurtosis greater or equal than 100, so all other classifiers are on a level playing field with HMMs. We also perform the standard transformation of centering the feature values as z-scores with mean zero and standard deviation one.

The value of our features changes thorough time. But what’s the minimum unit of time? Our methods depend on discrete time-steps, and so we considered the following alternatives: one second, a word uttered by the student, or a sentence encountered. For simplicity we decided to use a sentence encounter as the minimum time unit, since it was the easiest to map from the format of the tutor logs. Hence, to map dialogues into feature vectors, we extract features from a window of w sentences from the end of the dialogue. In the case of predicting task completion, the last dialogue turns are the most informative [González-Brenes and Mostow, 2011]. Sequences are computed by calculating the values of the features for each sentence encounter.

4.3 Describing Dialogues as Feature Vectors

In this subsection we explore the shortcomings of using feature vectors to describe tutorial dialogue. For this purpose, we compared the following feature vector classifiers:

- Random Forest, as implemented by the Statistics Toolbox of Matlab. We used 300 decision trees in each forest.
- Maximum Entropy classifier with L_1 and with L_2 regularization. We used PMTK for Matlab [Murphy, 2012, in preparation], February 28th, 2011 release¹.

We now analyze the effect of the window size on classification accuracy. Figure 4 shows the average cross-validation accuracy of the classifiers tested on the Development Set 2. The classifiers were trained using the Training Set portion of each fold. For Maximum Entropy, we tuned the regularization hyper-parameters using the Development Set 1 within each fold independently. The left panel of the figure shows the accuracy of classifiers averaging the values of the features across different window sizes, and the right panel shows the accuracy of classifiers using a different feature at each time step (for example, the “duration” would be represented with different features if it is computed over the last sentence, or the second to last sentence, and so on). All of the classifiers significantly outperform the expected value of a classifier that randomly picks class labels (the “guess” line), as determined by a one-sample t -test at the 5% significance level.

¹ <http://code.google.com/p/pmtk3/>

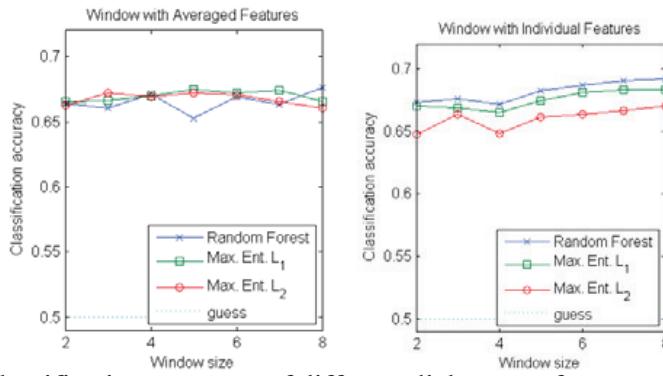


Figure 4: Classification accuracy of different dialogue to feature vector strategies

The strategy of aggregating features achieves approximately 67% classification accuracy across all different window sizes regardless of the classifier used – the differences between classifiers are not significant at $p < .05$. On the other hand, in the right panel of Figure 4, when features are not aggregated within time steps, there is a trend of increasing classification accuracy with larger windows. The best classifiers use the largest window size: Random Forest achieves 69.2% classification accuracy, followed by L_1 -regularized Maximum Entropy with 68.3% accuracy and L_2 -regularized Maximum Entropy with 67.0% accuracy.

We observe that when modeling individual features for each time step individually, longer windows have better classification accuracy. This finding supports the hypothesis that a representation that includes the whole dialogue is desirable. Because the size of a dialogue is unbounded, it is impossible to define a feature vector that could describe each of the time steps of any dialogue without aggregation. Furthermore, feature vector classifiers do not know that some features represent a value that is changing through time, and hence do not exploit any temporal relation. In the next subsection, we explore a more natural way to model tutorial dialogue as sequences.

4.4 Describing Dialogues as Sequences

We study modeling tutorial dialogue as sequences directly. For this purpose we compared the following sequence classifiers:

- Hidden Markov Models, using the PMTK toolkit mentioned earlier.
- Hidden Conditional Random Fields, using the HCRF Library,² version 2.0b. We chose the option of using L-BFGS as the optimizer.

The classifiers were trained using the training set portion of each fold. For HMMs, we used five random restarts, picking the best initialization using Development Set 1. To initialize the parameters, we chose the library’s default initialization, which uses a prior to favor a diagonal covariance matrix for the emission probability. We used the conventional Expectation Maximization (EM) algorithm to estimate the transition and emission probabilities. We did not perform random restarts for the HCRFs due to time constraints.

Since sequential models rely on hidden states, we want to understand the effect of the number of hidden states on classification accuracy. Figure 5 shows the average cross-validation accuracy of the classifiers tested on the Development Set 2. For HCRFs, the regularization hyper-parameters tuned were tuned with the Development Set 1 within each fold independently. All of the classifiers significantly outperform the expected value of a classifier that randomly picks class labels (the “guess” line), as determined by a one-sample t -test at the 5% significance level. We observe that using L_1 regularization does

² <http://sourceforge.net/projects/hcrf/>

not affect the classification accuracy across the number of hidden states, as it stays relatively constant around 69%. L_2 regularization is more prone to overfitting, and hence the addition of extra hidden states reduces classification accuracy, from 69% with two hidden states, to 65% with three. HMMs are the worst performing models, with a classification accuracy of 61% with two hidden states, which gets a small gain with three hidden states to 62%, and then decreases again to 61% with four hidden states.

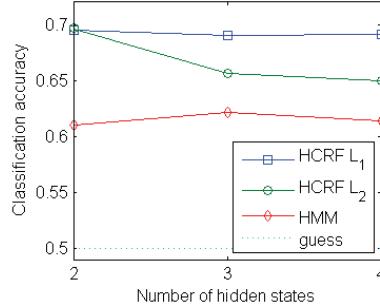


Figure 5: Classification accuracy across different number of hidden states

4.5 Feature Vectors versus Sequences

We now investigate whether classifying over entire sequences offers better classification accuracy than using feature vectors. For this comparison, we select the best classifiers from the two previous subsections, and test them on the unseen test set of each fold. That is, we compare HCRF with L_1 regularization, Random Forest, Maximum Entropy (using individual features for time steps) with L_1 regularization, and HMM.

Table 1 shows the classification accuracy of the best classifiers described earlier with their 95% confidence intervals. We observe that HCRFs using L_1 regularization outperforms all other classifiers, with a classification accuracy of 69.32%. Although the confidence intervals overlap, a t -test at the 5% level reveals that HCRFs are significantly better than Maximum Entropy (accuracy = 66.57%) and HMMs (accuracy = 62.50%).

Table 1 Classifier Comparison

	Accuracy	Precision	Recall
HCRF (L_1)	.6932 ± .03	.6909	.6890
Random Forest	.6799 ± .03	.6709	.7172
Maximum Entropy (L_1)	.6657 ± .03	.6669	.6704
HMM	.6250 ± .03	.5932	.8000
Random Baseline	.5000 ± .03	.5000	1

Random Forest is a strong contender, because unlike the other methods we compared, it does not assume any particular distribution of the data. However, a t -test reveals that its classification accuracy is not significantly different ($p > 0.05$) from the Maximum Entropy baseline used in previous work [González-Brenes and Mostow, 2011]. The t -test does not reject the null hypothesis that HCRFs and Random Forest (accuracy = 67.99%) have the same classification accuracy. This finding may suggest that the HCRF model allows more consistent results, but further experimentation is required to understand when and why each method works better than the other one. HCRF and Random Forests took a few hours to train, without making use of the parallelization options available.

5. CONCLUSION

We consider the contributions of this paper to be (i) introducing HCRFs to the EDM community, (ii) formulating tutorial dialogue classification as a sequence classification

problem, and (iii) evaluating and comparing dialogue classification algorithms to predict completion of a reading task by children using Project LISTEN's Reading Tutor. A limitation of our approach is that we did not perform explicit feature selection before learning a classifier. This omission may have had a negative impact on the classification accuracy of models more prone to over-fitting, particularly HMMs.

Although HMMs can also classify tutorial dialogue using sequences, they do not achieve good classification accuracy, presumably because they do not scale well to large feature sets. HCRF allows state-of-the-art results for predicting reading task completion. Moreover, HCRF allows modeling tutorial dialogues as sequences, which is a more natural representation than feature vectors.

Future work should study how the hidden states of HCRF segment the dialogues. We hypothesize that the hidden states of the model are related to the motivational states of the students. Additionally, we believe that further improvement in classification accuracy could be gained with a model that combines the strength of using a sequence representation with a non-parametric approach such as Random Forest.

ACKNOWLEDGEMENTS

This work was supported in part 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. We thank the educators, students, and LISTENers who helped generate, collect, and analyze our data, and the reviewers for their helpful comments. We appreciate Prof. Louis-Phillipe Morency's help with the HCRF library. José was partially supported by the Costa Rican Ministry of Science and Technology (MICIT).

REFERENCES

- AI, H., TETREAULT, J.R. and LITMAN, D.J. 2007. Comparing user simulation models for dialog strategy learning, Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers. Association for Computational Linguistics, Rochester, New York, 1-4.
- BECK, J. 2004. Using response times to model student disengagement, Proceedings of the ITS2004 Workshop on Social and Emotional Intelligence in Learning Environments, 13-20.
- BECK, J. and WOOLF, B.P. 2000. High-Level Student Modeling with Machine Learning, Proceedings of the 5th International Conference on Intelligent Tutoring Systems. Springer-Verlag, London, UK, 584-593.
- BERGER, A.L., PIETRA, V.J.D. and PIETRA, S.A.D. 1996. A maximum entropy approach to natural language processing. *Comput. Linguit.* 22, 39-71.
- BLAND, J.M. and KERRY, S.M. 1998. Weighted comparison of means. *British Medical Journal* 316(7125), 129.
- BOYER, K., PHILLIPS, R., INGRAM, A., HA, E., WALLIS, M., VOUK, M. and LESTER, J. 2010. Characterizing the Effectiveness of Tutorial Dialogue with Hidden Markov Models, Intelligent Tutoring Systems. Springer Berlin / Heidelberg, 55-64.
- BREIMAN, L. 2004. Consistency for a simple model of random forests. 670, University of California, Berkeley, Berkeley.
- CHI, M., JORDAN, P., VANLEHN, K. and HALL, M. 2008. Reinforcement Learning-based Feature Selection For Developing Pedagogically Effective Tutorial Dialogue Tactics, Proceedings of the 1st International Conference on Educational Data Mining, Montreal, 30-36.
- CHI, M., VANLEHN, K. and LITMAN, D. 2010. Do Micro-Level Tutorial Decisions Matter: Applying Reinforcement Learning to Induce Pedagogical Tutorial Tactics, Intelligent Tutoring Systems. Springer Berlin / Heidelberg, Berlin, 224-234.
- DUONG, M. and MOSTOW, J. 2010. Adapting a Prosodic Synthesis Model to Score Children's Oral Reading. In *Interspeech 2010*, Makuhari, Japan, Sept. 26-30, 2010.
- EISENSTEIN, J., BARZILAY, R. and DAVIS, R. 2008. Gestural cohesion for topic segmentation. *ACL: HLT*, 852-860.
- GONZÁLEZ-BRENES, J.P., BLACK, A.W. and ESKENAZI, M. 2009. Describing Spoken Dialogue Systems Differences. In *International Workshop on Spoken Dialogue Systems*, Irsee, Germany, 2009, Best Paper nominee.

- GONZÁLEZ-BRENES, J.P. and MOSTOW, J. 2010. Predicting Task Completion from Rich but Scarce Data. In *Proceedings of the 3rd International Conference on Educational Data Mining*, Pittsburgh, PA, June 11-13, 2010, R.S.J.D. BAKER, A. MERCERON and P.I.J. PAVLIK, Eds., 291-292.
- GONZÁLEZ-BRENES, J.P. and MOSTOW, J. 2011. Classifying Dialogue in High-Dimensional Space. *ACM Transactions on Speech and Language Processing* 7(3).
- GUNAWARDANA, A., MAHAJAN, M., ACERO, A. and PLATT, J.C. 2005. Hidden conditional random fields for phone classification. In *9th European Conference on Speech Communication and Technology - Interspeech* Lisboa, Portugal, 2005.
- HAJDINJAK, M. and MIHELIC, F. 2006. The PARADISE evaluation framework: Issues and findings. *Computational Linguistics* 32(2), 263-272.
- HUANG, X., ALLEVA, F., HON, H.W., HWANG, M.Y., LEE, K.F. and ROSENFELD, R. 1993. The SPHINX-II speech recognition system: an overview. *Computer Speech & Language* 7(2), 137-148.
- LODHI, H., SAUNDERS, C., SHAWE-TAYLOR, J., CRISTIANINI, N. and WATKINS, C. 2002. Text classification using string kernels. *J. Mach. Learn. Res.* 2, 419-444.
- LOURADOUR, J., DAOUDI, K. and BACH, F. 2006. SVM Speaker Verification using an Incomplete Cholesky Decomposition Sequence Kernel. In *Speaker and Language Recognition Workshop, 2006. IEEE Odyssey 2006: The*, 28-30 June 2006, 2006, 1-5.
- MÖLLER, S., ENGELBRECHT, K.P. and SCHLEICHER, R. 2008. Predicting the quality and usability of spoken dialogue services. *Speech Communication* 50(8-9), 730-744.
- MÖLLER, S., SMEELE, P., BOLAND, H. and KREBBER, J. 2007. Evaluating spoken dialogue systems according to de-facto standards: A case study. *Computer Speech and Language* 21(1), 26 - 53.
- MOSTOW, J. and AIST, G. 2001. Evaluating tutors that listen: An overview of Project LISTEN. In *Smart Machines in Education*, K. FORBUS and P. FELTOVICH, Eds. MIT/AAAI Press, Menlo Park, CA, 169-234.
- MURPHY, K. 2012, in preparation. *Machine Learning: a Probabilistic Perspective*. MIT Press.
- NG, A.Y. 2004. Feature selection, $\|\cdot\|_1$ vs. $\|\cdot\|_2$ regularization, and rotational invariance, ICML '04: Proceedings of the Twenty-first International Conference on Machine learning. ACM, Banff, Alberta, Canada, 78-86.
- OBOZINSKI, G., TASKAR, B. and JORDAN, M.I. 2007. Multi-task feature selection, The Workshop of Structural Knowledge Transfer for Machine Learning in the 23rd International Conference on Machine Learning (ICML), Pittsburgh, PA.
- QUATTTONI, A., WANG, S., MORENCY, L.P., COLLINS, M. and DARRELL, T. 2007. Hidden conditional random fields. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 29(10), 1848-1852.
- RABINER, L.R. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77(2), 257-286.
- REIGELUTH, C.M.A.S., F. S. (Editor), 1983. *The Elaboration Theory of Instruction*. Instructional Design Theories and Models: An Overview of their Current States. Lawrence Erlbaum, Hillsdale, NJ.
- SINGH, S.P., KEARNS, M.J., LITMAN, D.J. and WALKER, M.A. 1999. Reinforcement Learning for Spoken Dialogue Systems, Advances in Neural Information Processing Systems. MIT Press, Cambridge, MA, 956-962.
- STOLCKE, A., RIES, K., COCCARO, N., SHRIBERG, E., BATES, R., JURAFSKY, D., TAYLOR, P., MARTIN, R., ESS-DYKEMA, C.V. and METEER, M. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics* 26(3), 339-373.
- SY BOR, W. 2006. Hidden Conditional Random Fields for Gesture Recognition, 2006, Q. ARIADNA, M. LOUIS-PHILIPPE, D. DAVID and D. TREVOR, Eds., 1521-1527.
- WALKER, M., KAMM, C. and LITMAN, D. 2001. Towards developing general models of usability with PARADISE. *Natural Language Engineering* 6(3), 363-377.
- WALLACH, H.M. 2006. Topic modeling: beyond bag-of-words, Proceedings of the 23rd international conference on Machine learning. ACM, Pittsburgh, Pennsylvania, 977-984.
- XING, Z., PEI, J. and KEOGH, E. 2010. A brief survey on sequence classification. *SIGKDD Explor. Newsl.* 12(1), 40-48.
- ZOU, H. and HASTIE, T. 2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 67(2), 301-320.

Automatically Detecting a Student's Preparation for Future Learning: Help Use is Key

RYAN S.J.D. BAKER, SUJITH M. GOWDA

Worcester Polytechnic Institute

AND

ALBERT T. CORBETT

Carnegie Mellon University

We present an automated detector that can predict a student's later performance on a paper test of preparation for future learning, a post-test involving learning new material to solve problems involving skills that are related but different than the skills studied in the tutoring system. This automated detector operates on features of student learning and behavior within a Cognitive Tutor for College Genetics. We show that this detector predicts preparation for future learning better than Bayesian Knowledge Tracing, a widely-used measure of student learning in Cognitive Tutors. We also find that this detector only needs limited amounts of student data (the first 20% of a student's data from a tutor lesson) in order to achieve a substantial proportion of its asymptotic predictive power.

Categories and Subject Descriptors: I 2.7 [Artificial Intelligence]

General Terms: Preparation for Future Learning, Cognitive Tutor, Educational Data Mining

Additional Key Words and Phrases:

1. INTRODUCTION

Over the previous two decades, knowledge engineering and educational data mining (EDM) methods (cf. Baker & Yacef, 2009; Romero et al., 2010) have led to increasingly precise models of students' knowledge as they use intelligent tutoring systems and other types of interactive learning environments. Modeling student knowledge has been a key theme in research in intelligent tutoring systems from its earliest days. Models of student knowledge have become successful at inferring the probability that a student knows a specific skill at a specific time, from the student's pattern of correct responses and non-correct responses (e.g. errors and hint requests) up until that time (cf. Corbett & Anderson, 1995; Martin & VanLehn, 1995; Pavlik et al., 2009). In recent years, the debate about how to best model student knowledge has continued, with attempts to explicitly compare the success of different models at predicting students' future correctness within the tutoring software (cf. Pavlik et al., 2009; Gong et al., 2010; Baker, Pardos, et al., in press).

However, the ultimate goal of tutoring systems is not to improve future performance within the system itself but to improve unassisted performance outside the system. Ideally, interactive learning environments should promote "robust" learning (Koedinger et al., under review) that is retained (better remembered) over time (Pavlik & Anderson, 2008), transfers to new situations (Singley & Anderson, 1989), and prepares students for future learning (termed "PFL") (Bransford & Schwartz, 1999). The difference between transfer and PFL is whether a student has the ability to use their existing knowledge in

Authors' addresses: Ryan S.J.d. Baker, Sujith M. Gowda, Department of Social Science and Policy Studies, Worcester Polytechnic Institute, 100 Institute Road, Worcester, MA USA 01609. E-mail: rsbaker@wpi.edu, sujithmg@wpi.edu; Albert T. Corbett, Human-Computer Interaction Institute, Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh PA USA 15213; E-mail : corbett@cmu.edu

new situations or new fashions (transfer), or whether a student acquires new knowledge more quickly or effectively, using their existing knowledge (PFL).

Historically, student modeling research has paid limited attention to modeling the robustness of student learning. To the extent that there has been attention to modeling the robustness of learning, it has focused on retention and transfer. For example, Pavlik and Anderson (2008) predict how long knowledge will be retained after learning within an ILE teaching foreign language vocabulary. Martin and VanLehn (1995) and Desmarais et al. (2006) predict whether student knowledge of one skill will transfer to another skill. Baker, Gowda, and Corbett (in press) predict student performance on a paper post-test of transfer. However, it can be argued that the most important form of robust learning is the ability to apply learned skills and concepts to support future learning outside of the context where those skills and concepts were learned (Bransford & Schwartz, 1999). Though studies have demonstrated that learning from some types of interactive learning environments can prepare students for future learning (Tan & Biswas, 2006; Chin et al., 2010) student models have not yet explicitly modeled PFL.

Within this paper, we present a model designed to predict student performance on a paper post-test of PFL, a post-test where the student reads instructional text to learn new problem-solving skills related but different to those in the tutor, and then applies those skills in problem-solving. Such a model could be used both to understand the conditions of robust learning, and to drive interventions designed to increase the robustness of learning for students who are learning the skills in the tutor, but in a shallow fashion.

This model is generated using a combination of feature engineering and linear regression, and is cross-validated at the student level (e.g. trained on one group of students and tested on other students). We compare this model to Bayesian Knowledge Tracing – a student model shown to predict post-test performance – and to a model trained to detect transfer, in order to see how well each model can predict preparation for future learning. As a student model predicting PFL will be most useful if it can be used to drive interventions fairly early during tutor usage, we also analyze how much student data is needed for the model to be accurate.

2. DATA SET

The data set used in the analyses presented here came from the Genetics Cognitive Tutor (Corbett et al., 2010). This tutor consists of 19 modules that support problem solving across a wide range of topics in genetics. Various subsets of the 19 modules have been piloted at 15 universities in North America. This study focuses on a tutor module that employs a gene mapping technique called *three-factor cross*, in which students infer the order of three genes on a chromosome based on offspring phenotypes, as described in (Baker, Corbett, et al 2010). In this laboratory study, 71 undergraduates enrolled in genetics or in introductory biology courses at Carnegie Mellon University used the three-factor cross module. The students engaged in Cognitive Tutor-supported activities for one hour in each of two sessions. All students completed standard three-factor cross problems in both sessions. During the first session, some students were assigned to complete other cognitive-tutor activities designed to support deeper understanding; however, no differences were found between conditions for any robust learning measure, so in this analysis we collapse across the conditions and focus solely on student behavior and learning within the standard problem-solving activities. The 71 students completed a total of 22,885 problem solving attempts across 10,966 problem steps in the tutor.

Post-tests, given by paper-and-pencil, consisted of four activities (cf. Baker, Corbett, et al., 2010). Three tests were given immediately after tutor usage: a straightforward problem-solving post-test, a transfer test, and a test of preparation for future learning. The fourth test was a delayed retention test. Within this paper we focus on predicting

performance on the test of preparation for future learning, requiring the student to learn new skills after using the tutor. The PFL test consisted of 2½ pages of instruction on the reasoning needed for an analogous, but more complex, four-factor cross gene mapping task, followed by a single four-factor cross problem for students to solve.

Students demonstrated successful learning in this tutor, with an average pre-test performance of 0.31 ($SD=0.18$), an average post-test performance of 0.81 ($SD=0.18$), and an average PFL performance of 0.89 ($SD=0.15$). The correlation between the problem-solving post-test and the PFL test was 0.41, suggesting that, although problem-solving skill and preparation for future learning were related, PFL may be predicted by more than just simply skill at problem-solving within this domain.

3. ANALYSIS OF MODEL USING CROSS VALIDATION

In this paper, we introduce a model that predicts each student's performance on a test of preparation for future learning (PFL), using a hybrid of data mining and knowledge engineering methods. Within this approach, a small set of features is selected based on past literature. Each feature is defined as the proportion of times a specific student behavior occurs in the log files. Within feature selection, the goodness criterion is the cross-validated correlation between an individual feature and each student's performance on the PFL test. Cross-validated correlation is computed between the predictions from each of the test folds and the actual PFL test scores; positive cross-validated correlation indicates the relationship is consistent between the training and test folds (but has no implication about the relationship's direction). By contrast, negative cross-validated correlation indicates that the models obtained from the training folds fail to predict the actual scores in the test folds. . Finally a model is trained on these features to predict each student's performance on the PFL test, using cross-validation.

We then compare this model to a baseline prediction of PFL, Bayesian Knowledge Tracing (BKT) (Corbett & Anderson, 1995) fit using brute force. Bayesian Knowledge-Tracing fit in this fashion has been previously shown to predict student post-test problem-solving performance reasonably well within the Genetics Tutor (Baker et al., 2010). As BKT accurately predicts problem-solving post-tests, and the PFL test was reasonably correlated to the problem-solving post-test in this study, BKT should correlate reasonably well to PFL. But the goal of a robust learning test such as PFL is to measure depth of understanding that may not be reflected solely in basic problem-solving skill, which is tracked by BKT. Hence, it may be possible to develop a detector based on other performance features that predicts PFL better than BKT, under cross-validation.

3.1 FEATURE ENGINEERING

The first step of our process was to engineer a set of features based on a combination of theory and prior work detecting related behaviors. Since we were predicting post-test performance, we focused on proportions of behavior across the period of use of the tutoring system (e.g. what proportion of time a student engaged in behavior N). Many of the features below depend on a continuous variable, such as pause duration (2, 3, 4) or probability of knowing a skill (1,6). For each such feature, we used a cut-off value to indicate the presence or absence of a behavior, in order to identify the incidence of specific behaviors hypothesized to be associated with robust learning. That is, we empirically determined (see section 3.2) a cut-off value that indicates the student behavior occurred (e.g. a long pause or low probability), rather than averaging the actual values (pause durations or probabilities). We tested the following features:

1. Help avoidance (Aleven et al, 2006), not requesting help on poorly known skills, and its converse, feature 1', requesting help on relatively poorly known skills

2. Long pauses after receiving bug messages (error messages given when the student's behavior indicates a known misconception) which may indicate self-explanation (cf. Chi et al., 1989) of the bug message, and its converse, 2', short pauses after receiving bug messages (indicating a failure to self-explain)
3. Long pauses after reading on-demand hint messages (potentially indicating deeper knowledge or self-explanation), and a related feature, 3', short pauses after reading the on-demand hint message
4. Long pauses after reading an on-demand hint message and getting the current action right (cf. Shih, Koedinger, & Scheines, 2008), and a related feature, 4', short pauses after reading an on-demand hint message and getting the current action right. Features 4 and 4' can be seen as sub-sets of features 3 and 3'.
5. Off-task behavior (Baker, 2007), and a related feature, 5', long pauses that are not off-task (may indicate self-explanation, or asking teacher for help – cf. Schofield, 1995)
6. Long pauses on skills assessed as known (may indicate continuing to self-explain even after proceduralization), and a related feature, 6', short pauses on skills assessed as known
7. Gaming the system (Baker et al., 2008), and a related feature, 7', fast actions that do not involve gaming
8. Contextual slip/carelessness (known to predict post-test problem-solving performance – Baker et al, 2010)
9. The presence of spikes during learning using the moment-by-moment learning model, which estimates the probability that the student learned a relevant skill at each step in problem solving (spikes in this model have been found to predict final knowledge in the tutor – cf. Baker, Goldstein, Heffernan, in press).

Five of these features showed positive cross-validated correlations between the individual feature and the students' performance on the PFL test: 1 (failing to request help on poorly-known skills), 3 (long pauses after reading hint messages), 6' (short pauses on skills assessed as known), 7' (fast actions that do not involve gaming), and 9 (spikiness in the moment-by-moment learning model). The exact definition of these features was:

- 1: Proportion of actions where the student has a probability under N of knowing the skill, according to Bayesian Knowledge Tracing (Corbett & Anderson, 1995), does not ask for help, and makes an error on their first attempt. Initial cut-off value of N = 60% probability.
- 3: Proportion of actions where the student asks for hint, and then makes their next action in over N seconds. Initial value of N = 5 seconds.
- 6': Proportion of actions where the student has a probability over 0.95 of knowing the skill, according to Bayesian Knowledge Tracing (Corbett & Anderson, 1995), and applies the skill in under N seconds. Initial value of N = 5 seconds.
- 7': Proportion of actions where the student enters an answer or requests a hint in under N seconds, but the action is not labeled as gaming, using a gaming detector previously trained on data from a high school algebra course (cf. Baker & de Carvalho, 2008 – where a single detector was trained on all lessons to maximize detector generalizability – cf. Baker et al., 2008). This detector has previously been shown to achieve a correlation over 0.3 to the post-test within another dataset from the same lesson in the Genetics Tutor. Initial value of N = 5 second.
- 9: Highest value of moment-by-moment learning model estimate (cf. Baker, Goldstein, & Heffernan, in press) for each skill, divided by the average moment-by-moment learning estimate for that skill, averaged across skills, for the student. This model infers student

learning at each problem step, and is initially trained using data from future student correctness within the tutor, but the model itself uses only data from the past.

As can be seen, four of these features depend on a threshold parameter, N; adjusting this parameter can result in very different behavior. In all three cases, we started with an initial plausible value of N, as given above. The following section discusses how these features were optimized later in the modeling process.

3.2 FEATURE OPTIMIZATION

We used brute-force grid search to find an optimal cut-off level for four of the above mentioned features (in grid search, values are tried for every step at the same interval – for instance 0.5 seconds, 1 second, 1.5 seconds, 2 seconds, etc.). Variables involving probabilities were searched at a grid size of 0.05; variables involving time were searched at a grid size of 0.5 seconds with the exception of feature 6', which was searched at a grid size of 5 seconds. After the generation of the features at different grids, we built one-parameter linear regression models predicting PFL from each feature using leave-out-one-cross-validation, in RapidMiner 4.6 (Mierswa et al., 2006). Cross-validated correlation was used as the goodness measure. Single-feature regression models fit on the whole data set and their associated cross-validated correlations are shown in Table I.

Many of the features, subsequent to optimization, changed meaning. For instance, feature 1, initially conceptualized as Help Avoidance, achieved optimal performance when help avoidance occurred on skills for which the probability the student knew the skill was ≤ 1 – that is to say, on all help. So feature 1 can be re-conceptualized as the help/error ratio, and specifically as MoreErrorsLessHelp. Similarly, feature 6' was initially conceptualized as short pauses on mastered skills, but the optimal performance was achieved when the cut-off for shortness was set to 55 seconds. Hence, feature 6' can be re-conceptualized as pauses which are not long, on mastered skills.

The feature most strongly associated with PFL was making errors rather than requesting help, which was negatively associated with PFL (cross-validated $r=0.356$). This feature may have multiple interpretations; for instance, it may be that these students learned less by avoiding help (cf. Aleven et al., 2006), perhaps learning less at a conceptual level as the tutor hints are fairly conceptual in nature. This may in turn have made these students less prepared for future learning in the same domain. Alternatively, it may be that these students are less successful at learning from text (or less motivated to learn from text), causing them both to avoid hints in the tutor, and to perform less well as reading the text needed to succeed on the future learning test. Distinguishing these two hypotheses is challenging, but may be a productive avenue for future research.

A second feature individually associated with PFL is spending less than a minute on skills assessed as known, which achieved a cross-validated r of 0.340. This feature suggests that relatively quick performance on known skills is indicative of robust learning. Similarly, non-gaming actions taking less than 4 seconds were positively correlated with PFL (cross-validated $r=0.166$).

Additionally, the spikiness of the moment-by-moment learning model, is positively associated with PFL, achieving a cross-validated r of 0.286. This finding suggests that if a student's learning more frequently occurs in relatively sudden "aha" moments, as compared to occurring more gradually, deeper learning is occurring.

Finally, spending more than 5.5 seconds to answer after receiving a hint was negatively correlated with PFL (cross-validated $r=0.230$). This result is unexpected, as pauses after reading hints have previously been shown to be positively correlated with post-test performance (e.g. Shih, Koedinger, & Scheines, 2008). One possible explanation is that pausing after reading help is generally beneficial, but that the students

Table I. Linear regression model predicting the PFL test using single optimized features.

Feature	PFL=	Cross-validated r
1. MoreErrorsLessHelp	-0.980 * MoreErrorsLessHelp + 1.01	0.356
3. Long pauses After Hint with Time > 5.5	-2.445 * LongPausesAfterHint + 0.912	0.230
6'. Non-Long Mastered Skill with Time < 55	+ 0.768 * Non-LongMasteredSkill + 0.296	0.340
7'. Fast Not Gaming with Time < 4	+0.340 * FastNotGaming + 0.739	0.166
9. Spikiness	+0.0083 * spikiness + 0.7773	0.286

who do so with high frequency are the students who are struggling. Hence, it may be an interesting question for future research to examine whether there is a non-linear relationship between lengthy pauses after reading hints and PFL (and learning in general).

3.3 DETECTOR DEVELOPMENT

Given the set of optimal features, we developed linear regression models in RapidMiner 4.6 (Mierswa et al., 2006) using Forward Selection (Ramsey & Schafer, 1997), conducted by hand. In Forward Selection, the best single-parameter model is chosen, and then the parameter that most improves correlation (cross-validated in this case) is repeatedly added until no more parameters can be added which improve the correlation.

Within RapidMiner, feature selection was turned off, and each potential model was tested in a separate run – while this creates some risk of over-fitting (even given the use of cross-validation), it enables us to determine how well a specific set of features predicts PFL. Keeping feature selection on would result in some features being filtered out for some sub-sets of the data, making it harder to infer how well a specific set of features predicts PFL. As before, Leave-One-Out Cross-Validation (LOOCV) was used to reduce the risk of over-fitting, and the goodness metric used was the Pearson correlation between the predictions and each student's performance on the PFL test. In addition, as an additional control on over-fitting, we did a first pass where we eliminated all features that, taken individually, had cross-validated correlation below zero. We give differences in cross-validated correlation rather than statistical significance tests, as a measure of generalizability; differences in non-cross-validated correlations of non-nested models have low statistical power – (Cohen, 1988) – and comparing cross-validated correlations is a redundant test – (cf. Efron & Gong, 1983).

The best model, using the optimal feature cut-offs, and fit to all data (not cross-validated; cross-validation produces one model per each of the 71 training sets) was as follows:

$$PFL = -0.7837 * MoreErrorsLessHelp(1) -1.3042 * LongPausesAfterHints(3) + 0.9936$$

3.4 DETECTOR GOODNESS

The overall correlation of this model to the PFL test was 0.360, only very slightly better than feature 1 alone (0.356). By comparison, fitting a baseline model consisting of Bayesian Knowledge Tracing post-test predictions (using brute force – cf. Baker et al., 2010) to the PFL test results, under LOOCV, achieved a correlation of 0.285 to the PFL test. This is a reasonable baseline, as Bayesian Knowledge-Tracing has previously been shown to predict the post-test well in this tutor (Baker et al., 2010) as well as in general (e.g. Corbett & Anderson, 1995; Corbett & Bhatnagar, 1997), and performance on the PFL test was correlated reasonably well to performance on the post-test in this data set (non-cross-validated $r=0.41$). Hence, the optimal feature model appears to perform

substantially better at predicting PFL than this reasonable baseline, although there is still likely to be substantial room for improvement.

Interestingly, if the post-test and the PFL detector are used together in linear regression to predict the PFL test, the cross-validated correlation is 0.391. However, if the Bayesian Knowledge Tracing estimates and the PFL detector are used together in this way to predict the PFL test, the cross-validated correlation drops to 0.309. This result suggests that, despite the PFL detector's reasonable effectiveness at detecting PFL, the paper post-test still captures a small amount of variance in students' preparation for future learning which is not yet detectable from student behavior in the tutor software. Furthermore, this additional predictive power is separate from the assessment of student knowledge made by Bayesian Knowledge Tracing (since combining BKT with the PFL detector does not lead to better prediction).

As another test of the PFL detector's unique predictive power, we can compare it to another model of robust learning. In past work, we have developed a model that predicts transfer (i.e., the application of problem-solving knowledge to novel but related problems without additional instruction), using the same overall method that was used to develop the PFL detector (Baker et al., in press). That model is:

$$\text{Transfer} = -1.20 * \text{HelpAvoidance}(1) - 14.764 * \text{FastAfterBugs}(2') + 0.234 * \text{FastNotGaming}(7') + 0.832$$

Though there is considerable correlation (0.520) between the transfer test and the PFL test, the transfer detector did not do as well as the PFL detector at predicting PFL. With no re-fitting, the transfer detector achieved a cross-validated correlation of 0.273 to the PFL-test, comparable to BKT's performance at predicting PFL, and lower than the PFL detector's cross-validated performance of 0.360. This result suggests that PFL, to at least a moderate extent, is associated with different student behavior in the tutor than transfer is. These results, though somewhat small in absolute terms, are fairly large in relative terms (0.360 is 32% higher than 0.273), suggesting that it is unlikely this difference is solely due to noise in the two test measures.

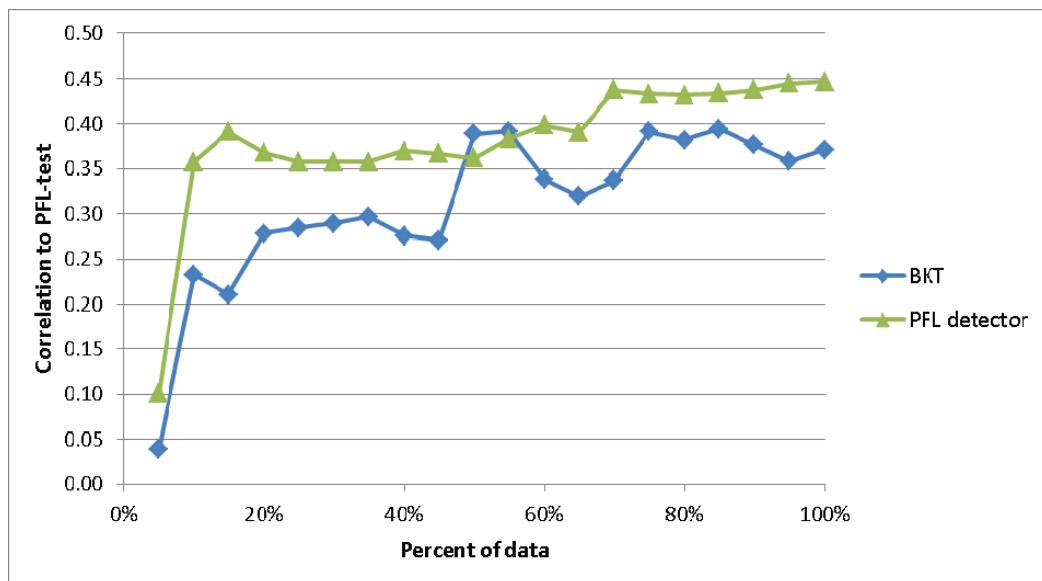


Fig.1 Predicting PFL with first N percent of the data.

4. ANALYSIS OF MODEL FOR USE IN RUNNING TUTOR

One potential criticism of models developed using proportions of behavior across entire episodes of tutor use is that the models, in their initial form, may not be usable in a running tutor. Bayesian Knowledge Tracing makes a prediction after each problem-solving step, which can be used to drive Cognitive Mastery Learning (Corbett & Anderson, 1992). If an entire tutor lesson worth of data is required for accurate inference, the detector may have low usefulness for intervention.

However, it is possible to make a version of the model that can be used in a running tutor. Similar to the way that Bayesian Knowledge Tracing makes a prediction after each problem-solving step, it is possible to take the data up to a specific problem step and attempt to make an overall inference about the probability of PFL, using only the data collected up until that point. In other words, the features used in the model can be computed at any time, using the data collected so far. In this section, we investigate how much data is needed for the model to make accurate predictions within this data set, comparing our model's predictive power to Bayesian Knowledge-Tracing, when both are given limited data.

Our first step in this process is to construct subsets of data containing the first N percent of each student's interactions within the tutor. We use every increment of 5% -- e.g. a subset with the first 5% of each student's data (not taking skills into account – e.g. data from some skills may not be present in the first 5%), a subset with the first 10% of each student's data, a subset with the first 15% of each student's data, up to 100%. This gives us 20 data sets. We then compute the optimal features discussed in section 3 for each subset of data. Next, we apply the PFL prediction model generated using the full data set (e.g. we do not refit the models for the new data sets). We also apply Bayesian Knowledge Tracing on the limited data sets without re-fitting the BKT parameter estimates. After obtaining the predictions we compute the correlation between each of the predictions and each student's performance on the PFL test. Cross-validation is not used, as the model is not being re-fit in either case.

Figure 1 shows the graph with x-axis as percent of data and y-axis as the correlation to the PFL test. The graph depicts the predictive performance of the PFL prediction model and BKT based on having the first N percent of the data. From the graph we can see that the PFL prediction model performs substantially better than BKT for small amounts of data. For instance, with only the first 20% of the data, the PFL prediction model achieves a solid correlation of 0.368 to the PFL test, while the BKT model achieves a weaker correlation of 0.278. These findings suggest that it may be possible to use the PFL prediction model to drive interventions, from very early in tutor usage.

5. DISCUSSION AND CONCLUSIONS

Within this paper, we have presented a model which can predict with reasonable accuracy how well a student will perform on a post-test measuring how well the student is prepared for future learning (PFL), within a Cognitive Tutor for College Genetics. We find that this model achieves decent cross-validated prediction of this PFL post-test, and achieves better cross-validation prediction than Bayesian Knowledge Tracing, a measure of skill learning within the tutor software, or a detector trained to detect transfer. Furthermore, we find that the PFL detector achieves a large proportion of its predictive power by the time the student has completed 20% of the tutor software, suggesting that the PFL detector can be used to drive intervention early enough to influence overall learning. Overall, we view this detector as a potential step towards educational software that can predict and respond automatically to differences in the robustness of student learning, an important complement to ongoing research on designing educational

software that promotes preparation for future learning (Tan & Biswas, 2006; Chin et al., 2010).

This model is based on the proportion of time the student engages in long pauses after requesting help (cf. Shih et al., 2008), and the ratio of help requests to errors, with more help use associated with better preparation for future learning, but lengthy pauses after help requests associated with poorer preparation for future learning. Both of these features indicate that the use of help is particularly essential for preparing students for future learning. Past studies have found mixed relationships between help and domain learning (cf. Aleven et al., 2003, 2006; Beck et al., 2008); this analysis, however, suggests that help use may under certain conditions lead to robust forms of learning that are not captured by typical metrics of in-tutor performance (e.g. Beck et al., 2008) and problem-solving post-tests (e.g. Aleven et al., 2006). We recommend that future research on help-seeking and learning consider measures of preparation for future learning of new skills and concepts to a greater degree.

ACKNOWLEDGEMENTS

This research was supported via grant “Empirical Research: Emerging Research: Robust and Efficient Learning: Modeling and Remediating Students’ Domain Knowledge”, National Science Foundation award number #DRL-0910188, via grant “Promoting Robust Understanding of Genetics with a Cognitive Tutor that Integrates Conceptual Learning with Problem Solving”, Institute of Education Sciences award number #R305A090549, and by the Pittsburgh Science of Learning Center, NSF award number #SBE-0836012.

REFERENCES

- ALEVEN, V., MCLAREN, B., ROLL, I., & KOEDINGER, K. (2006). Toward meta-cognitive tutoring: A model of help seeking with a Cognitive Tutor. *International Journal of Artificial Intelligence and Education*, 16, 101-128.
- BAKER, R.S.J.D. (2007) Modeling and Understanding Students' Off-Task Behavior in Intelligent Tutoring Systems. *Proceedings of ACM CHI 2007: Computer-Human Interaction*, 1059-1068.
- BAKER, R.S.J.D., CORBETT, A.T., ALEVEN, V. (2008) More Accurate Student Modeling Through Contextual Estimation of Slip and Guess Probabilities in Bayesian Knowledge Tracing. *Proceedings of the 9th International Conference on Intelligent Tutoring Systems*, 406-415.
- BAKER, R.S.J.D., CORBETT, A.T., GOWDA, S.M., WAGNER, A.Z., MACLAREN, B.M., KAUFFMAN, L.R., MITCHELL, A.P., GIGUERE, S. (2010) Contextual Slip and Prediction of Student Performance After Use of an Intelligent Tutor. *Proceedings of the 18th Annual Conference on User Modeling, Adaptation, and Personalization*, 52-63
- BAKER, R.S.J.D., CORBETT, A.T., ROLL, I., KOEDINGER, K.R. (2008) Developing a Generalizable Detector of When Students Game the System. *User Modeling and User-Adapted Interaction*, 18, 3, 287-314.
- BAKER, R.S.J.D., DE CARVALHO, A. M. J. A. (2008) Labeling Student Behavior Faster and More Precisely with Text Replays. *Proceedings of the 1st International Conference on Educational Data Mining*, 38-47.
- BAKER, R.S.J.D., GOLDSTEIN, A.B., HEFFERNAN, N.T. (in press) Detecting Learning Moment-by-Moment. To appear in *International Journal of Artificial Intelligence in Education*.
- BAKER, R.S.J.D., GOWDA, S.M., CORBETT, A.T. (in press) Towards predicting future transfer of learning. To appear in the Proceedings of the 15th International Conference on Artificial Intelligence in Education.
- BAKER, R.S.J.D., PARDOS, Z.A., GOWDA, S.M., NOORAEI, B.B., HEFFERNAN, N.T. (in press) Ensembling Predictions of Student Knowledge within Intelligent Tutoring Systems. To appear in *Proceedings of the 19th International Conference on User Modeling, Adaptation, and Personalization*.
- BRANSFORD, J.D., SCHWARTZ, D.L. (1999) Rethinking transfer: a simple proposal with multiple implications. *Review of Research in Education*, 24, 61-100.
- CHI, M.T.H., BASSOK, M., LEWIS, M.W., REIMANN, P., GLASER, R. (1989) Self-explanations: how students study and use examples in learning to solve problems. *Cognitive Science*, 13, 145-182.
- CHIN, D.B., DOHMHEN, I.M., CHENG, B.H., OPPEZZO, M.A., CHASE, C.C., SCHWARTZ, D. L. (2010) Preparing Students for Future Learning with Teachable Agents. *Educational Technology Research and Development*, 58 (6), 649-669.
- COHEN, J. (1988) Statistical Power Analysis for the Behavioral Sciences, 2nd Edition. Hillsdale, NJ: LEA.

- CORBETT A., BHATNAGAR A. (1997). Student Modeling in the ACT Programming Tutor: Adjusting Procedural Learning Model with Declarative Knowledge. User Modeling: Proceedings of the 6th International Conference, 243-254.
- CORBETT, A.T., ANDERSON, J.R. (1992) Student modeling and mastery learning in a computer-based programming tutor. Proceedings of the International Conference on Intelligent Tutoring Systems, 413-420.
- CORBETT, A.T., ANDERSON, J.R. (1995) Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. User Modeling and User-Adapted Interaction, 4, 253-278.
- CORBETT A.T., MACLAREN, B., KAUFFMAN, L., WAGNER, A., & JONES, E. (2010). A Cognitive Tutor for genetics problem solving: Learning gains and student modeling. Journal of Educational Computing Research, 42, 219-239.
- DESMARAIS, M. C., MESHKINFAM, P., AND GAGNON, M. Learned Student Models with Item to Item Knowledge Structures. User Modeling and User-Adapted Interaction, 16 5 (2006), 403-434
- EFRON, B. & GONG, G. (1983). A leisurely look at the bootstrap, the jackknife, and cross-validation. American Statistician, 37, 36-48
- GONG, Y., BECK, J. E., HEFFERNAN, N. T. (2010). Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting Procedures. Proceedings of the 10th International Conference on Intelligent Tutoring Systems, 35-44.
- MARTIN, J., VANLEHN, K. (1995) Student assessment using Bayesian Nets. *International Journal of Human-Computer Studies*, 42, 575-591.
- MIERSWA, I., WURST, M., KLINKENBERG, R., SCHOLZ, M., EULER, T. (2006) YALE: rapid prototyping for complex data mining tasks. Proceedings of the 12th ACM SIGKDD international conference on Knowledge Discovery and Data Mining, 935-940.
- PAVLIK, P.I., ANDERSON, J.R. (2008). Using a Model to Compute the Optimal Schedule of Practice. Journal of Experimental Psychology: Applied, 14 (2), 101-117.
- PAVLIK, P.I., CEN, H., KOEDINGER, J.R. (2009) Performance Factors Analysis – A New Alternative to Knowledge Tracing. Proceedings of the 14th International Conference on Artificial Intelligence in Education, 531-540.
- RAMSEY, R.L., SCHAFER, D.W. (1997) The Statistical Sleuth. Belomnt, CA: Wadsworth Publishing.
- SCHOFIELD, J. W. (1995). Computers and Classroom Culture. Cambridge, UK: Cambridge University Press.
- SHIH, B., KOEDINGER, K.R. AND SCHEINES, R. (2008) A response time model for bottom-out hints as worked examples. Proceedings of the 1st International Conference on Educational Data Mining, 117-126.
- SINGLEY, M.K., ANDERSON, J.R. (1989) The Transfer of Cognitive Skill. Cambridge, MA: Harvard University Press.
- TAN, J., BISWAS, G: The Role of Feedback in Preparation for Future Learning: A Case Study in Learning by Teaching Environments. Intelligent Tutoring Systems 2006: 370-381

Ensembling Predictions of Student Post-Test Scores for an Intelligent Tutoring System

ZACHARY A. PARDOS, SUJITH M. GOWDA, RYAN S.J.D.

BAKER, NEIL T. HEFFERNAN

Worcester Polytechnic Institute

Over the last few decades, there have been a rich variety of approaches towards modeling student knowledge and skill within interactive learning environments. There have recently been several empirical comparisons as to which types of student models are better at predicting future performance, both within and outside of the interactive learning environment. A recent paper (Baker et al., *in press*) considers whether ensembling can produce better prediction than individual models, when ensembling is performed at the level of predictions of performance within the tutor. However, better performance was not achieved for predicting the post-test. In this paper, we investigate ensembling at the post-test level, to see if this approach can produce better prediction of post-test scores within the context of a Cognitive Tutor for Genetics. We find no improvement for ensembling over the best individual models and we consider possible explanations for this finding, including the limited size of the data set.

Categories and Subject descriptors: I 2.7 [Artificial Intelligence]

General Terms: Student modeling, ensemble methods, Bayesian Knowledge-Tracing, Performance Factors Analysis, Cognitive Tutor

Additional Key Words and Phrases:

1. INTRODUCTION

In recent years, there has been a vigorous debate as to which approach for assessing student knowledge and skill within interactive learning environments achieves the most precise assessment of students' latent knowledge and skills. This debate has been particularly vigorous for the relatively simple case of intelligent tutoring systems where each item is related to a single skill, and for which the item-skill mapping has been well-developed (the issue of how to develop these item-skill mappings is of course a key issue in its own right – cf. Barnes, Bitzer, & Vouk, 2005). For this problem, recent comparisons have studied the difference between variants of Bayesian Knowledge Tracing (Baker, Corbett, & Aleven 2008), the differences between variants of Bayesian Knowledge Tracing and Performance Factors Analysis (Pavlik, Cen, & Koedinger; 2009; Gong, Beck, & Heffernan, 2010), and the differences between these algorithms and baseline approaches such as average performance and lookups based on the correctness of the previous three actions (Baker et al., *in press*). However, these comparisons have often had contradictory results, likely due to differences between the tutoring systems, populations studied, and exact methods used to make comparisons.

Based on the contradictory results of these comparisons, Baker et al. (*in press*) proposed that it might be more productive to ensemble the available algorithms (cf. Dietterich, 2000) rather than attempting to determine which algorithm is best. Within ensemble methods, multiple models are integrated into a single predictor. Ensemble

Authors' addresses: Zachary A. Pardos, Sujith M. Gowda, Ryan S.J.d. Baker, Neil T. Heffernan, Department of Social Science and Policy Studies, Department of Computer Science, Worcester Polytechnic Institute, 100 Institute Road, Worcester, MA USA 01609. E-mail: zpardos@wpi.edu, sujithmg@wpi.edu, rsbaker@wpi.edu, nth@wpi.edu

methods often achieve better performance than single algorithms, since they are able to leverage each algorithm's strengths in different contexts (Dietterich, 2000; Niculescu-Mizil et al., 2009). In their attempt to do this, Baker and colleagues selected nine algorithms predicting student performance and latent knowledge within intelligent tutoring systems, and evaluated the success of simple linear and logistic ensembling methods in data from a Cognitive Tutor for Genetics. This paper conducted ensembling at the level of individual student actions within the tutor (e.g. each model's prediction of latent student knowledge at a given time within the tutor was ensembled into a single prediction of latent student knowledge). The results were mixed; depending on what assumptions were used, ensemble models were either slightly more successful or slightly less successful than the best single model. In addition, multiple individual models were more successful at predicting post-test scores than ensembling conducted in this fashion.

However, it is known that tutor performance often does not perfectly match post-test performance, and that models trained to predict performance within the tutor software often over-predict post-test performance (Corbett & Anderson, 1995). Predicting post-test scores is important, as it provides a test not just of what students can do within the tutor, but also what knowledge they transfer outside of the tutor software. Recent analyses have also suggested that combining assessment of student knowledge with assessments of student slipping/carelessness can lead to more accurate prediction of post-test performance (Baker et al., 2010). Hence, it may be possible to use ensemble methods to better predict post-test performance by ensembling predictions of post-test scores, including predictions of post-test related constructs such as slipping, instead of ensembling predictions of within-tutor performance. To this end, within this paper, we compare the predictive power of ensemble models and single models for predicting post-test performance among students learning from a Cognitive Tutor.

2. STUDENT MODELS USED

2.1 Bayesian Knowledge-Tracing

Corbett & Anderson's (1995) Bayesian Knowledge Tracing model is one of the most popular methods for estimating students' knowledge. It underlies the Cognitive Mastery Learning algorithm used in Cognitive Tutors for Algebra, Geometry, Genetics, and other domains (Koedinger & Corbett, 2006).

The canonical Bayesian Knowledge Tracing (BKT) model assumes a two-state learning model: for each skill/knowledge component the student is either in the learned state or the unlearned state. At each opportunity to apply that skill, regardless of their performance, the student may make the transition from the unlearned to the learned state with *learning* probability $P(T)$. The probability of a student going from the learned state to the unlearned state (i.e. forgetting a skill) is fixed at zero. A student who knows a skill can either give a correct performance, or *slip* and give an incorrect answer with probability $P(S)$. Similarly, a student who does not know the skill may *guess* the correct response with probability $P(G)$. The model has another parameter, $P(L_0)$, which is the probability of a student knowing the skill from the start. After each opportunity to apply the rule, the system updates its estimate of student's knowledge state, $P(L_n)$, using the evidence from the current action's correctness and the probability of learning:

$$P(L_{n-1}|Correct_n) = \frac{P(L_{n-1}) * (1 - P(S))}{P(L_{n-1}) * (1 - P(S)) + (1 - P(L_{n-1})) * (P(G))}$$

$$P(L_{n-1}|Incorrect_n) = \frac{P(L_{n-1}) * P(S)}{P(L_{n-1}) * P(S) + (1 - P(L_{n-1})) * (1 - P(G))}$$

$$P(L_n|Action_n) = P(L_{n-1}|Action_n) + ((1 - P(L_{n-1}|Action_n)) * P(T))$$

The four parameters of BKT, $(P(L_0), P(T), P(S)$, and $P(G)$, are learned from existing data, historically using curve-fitting (e.g. Corbett & Anderson, 1995), but more recently using expectation maximization (*BKT-EM*) (Chang et al., 2006) or brute force/grid search (*BKT-BF*) (cf. Baker et al., 2010; Pardos & Heffernan, 2010). Within this paper we use BKT-EM and BKT-BF as two different models in this study. Within BKT-BF, for each of the 4 parameters all potential values at a grain-size of 0.01 are tried across all the students (for e.g.: 0.01 0.01 0.01 0.01, 0.01 0.01 0.01 0.02, 0.01 0.01 0.01 0.03..... 0.99 0.99 0.3 0.1). The sum of squared residuals (SSR) is minimized. For *BKT-BF*, the values for Guess and Slip are bounded in order to avoid the “model degeneracy” problems that arise when performance parameter estimates rise above 0.5 (Baker, Corbett, & Aleven, 2008). For *BKT-EM* the parameters were unbounded and initial parameters were set to a $P(G)$ of 0.14, $P(S)$ of 0.09, $P(L_0)$ of 0.50, and $P(T)$ of 0.14, a set of parameters previously found to be the average parameter values across all skills in modeling work conducted within a different tutoring system.

In addition, we include three other variants on BKT. The first variant changes the data set used during fitting. BKT parameters are typically fit to all available students’ performance data for a skill. It has been argued that if fitting is conducted using only the most recent student performance data, more accurate future performance prediction can be achieved than when fitting the model with all of the data (Pardos & Heffernan, in press). In this study, we included a BKT model trained only on a maximum of the 15 most recent student responses on the current skill, *BKT-Less Data* (Nooraei B et al., in press).

The second variant, the *BKT-CGS* (Contextual Guess and Slip) model, is an extension of BKT (Baker, Corbett, & Aleven, 2008). In this approach, Guess and Slip probabilities are no longer estimated for each skill; instead, they are computed each time a student attempts to answer a new problem step, based on machine-learned models of guess and slip response properties in context (for instance, longer responses and help requests are less likely to be slips). The same approach as in (Baker, Corbett, & Aleven, 2008) is used to create the model, where 1) a four-parameter BKT model is obtained (in this case *BKT-BF*), 2) the four-parameter model is used to generate labels of the probability of slipping and guessing for each action within the data set, 3) machine learning is used to fit models predicting these labels, 4) the machine-learned models of guess and slip are substituted into Bayesian Knowledge Tracing in lieu of skill-by-skill labels for guess and slip, and finally 5) parameters for $P(T)$ and $P(L_0)$ are fit.

Recent research has suggested that the average Contextual Slip values from this model, combined in linear regression with standard BKT, improves prediction of post-test performance compared to BKT alone (Baker et al., 2010). Hence, we include average *Contextual Slip* so far as an additional potential model.

The third BKT variant, the *BKT-PPS* (Prior Per Student) model, breaks from the standard BKT assumption that each student has the same incoming knowledge, $P(L_0)$. This individualization is accomplished by modifying the prior parameter for each student with the addition of a single node and arc to the standard BKT model (Pardos &

Heffernan, 2010). The model can be simplified to only model two different student knowledge priors, a high and a low prior. No pre-test needs to be administered to determine which prior the student belongs to; instead their first response can be used. If a student answers their first question of the skill incorrectly they are assumed to be in the low prior group. If they answer correctly, they assumed to be in the high prior group. The prior of each group can be learned or it can be set *ad-hoc*. The intuition behind the *ad-hoc* high prior, conditioned upon first response, is that it should be roughly 1 minus the probability of guess. Similarly, the low prior should be equivalent to the probability of slip. Using PPS with a low prior value of 0.10 and a high value of 0.85 has been shown to lead to improved accuracy at predicting student performance (Pardos & Heffernan, 2010).

2.2 Tabling

A very simple baseline approach to predicting a student's performance, given his or her past performance data, is to check what percentage of students with that same pattern of performance gave correct answer to the next question. That is the key idea behind the student performance prediction model called *Tabling*.

In the training phase, a table is constructed for each skill: each row in that table represents a possible pattern of student performance in n most recent data points. For $n = 3$ (which is the table size used in this study), we have 8 rows: 000, 001, 010, 011, 100, 101, 110, 111. (0 and 1 represent incorrect and correct responses respectively.) For each of those patterns we calculate the percentage of correct responses immediately following the pattern. For example, if we have 47 students that answered 4 questions in a row correctly (111 1), and 3 students that after answering 3 correct responses, failed on the 4th one, the value calculated for row 111 is going to be 0.94 (47/(47+3)). When predicting a student's performance, this method simply looks up the row corresponding to the 3 preceding performance data, and uses the percent correct value as its prediction.

2.3 Performance Factor Analysis

Performance Factors Analysis (PFA) (Pavlik, Cen, & Koedinger, 2009) is a logistic model, an elaboration of the Rasch item response model, which predicts student performance based on the student's number of prior failures f and successes s for that skill, with skill-specific weightings γ and ρ for failures and successes. PFA also includes an overall difficulty parameter β for each skill or item, depending on the formulation. Within this paper, we fit β at the skill level. The PFA equation is:

$$m(i, j \in KCs, s, f) = \beta_j + \sum(\gamma_j S_{ij} + \rho_j F_{ij})$$

2.4 CFAR

CFAR, which stands for "Correct First Attempt Rate", is an extremely simple algorithm for predicting student knowledge and future performance, utilized by the winners of the educational data KDD Cup in 2010 (Yu et al., 2010). The prediction of student performance on a specific knowledge component (KC) is the student's average correctness on that KC, up until the current point.

3. GENETICS DATA SET

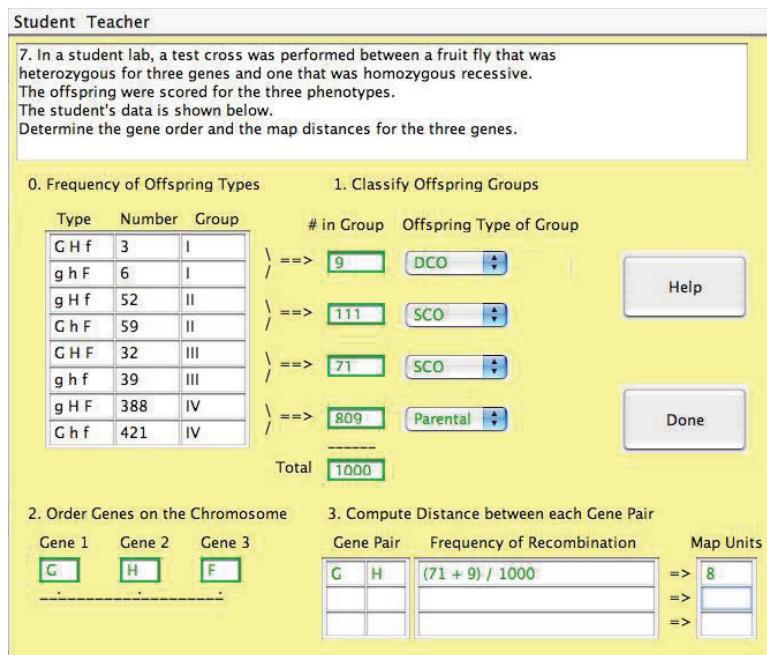


Fig.1 The Three-Factor Cross lesson of the Genetics Cognitive Tutor

The dataset contains the results of in-tutor performance data of 76 students on 9 different skills, with data from a total of 23,706 student actions (entering an answer or requesting help). This data was taken from a Cognitive Tutor for Genetics (Corbett et al., 2010). This tutor consists of 19 modules that support problem solving across a wide range of topics in genetics (Mendelian transmission, pedigree analysis, gene mapping, gene regulation and population genetics). Various subsets of the 19 modules have been piloted at 15 universities in North America.

This data set is drawn from a Cognitive Tutor lesson on three-factor cross, shown in Figure 1. In three factor-cross problems, two organisms are bred together, and then the patterns of phenotypes and genotypes on a chromosome are studied. In particular, the interactions between three genes on the same chromosome are studied. During meiosis, segments of the chromosome can “cross over,” going from one paired chromosome to the other, resulting in a different phenotype in the offspring than if the crossover did not occur. Within this tutor lesson, the student identifies, within the interface, the order and distance between the genes on the chromosome by looking at the relative frequency of each pattern of phenotypes in the offspring. The student also categorizes each phenotype in terms of whether it represents the same genotype as the parents (e.g. no crossovers during meiosis), whether it represents a single crossover during meiosis, or whether it represents two crossovers during meiosis.

In this study, 76 undergraduates enrolled in a genetics course at Carnegie Mellon University used the three-factor cross module as a homework assignment. The 76 students completed a total of 23,706 problem solving attempts across 11,582 problem steps in the tutor. On average, each student completed 152 problem steps ($SD=50$). In the first session, students were split into four groups with a 2x2 design; half of students spent half their time in the first session self-explaining worked examples; half of students spent half their time in a forward modeling activity. Within this paper, we focus solely on behavior logged within the problem-solving activities, and we collapse across the original four conditions.

The post-test, given on paper-and-pencil, consisted of four activities: a straightforward problem-solving post-test, a transfer test, a test of preparation for future learning, and a delayed retention test. Each of the two problems on the problem-solving test (administered at pre-test and post-test) consisted of 11 steps involving 7 of the 9 skills in the Three-Factor Cross tutor lesson, with two skills applied twice in each problem and one skill applied three times. The average performance on the pre-test was 0.33, with a standard deviation of 0.2. The average performance on the post-test was 0.83, with a standard deviation of 0.19. This provides evidence for substantial learning within the tutor, with an average pre-post gain of 0.50.

4. ENSEMBLE METHODS

The premise behind ensembling is to combine the prediction of different models such that the combination results in a more accurate prediction than any single model could produce. We evaluated six methods of combining post-test predictions to investigate the utility of ensembling with our dataset. The six methods evaluated were: Uniform averaging, linear regression, stepwise regression, stepwise model selection with uniform averaging, logistic regression and Random Forests (Brieman, 2001). More detail on the trade-offs between different methods for ensembling can be found in (Dietterich, 2000; Brown, Wyatt, & Tino, 2005). A description of each method is as follows:

Uniform averaging: Uses the mean of the 9 model predictions. While this is the most simple approach, it is also the only approach that is not susceptible to overfitting (since it does not use any form of training).

Linear regression: Fits coefficients to each model's prediction. This approach assumes that there is a linear weighting of models that can lead to better prediction. Predictions lower than zero are changed to zero and predictions higher than one are changed to one.

Stepwise regression: The same as linear regression except that this approach removes models that are not significantly contributing to a decrease in training set error.

Stepwise model selection with uniform averaging: Uses stepwise regression to remove bad models and then averages the prediction of the remaining models. This approach assumes that model selection could be useful but that fitting coefficients might overfit.

Logistic regression: Similar to linear regression except that coefficients are fit based on a logistic curve. The logistic function outputs probabilistic values (between 0 and 1).

Random Forests: Trains many decision trees, each based on a random sampling of models and random resampling of the data, and then averages each decision tree's prediction. This is the only ensemble technique we evaluate which combines model predictions non-linearly. Default MATLAB parameters were used with 20 trees.

Each ensembling method was evaluated with 5-fold cross validation, where four folds were used to train and one to test (note that cross-validating uniform averaging has no impact on goodness of fit, since no training occurs). The same fold assignments were used for this cross-validation as for training and testing the 9 individual models. For simplicity we decided not to use ensemble methods which have parameters that need to be tuned such as Neural Networks and Support Vector Machines. This would have added yet another level of cross-validation within the training set. The effectiveness of those methods is a topic for future exploration.

4.1 Using Ensemble Methods in the Real World

It is worth noting that, in evaluating these methods, different forms of cross-validation are required for ensembles versus for the individual models. The individual models are cross-validated at the action level, when the models are trained. They are not fit to the post-test performance, and thus are not cross-validated at this level. However, in ensembling, it is necessary to cross-validate, because fitting occurs at this level.

This form of cross-validation fits to how ensemble methods would typically be used for this problem in the real world. Typically, intelligent tutors use knowledge-tracing (or other assessment) model parameters trained based on data from a previous cohort of students. The tutor software then traces the current year's student responses and makes predictions about their knowledge on the post-test. In predicting post-test scores, an individual model's predictions can be used (and currently, this is the approach commonly used). If a researcher wanted to integrate multiple models in predicting the post-test, uniform averaging could be used post-hoc but with no fitting, in a direct fashion. Neither of these approaches risk over-fitting, as neither use the post-test data to make predictions. However, if the researcher wanted to use ensembling methods as discussed above, one way to do so without risk of over-fitting would be to conduct ensembling using a previous cohort's post-test data, and then to apply the ensembled model to the current cohort's data. The success of the ensembling approach relies on whether the weights learned from last year's data generalize to this year's data. If the ensemble selection is generalizable, it should perform better than uniform averaging and also better than the single model predictions, for the new population. In our analysis, the previous cohort's data is analogous to the four training folds of the 5-fold cross validation whereas the fifth fold, the test fold, represents the current cohort of students.

5. RESULTS

In predicting the post-test using individual model we calculate the $n+1$ predictions for each skill and student, by applying the rule, where time n equals the last student action in the tutor, and time $n+1$ equals the student's action after the last action in the tutor (e.g. their action on the post-test):

$$P(\text{correct}_{n+1}) = P(L_n) * (1 - P(S)) + (1 - P(L_n)) * P(G)$$

After applying the above rule, we account for the number of times each skill is used in the post-test. Of the eight skills in the tutor lesson, one is not exercised on the test, and is eliminated from the individual models predicting the post-test. Of the remaining seven skills, one is exercised three times, two are exercised twice and four skills are exercised once, in each of the post-test problems. In order to predict the post-test with maximal accuracy, we weight the tutor's prediction of student knowledge of each skill in line with the number of times it appears in the post-test. As each of the individual models can already predict performance on the $n+1$ step, the opportunity to practice after tutor usage, we do not fit any function to post-test using the individual models; hence the cross-validation gives the same results as not using cross-validation. We use RMSE (root mean squared error) and the Pearson correlation between the model predictions and post-test score, as the model estimates and the post-test scores are both numeric.

As seen in table I, the individual model with the best value of RMSE for predicting the post-test is CFAR, achieving an RMSE of 0.1636 and a correlation of 0.533 to the post-test. However, although BKT-LessData, BKT-EM, and BKT-BF perform slightly worse in terms of RMSE, achieving RMSEs between 0.1754 and 0.1834, each of them

achieves better correlation to the post-test than CFAR. The model with the best correlation to the post-test is BKT-LessData, achieving a correlation of 0.565.

Table I. RMSEs and Correlations between the individual models and the post-test, sorted on RMSE

Model	RMSE	Correlation
AvgCFAR	0.1636	0.5326
Avg-BKT-LessData	0.1754	0.5646
Avg-BKT-EM	0.1781	0.5518
Avg-BKT-BF	0.1834	0.5476
Avg-BKT-PPS	0.1840	0.4988
AvgPFA	0.1895	0.3243
AvgTabling	0.1901	0.2719
Avg-BKT-CGS	0.2812	-0.2367
AverageSlip	0.4279	0.0571

Table II. RMSE and correlations between the ensemble models and the post-test.

Ensemble Model	RMSE	Correlation
Stepwise	0.1652	0.5151
StepwiseWithAveraging	0.1773	0.5477
RandomForest	0.1787	0.3960
UniformAveraging	0.1806	0.5177
LinearRegression	0.2009	0.2671
LogisticRegression	0.2074	0.2645

As discussed above, 5-fold cross-validation is used when evaluating the predictive power of the ensemble models, as data from the post-test is used in creating these models. By cross-validating we can have a greater confidence that the ensemble models generalize to new groups of students.

The cross-validated RMSE and cross-validated correlations (between each of the ensemble models and the post-test) is summarized in table II. The stepwise ensemble model achieves the best RMSE among the ensemble models, achieving an RMSE of 0.1652 and a correlation of 0.515. However, stepwise ensembling achieves a poorer RMSE than the individual model with the highest RMSE, AvgCFAR. The ensemble model with the best correlation was stepwise-with-averaging, which achieved an RMSE of 0.1773 and a correlation of 0.548. However, this was a lower correlation than the top three individual models for this metric, BKT-LessData, BKT-EM, and BKT-BF.

6. CONCLUSIONS

Within this paper, we have considered the effectiveness of ensemble methods to improve prediction of post-test scores for students using a Cognitive Tutor for Genetics. Nine algorithms for predicting latent student knowledge in the post-test were used. Unlike in previous research (e.g. Baker et al., in press), we conducted ensembling at the level of the post-test rather than at the level of performance within the tutor software. It was hypothesized that doing so would lead to superior prediction of the post-test, based on past successes of combined algorithms at predicting the post-test (e.g. Baker et al., 2010), but in fact, the best individual models predict the post-test better than any ensemble

method. Using Root Mean Squared Error (RMSE) as the goodness metric, the CFAR model achieves the best post-test prediction (0.1636), followed closely by stepwise ensembling (0.1652), the best ensemble method. Using correlation as the goodness metric, the BKT-LessData model achieves the best post-test prediction, with stepwise with averaging ensembling (the best ensemble method) tied for the third best correlation with post-test.

Given the past success of ensembling methods in education (Pardos & Heffernan, *in press*) and other domains (Niculescu-Mizil et al., 2009), this lack of success is highly surprising. There are a few possible reasons for this. First of all, the data set used in this study was relatively small, with only 76 students. Ensembling methods can be expected to be more effective for larger data sets, as more complex models can only achieve optimal performance for large data sets. This is a general problem for analyses of post-test prediction. While large data sets are increasingly available for educational data (Koedinger et al., 2010), it is rare to have post-tests tailored to specific tutor lessons administered to large numbers of students. That said, tutor data sets are increasingly aligned to standardized tests such as the MCAS (Feng, Heffernan & Koedinger, 2009). It is possible to distill post-test-equivalent measures of specific skills from these standardized tests, potentially making it possible to study ensembling of knowledge transferred outside the tutor software at a larger scale, where the benefits of ensembling may be more salient. That said, it is important to be able to improve student models with relatively limited amounts of data (by the time large amounts of data have been collected, many students will have used a version of the tutoring software which is not optimized). One potential option is to optimize a generally successful model (e.g. any of the more successful models in this study) as a first pass on model improvement, and then try ensemble selection on a larger data set, when that data set becomes available.

Another reason why ensemble selection may have been less effective was high inter-correlation between the models' predictions. The predictions made by the nine models were highly correlated (except for the average slip model and BKT-CGS model), with an average inter-correlation of 0.692 (excluding those two models), and the variants on BKT were even more correlated, with an average inter-correlation of 0.927 (excluding BKT-CGS). This high degree of correlation reduces the potential gain from using models in tandem, and may suggest that ensembling may be more effective if methods for post-test prediction which leverage different information are used.

Hence, the final conclusion of this paper is negative, but the possibility remains that alternate takes on ensemble selection may be found to be a valuable part of the repertoire of methods for modeling student knowledge in intelligent tutoring systems.

ACKNOWLEDGEMENTS

This research was supported by the National Science Foundation via grant "Empirical Research: Emerging Research: Robust and Efficient Learning: Modeling and Remediating Students' Domain Knowledge", award number DRL0910188, and by a "Graduates in K-12 Education" (GK-12) Fellowship, award number DGE0742503. We would like to thank Albert Corbett for providing the data set used in this paper, and for comments and suggestions.

REFERENCES

- BAKER, R.S.J.d., CORBETT, A.T., ALEVEN, V. (2008) Improving Contextual Models of Guessing and Slipping with a Truncated Training Set. In Proceedings of 1st International Conference on Educational Data Mining, 67-76.

- BAKER, R.S.J.d., CORBETT, A.T., GOWDA, S.M., WAGNER, A.Z., MACLAREN, B.M., KAUFFMAN, L.R., MITCHELL, A.P., GIGUERE, S. (2010) Contextual Slip and Prediction of Student Performance After Use of an Intelligent Tutor. Proceedings of the 18th Annual Conference on User Modeling, Adaptation, and Personalization, 52-63.
- BAKER, R.S.J.d., PARDOS, Z.A., GOWDA, S.M., NOORAEI, B.B., HEFFERNAN, N.T. (in press) Ensembling Predictions of Student Knowledge within Intelligent Tutoring Systems. To appear in Proceedings of the 19th International Conference on User Modeling, Adaptation, and Personalization.
- BARNES, T., D. BITZER, VOUK, M. (2005) Experimental analysis of the q-matrix method in knowledge discovery. Proceedings of the 15th International Symposium on Methodologies for Intelligent Systems.
- BREIMAN, L. (2001) Random Forests. *Machine Learning*, 45, 1, 5-32.
- BROWN, G., WYATT, J. L., TINO, P. (2005). Managing diversity in regression ensembles. *Journal of Machine Learning Research*, 6, 1621-1650.
- CHANG, K.-m., BECK, J., MOSTOW, J., CORBETT, A. (2006) A Bayes Net Toolkit for Student Modeling in Intelligent Tutoring Systems. Proceedings of the 8th International Conference on Intelligent Tutoring Systems, 104-113.
- CORBETT, A.T., ANDERSON, J.R. (1995) Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction*, 4, 253-278.
- CORBETT, A., KAUFFMAN, L., MCLAREN, B., WAGNER, A., JONES, E. (2010) A Cognitive Tutor for Genetics Problem Solving: Learning Gains and Student Modeling. *Journal of Educational Computing Research*, 42, 219-239.
- DIETTERICH, T. G. (2000). Ensemble Methods in Machine Learning. Proceedings of First International Workshop on Multiple Classifier Systems, 1-15.
- FENG, M., HEFFERNAN, N.T., KOEDINGER, K.R. (2009). Addressing the assessment challenge in an Intelligent Tutoring System that tutors as it assesses. *User Modeling and User-Adapted Interaction: The Journal of Personalization Research*, 19, 243-266.
- GONG, Y., BECK, J.E., HEFFERNAN, N.T. (2010) Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting Procedures. In Proceedings of the 10th International Conference on Intelligent Tutoring Systems, 35-44.
- KOEDINGER, K.R., BAKER, R.S.J.d., CUNNINGHAM, K., SKOGSHOLM, A., LEBER, B., STAMPER, J. (2010) 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, 43-56.
- KOEDINGER, K. R., CORBETT, A. T. (2006) Cognitive tutors: Technology bringing learning science to the classroom. In K. Sawyer (Ed.), *The Cambridge handbook of the learning sciences* (pp. 61-78). New York: Cambridge University Press.
- NICULESCU-MIZIL, A., PERLICH, C., SWIRSZCZ, G., SIND-HWANI, V., LIU, Y., MELVILLE, P., et al. (2009) Winning the KDD Cup Orange Challenge with Ensemble Selection. *Journal of Machine Learning Research W & CP*, 7, 23-34.
- NOORAEI B., PARDOS, Z.A., HEFFERNAN, N.T., BAKER, R.S.J.d. (in press) Less Is More: Improving the Speed and Prediction Power of Knowledge Tracing by Using Less Data. To appear in Proceedings of the 4th International Conference on Educational Data Mining.
- PARDOS, Z. A., HEFFERNAN, N. T. (2010) Navigating the parameter space of Bayesian Knowledge Tracing models: Visualizations of the convergence of the Expectation Maximization algorithm. *Proceedings of the 3rd International Conference on Educational Data Mining*, 161-170.
- PARDOS, Z.A., HEFFERNAN, N. T. (in press) Using HMMs and bagged decision trees to leverage rich features of user and skill from an intelligent tutoring system dataset. To appear in *Journal of Machine Learning Research W & CP*.
- PAVLIK, P.I., CEN, H., KOEDINGER, K.R. (2009) Learning Factors Transfer Analysis: Using Learning Curve Analysis to Automatically Generate Domain Models. *Proceedings of the 2nd International*
- YU, H-F., LO, H-Y., HSIEH, H-P., LOU, J-K., MCKENZIE, T.G., et al. (2010) Feature Engineering and Classifier Ensemble for KDD Cup 2010. *Proceedings of the KDD Cup 2010 Workshop*, 1-16.

Improving Models of Slipping, Guessing, And Moment-By-Moment Learning with Estimates of Skill Difficulty

SUJITH M. GOWDA

Worcester Polytechnic Institute

JONATHAN P. ROWE

North Carolina State University

RYAN S.J.d. BAKER

Worcester Polytechnic Institute

MIN CHI

Stanford University

AND

KENNETH R. KOEDINGER

Carnegie Mellon University

Over the past several years, several extensions to Bayesian knowledge tracing have been proposed in order to improve predictions of students' in-tutor and post-test performance. One such extension is Contextual Guess and Slip, which incorporates machine-learned models of students' guess and slip behaviors in order to enhance the overall model's predictive performance [Baker et al. 2008a]. Similar machine learning approaches have been introduced in order to detect specific problem-solving steps during which students most likely learned particular skills [Baker, Goldstein, and Heffernan in press]. However, one important class of features that have not been considered in machine learning models used in these two techniques is metrics of item and skill difficulty, a key type of feature in other assessment frameworks [e.g Hambleton, Swaminathan, & Rogers, 1991; Pavlik, Cen, & Koedinger 2009]. In this paper, a set of engineered features that quantify skill difficulty and related skill-level constructs are investigated in terms of their ability to improve models of guessing, slipping, and detecting moment-by-moment learning. Supervised machine learning models that have been trained using the new skill-difficulty features are compared to models from the original contextual guess and slip and moment-by-moment learning detector work. This includes performance comparisons for predicting students' in-tutor responses, as well as post-test responses, for a pair of Cognitive Tutor data sets.

Categories and Subject Descriptors: I 2.7 [Artificial Intelligence]

General Terms: Educational Data Mining, Cognitive Tutor, Feature Engineering, Post-test prediction, Moment-by-Moment Learning, Contextual Guess, Contextual Slip, Bayesian Knowledge-Tracing

Additional Key Words and Phrases:

Authors' addresses: Sujith M. Gowda, Ryan S.J.d. Baker, Department of Social Science and Policy Studies, Worcester Polytechnic Institute, 100 Institute Road, Worcester, MA USA 01609. E-mail: sujithmg@wpi.edu, rsbaker@wpi.edu; Jonathan P. Rowe, Department of Computer Science, North Carolina State University, 890 Oval Drive, Raleigh, NC 27695. E-mail: jprowe@ncsu.edu; Min Chi, Developmental and Psychological Sciences, Stanford University, 485 Lasuen Mall, Stanford, CA 94305, E-mail: minchi@stanford.edu; Kenneth R. Koedinger, Human-Computer Interaction Institute, Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh PA USA 15213; E-mail : koedinger@cmu.edu

1. INTRODUCTION

Bayesian knowledge tracing (BKT) is a well-established approach for modeling student knowledge during interactions with an intelligent tutoring system. First deployed by Corbett and Anderson [1995] in the ACT Programming Tutor, BKT has since been applied successfully across a range of academic subjects and student populations, including elementary reading [Beck and Chang 2007], middle school mathematics [Koedinger 2002], and college-level genetics [Corbett et al. 2010]. In the approach, a two-node dynamic Bayesian network (DBN) is associated with each skill, and it is used to monitor student knowledge and problem-solving actions associated with that skill. The relationship between skills and steps during problem-solving is typically defined by a running cognitive model, although other forms of skill-item mappings can also be used. The probability values in BKT are defined using four parameters: $p(T)$ is the Transition parameter, which is the probability of learning a skill immediately after an opportunity to apply it; $p(L_0)$ is the Initial Learning parameter, which is the probability of knowing a skill prior to the first opportunity to apply it; $p(G)$ is the guess parameter, which is the probability of providing a correct response despite not knowing an associated skill; and $p(S)$ is the Slip parameter, which is the probability of providing an incorrect response despite knowing an associated skill. This concept of guess and slip is also seen in DINA IRT models [De La Torre, 2004]. The parameter values are typically machine-learned from data collected during prior students' tutorial interactions.

Several attempts have been made to extend BKT in order to relax some of its assumptions about learning and performance, and improve predictive power. Corbett and Anderson [1995] proposed a technique that uses individualized weights to adjust the model's four parameters for each student. A key limitation of this method is that it cannot be used at run-time, as optimization is conducted after all data is obtained. Recent work by Pardos and Heffernan introduced the Prior Per Student individualization technique, which slightly modifies the standard two-node DBN structure [Pardos and Heffernan 2010]. In the technique, an individualization node is added with states for each student, and it is connected to nodes in the model to be individualized. Unlike Corbett and Anderson's technique, this approach uses data from the first action to individualize student priors, and can thus be used in a running tutor. An empirical evaluation found the Prior Per Student extension to be significantly more accurate in predicting student responses during tutoring sessions with the ASSISTment system. While both of these approaches improved upon the predictive accuracy of standard BKT, neither approach addressed the assumption that knowledge-tracing parameters are fixed over the course of a tutoring session.

An alternate extension to BKT is Contextual Guess and Slip, which focuses on relaxing the assumption that guess and slip probabilities are fixed [Baker, Corbett and Aleven 2008]. Contextual models of guessing and slipping leverage information about the current tutorial state in order to dynamically adjust the guess and slip parameters associated with each opportunity to apply a skill. Baker and colleagues [2008a] have demonstrated that contextual models of guessing and slipping improve the fit of knowledge tracing models to existing tutoring data, although later results contradict that finding [Baker et al. 2010]. However, Baker and colleagues [2010] have found that estimates of contextual slip, used in conjunction with standard Bayesian Knowledge-Tracing estimates of student knowledge, can be used to improve predictions of students' post-test performance outside of a tutor. A similar approach involving Bayesian analysis and supervised machine learning has been devised to detect how much learning occurs in each problem step (termed moment-by-moment learning) [Baker, Goldstein and Heffernan in press]. Accurately detecting the amount of learning, moment-by-moment,

may enable intelligent tutoring systems to tailor practice opportunities more precisely, as well as shed light on the differences in learning rates between skills [e.g. Baker, Goldstein and Heffernan in press].

An integral step in constructing contextual models of guessing and slipping, as well as moment-by-moment learning detectors, is selecting appropriate predictor features for supervised machine learning. Baker and colleagues originally used a set of twenty three features to train linear regression models to predict guess and slip probabilities [Baker et al. 2008a], drawn in turn from earlier work detecting gaming the system [Baker et al. 2008b]. One important class of features that was not included in the original work is metrics of skill difficulty. This class of features is an important component in alternative student modeling techniques, such as Item Response Theory (IRT). In Item Response Theory, each test question is associated with an item characteristic curve, which estimates the probability that a student with a given ability will answer the question correctly [Baker 2001; Hambleton, Swaminathan and Rogers 1991]. A key parameter of IRT is question difficulty, which quantifies the level of ability necessary to answer the question correctly. Leveraging features that quantify skill and item difficulty is a promising direction for enhancing the performance of contextual models of guessing and slipping, as well as moment-by-moment learning detectors. For example, in the 2010 KDD Cup, an annual Data Mining and Knowledge Discovery competition where data mining teams across the world compete to solve a practical data mining problem, the goal was to predict student performance within a Cognitive Tutor. Several of the top performers in the 2010 KDD Cup leveraged skill difficulty features in their solutions [Pardos and Heffernan 2010b; Shen et al 2010; Yu et al. 2010]. Skill difficulty is also a component of Performance Factors Analysis, a competing approach to assessing student proficiency in intelligent tutors [Pavlik et al. 2009].

Within this paper, we investigate the addition of several skill-difficulty features for training models of guessing and slipping, as well as detecting moment-by-moment learning. We describe seven features that quantify different aspects of skill difficulty, and report findings about their impacts on the goodness of the resultant machine-learned models. Models that incorporate the new skill-difficulty features are compared to models that use only the original set of predictor features. We investigate whether the new models lead to improved predictions of students' guesses and slips, moment-by-moment learning, and post-test performance.

This paper is organized as follows. Additional background on contextual models of guessing and slipping is provided and details about detecting moment-by-moment learning are discussed. Afterwards, the data and method used in the current investigation are described. Results are presented about improvements in contextual models of guessing and slipping, moment-by-moment learning detectors, and predictions of post-test performance. The paper concludes with a discussion of implications for practice and future directions.

1.1 Contextual Models of Slipping and Guessing

The Contextual Guess and Slip (CGS) model of student knowledge is an extension of Corbett and Anderson's [1995] BKT model, proposed by Baker, Corbett and Aleven [2008]. Unlike Corbett and Anderson's approach, the CGS model estimates whether each individual student response is a guess, denoted $P(G)$, or a slip, denoted $P(S)$, based on contextual information. This is in contrast to approaches that utilize fixed guess and slip probability estimates throughout tutoring [e.g. Corbett & Anderson 1995].

In the BKT-CGS model, each skill has a separate parameter for Initial Knowledge and Transition, as in standard BKT. However, unlike in standard BKT, guess and slip probabilities are not estimated for each skill. Instead, they are computed each time a

student attempts to answer a new problem step in the tutor, and they are based on machine-learned models of guessing and slipping behaviors within the current tutorial context (for example, longer responses and help requests are less likely to be slips). The Contextual Guess and Slip procedure involves five stages, and it proceeds as follows. First, a four-parameter model is obtained using the brute force variant of Bayesian knowledge tracing [Baker et al 2010]. Second, the four-parameter model is used to assign slip and guess probability labels to each action in the data set. Third, supervised machine learning is used to obtain models that predict the slip and guess labels. Fourth, the machine-learned models of guessing and slipping are incorporated into the BKT models in lieu of skill-by-skill labels for guessing and slipping. Last, parameters for Initial Knowledge, denoted $P(L_0)$, and Transition, denoted $P(T)$, are fit. Additional details about this approach are available in [Baker et al. 2008a]. Baker et al. [2010] have found that two aggregations of contextual slip are significant predictors of post-test performance. While average contextual slip is a good predictor of post-test performance, the “certainty of slip” – the average contextual slip among actions thought to be slips (e.g. contextual slip > 0.5) – is an even stronger predictor of post-test performance.

1.2 Detecting Moment-by-Moment learning

Baker, Goldstein and Heffernan [in press] presents a model that predicts the probability that a student has learned a specific knowledge component at a specific problem step, termed $P(J)$. The underlying architecture of this model is similar to the Contextual Guess and Slip model. First, training labels to detect moment-by-moment learning are generated for each problem step in a tutor data set. The labels are generated by applying Bayes’ Rule to a combination of knowledge estimates from a traditional BKT model, as well as information about the correctness of two future problem-solving actions. Next, a set of predictor features is generated using past tutor data to form a training data set. Finally, these predictor features are used during supervised machine learning to train a model that predicts the moment-by-moment learning labels. The machine-learned model computes a learning probability for each problem-solving step using no data from the future. A recent investigation observed that a distillation of the variance in this model, termed “spikiness”, is a good predictor of students’ eventual learning within the tutor, as assessed by BKT. Additionally, spikiness aids in understanding the differences between gradual learning and learning via “eureka” moments, where a KC is understood suddenly [Baker, Goldstein and Heffernan in press].

2. METHOD

The current work investigates whether contextual models of slipping, guessing, and moment-by-moment learning can be improved by incorporating skill-difficulty features in the machine-learned models. Data from two Intelligent Tutoring Systems, The Middle School Mathematics Cognitive Tutor [Koedinger 2002] and the Genetics Cognitive Tutor [Corbett et al. 2010], are considered. The Middle School Cognitive tutor, a precursor to the current curriculum Bridge to Algebra, covers a wide span of Mathematics topics in middle school mathematics. Topics covered by the Middle School Cognitive Tutor include fraction concepts, areas and perimeters with decimals, and simple histograms. The Middle School data set consists of an entire year’s use of an intelligent tutor in suburban schools in the Northeastern USA. Within the data set, actions that were not labeled with skills were excluded, because skill information is necessary for the application of Bayesian knowledge tracing techniques.

The Genetics data set comes from the Genetics Cognitive Tutor [Corbett et al. 2010], which consists of 19 modules that support problem solving across a wide range of topics in genetics (Mendelian transmission, pedigree analysis, gene mapping, gene regulation

and population genetics). Various subsets of the 19 modules have been piloted at 15 universities in North America. In the study that generated the current data set, 70 undergraduate students enrolled in a genetics course at Carnegie Mellon University used the three-factor cross module as a homework assignment. Students completed a paper-and-pencil problem-solving pre-test and post-test consisting of two problems. There were two test forms, and students were randomly selected to receive one version at pre-test and the other version at post-test in order to counterbalance test difficulty. Each problem on the tests consisted of 11 steps involving 7 of the 8 skills in the Three-Factor Cross tutor lesson, with two skills applied twice in each problem and one skill applied three times. Note that this data set is the same one used in [Baker et al. 2010]. The size of each data set is shown in Table 1.

Table 1. The size of each data set (after exclusion of actions not labeled with skills)

Data set	Actions	Problem Steps	Skills	Students
Middle school	581,785	171,987	253	232
Genetics	19,150	9,259	8	71

In the original work on contextual models of guessing and slipping [Baker, Corbett, and Aleven, 2008] and detecting moment-by-moment learning [Baker, Goldstein, and Heffernan 2010], four primary categories of predictor features were used during the supervised machine learning stages of the procedures. The original features date back to the development “gaming the system” detectors for Cognitive Tutors [Baker et al. 2008b]. The four categories include:

- **Action Correctness.** This category includes features that characterize whether a problem-solving action is correct, incorrect, constitutes a known misconception, or a help request.
- **Step Interface Type.** This category includes features that are based on the type of interface widget involved in the problem-solving action. For example, a particular problem step may involve typing a string or specifying a number.
- **Response Times.** This category includes features that are derived from the amount of time taken to complete problem-solving steps. Example features include the amount of time (seconds) taken on a particular step, time-taken expressed as standard deviations from the mean, and total time spent during the last three problem-solving actions.
- **Problem-Solving History.** This category includes features that characterize the student’s problem-solving history in the tutor. Examples include: the number of past five actions that were involved in the current step, and the number of times that the student asked for help in the past eight problem-solving actions.

Within this paper, we add seven new predictor features that quantify key aspects of the skill associated with each problem-solving step. As discussed in the introduction, the new features are inspired by Item Response Theory concepts of skill difficulty, which were prominent in many of the successful entries in the KDD cup. The features are calculated by determining average metric values using other students’ problem-solving histories; this procedure is cross-validated at the student level during evaluation of this approach (see below for details). The newly engineered features include the following:

- Features based on item difficulty:
In Cognitive Tutors, a student’s action is represented using four predefined categories:
 - Right – when the user inputs correct answer

- Help - when the user requests help
- Bug - when the tutor gives a bug message, indicating a misconception
- Wrong - when the user inputs incorrect answer

We assessed average performance among all students for each of these categories: "Average-right", "Average-bug" and "Average-help". These features were calculated by computing the proportion of each of category across all the students for a given skill.

- Features based on user input:

In the data distillation, a user's input type is categorized as either a number or a string. Based on user input information we derived two features "Average-number" and "Average-string" by computing the proportion of "inputs which are a number" and "inputs which are a string" respectively across all the students for a given skill. Students will have a greater proportion of actions of one type, relative to other students completing the same curriculum, when they have more difficulty with that type of action. Consequently, this feature indirectly measures the relative difficulty of skill groups.

- Feature based on time:

The Cognitive tutor records the time taken by the user to answer a step. Using this time we derived the feature "Average-time" by calculating the average time taken to answer problem steps across all students for a given skill. Students who take longer can, in the aggregate, are assumed to be having greater difficulty.

- Feature based on number of opportunities to practice a skill:

This feature is a distillation of the feature "optoprac" from [Baker et al. 2008b], which computes the number of times that the same skill has been seen by the student prior to this action. Using optoprac, the feature "Average-optoprac" was calculated by aggregating across actions and students for each skill.

The new set of predictor features provides several aggregate assessments of the skills involved in the problem-solving process. These new features are considered for machine learning contextual models of guessing and slipping behaviors, as well as models for detecting moment-by-moment learning. The next section describes the supervised machine learning analysis that has been conducted to investigate the impact of including these features, and the results in terms of model goodness.

3. RESULTS

In discussing the results, we will first present the linear regression analyses we conducted to develop each type of model, and then discuss improvements in the overall model from incorporating the new features. We will discuss model improvements in terms of fit to training labels, as well as external measures such as predicting post-test performance. We use linear regression for consistency with the original analyses [Baker, Corbett, and Aleven 2008; Baker, Goldstein, and Heffernan, in press], where linear regression was found to lead to acceptable cross-validated performance.

3.1 Model Development

Using both the original features and new features, we used RapidMiner [Mierswa, et. al. 2006] to develop linear regression models, using default settings (e.g. M5-prime feature selection). Leave-one-out cross-validation (LOOCV) was conducted, at the student level, to evaluate the models. By cross-validating at the student level rather than the action level, we can have greater confidence that the resultant models will generalize to new groups of students. The LOOCV procedure is as follows: Before the generation of folds, labels were generated for contextual slip and guess and for moment-by-moment learning

using the same labeling process as in [Baker et al. 2008a; Baker, Goldstein and Heffernan in press]. Special care needed to be taken to ensure that skill-level features did not violate independence during cross-validation. Values for all of the skill-level features were calculated using only data from the training set. After calculating the skill-level feature values for each fold, the features were added to both the training set and the test set (e.g. features generated on training set data were used within the test set). This corresponds to what would occur in real-world applications of this approach, where skill-level features might be generated on data from one cohort and applied to students in a later cohort.

For each training set, three linear regression models were machine-learned: one for guessing, one for slipping, and one for detecting learning moment-by-moment. By applying these models to the corresponding test set, model predictions for guess, slip, and P(J) were obtained. Correlations were next calculated between the labels and guess, slip, and P(J) model predictions for each student and then averaged across the students.

After obtaining correlation values for the new models, the same cross-validation procedure was applied to obtain training and test sets that included only the original predictor features. These models did not include any of the new skill-difficulty features, and were intended to serve as a comparison group. The statistical significance of the difference between the new feature model and the old feature model was computed by computing the correlation coefficient for each fold (student), converting the correlation coefficients to Z values, and then aggregating across folds (students) using Stouffer's Z.

The cross-validated correlation of each model to the training labels is shown in Table 2. As can be seen, all models involving the new features achieve statistically significantly better fit to the training labels in the test folds, at the $p < 0.001$ level, than the models that use only the original feature set. These significant differences are observed for all three types of models—guessing, slipping, and detecting moment-by-moment learning—and they are observed in both the Middle School and Genetics data sets.

In the Middle School data set, the slip model using the new features achieves a 32% improvement over the model with the original features, the guess model achieves a 46% improvement, and the P(J) model achieves a 17% improvement. In the Genetics data set, the slip model with the new features achieves a 22% improvement over the original model. The P(J) model with the new features achieves a 3% improvement over the model with the original features. The guess model with the new features achieves a very large improvement in correlation, from 0.029 to 0.422. Of the seven newly engineered features, five are found in all three regression models in both data sets: Average-right, average-help, average-string, average-time and average-optoprac. Of the three types of models, the strong performance for the slip models in both data sets is of particular interest, because slipping within a Cognitive Tutor has been shown to be a significant predictor of post-test performance [Baker et al. 2010], even after controlling for student knowledge.

Table 2: Correlation coefficients between labels and predictions

Data-Set	Model	OldFeatures- <i>r</i>	NewFeatures- <i>r</i>	Z	P
Middle School	Slip	0.322	0.424	23.91	< 0.001
	Guess	0.112	0.163	5.42	< 0.001
	P(J)	0.476	0.558	84.16	< 0.001
Genetics	Slip	0.456	0.582	6.73	< 0.001
	Guess	0.029	0.422	33.92	< 0.001
	P(J)	0.764	0.790	21.09	< 0.001

3.2 Post-test Prediction Using Final Knowledge and Contextual Slip Estimates

Baker, et al. [2010] showed that a linear combination of average contextual slip over 0.5 (termed “certainty of slip”) and BKT estimates of post-test performance predicts

students' post-tests statistically significantly better than BKT estimates alone. We replicate this analysis using the same Genetics data set and the new models with skill-difficulty features. We also analyze regression models combining average slip (average of Contextual slip for each student across all the actions) and BKT. The analysis is not replicated for the Middle School data, because that data set lacks post-test scores. Unlike Baker et al. [2010], we use cross-validated values of contextual slip in this analysis. By using the cross-validated contextual slip values, we can have greater confidence that slip models can be generalized for new data sets.

Several models were compared in terms of correlations with students' post-test performance after interacting with the Genetics Tutor. The four-parameter brute force variant of Bayesian Knowledge Tracing (BKT-BF) model achieves a cross-validated correlation of $r = 0.426$ to students' post-tests, which is statistically significantly better than chance, $F(1,69) = 15.27$, $p < 0.001$. For the model generated using the original features, we find that a combination of cross-validated certainty of slip (average of slip above 0.5) and BKT-BF estimates achieves a cross-validated correlation of 0.437. The certainty of slip term in this model is not statistically significant, when added to a model containing BKT-BF, $t(68) = 0.92$, $p = 0.36$, for a two-tailed test. When the same model is generated using the new feature set, the model achieves a cross-validated correlation of 0.448. Again, the certainty of slip term in the combined model is not statistically significant, $t(68) = 1.29$, $p=0.20$.

However, when average slip is taken rather than certainty of slip, a different pattern emerges. The original feature version of average slip, when combined with BKT-BF, achieves a cross-validated correlation of 0.479. In this model, the average slip term is statistically significant, $t(68)=2.06$, $p=0.04$, signifying that the model containing average slip is significantly better than the model containing BKT-BF alone. The new feature version of average slip, when combined with BKT-BF, achieves a cross-validated correlation of 0.485. In this model, the average slip term is also statistically significant, $t(68)=2.19$, $p=0.03$.

The difference between the four models combining variants on contextual slip can be compared using BiC', the Bayesian Information Criterion for Linear Regression Models (Raftery, 1995). Within these four models, the model containing the new feature version of average slip achieves the best BiC', but does not perform significantly better than any of the other three models (differences of 6 between model values for BiC' are considered equivalent to statistical significance at the $p<0.05$ level).

Hence, the new features significantly improve cross-validated fit to the test-fold training labels for each of these models, but do not appear to significantly improve post-test prediction, although there is some trend in that direction. Nonetheless, as these model fits are cross-validated, there is evidence for improvement (as conducting significance tests or computing BiC' on cross-validated data is doubly-stringent).

Table 3. Cross-validated correlations between post-test and t-test scores of 2nd parameter using regression analysis and BiC' values

Model	Correlation	t-test of 2nd Param	P-value of 2nd Param	BiC'
BKT-BF-Predictions Only	0.426			-9.763
BKT-BF-Preds + Old_Certainty_Slip	0.437	$t(68) = -0.920$	0.361	-6.333
BKT-BF-Preds + New_Certainty_Slip	0.448	$t(68) = -1.285$	0.203	-7.180
BKT-BF-Preds + Old_Avg_Slip	0.479	$t(68) = -2.059$	0.043	-9.742
BKT-BF-Preds + New_Avg_Slip	0.485	$t(68) = -2.186$	0.032	-10.269

4. CONCLUSION

In this paper, we present a new set of features related to metrics of skill difficulty, which when combined with original features [Baker et al 2008a], have been shown to statistically significantly improve the predictive capabilities of guessing, slipping and moment-by-moment learning models. The newly engineered features are inspired by Item Response Theory concepts of item difficulty, and they aggregate empirical estimates of skill performance, types of user input, problem-solving action time, and average practice opportunities for each skill. We find that guess, slip, and moment-by-moment learning models that use the new skill-difficulty features outperform the original-feature models under cross validation for two Cognitive Tutor data sets. Therefore, the findings suggest that the new features can be computed using previous years' class data, and they can then be directly incorporated into models for new students.

Of the three models that were investigated, the slip model is of particular interest because certainty of slip has been shown to be a significant predictor of post-test performance [Baker et al. 2010]. In this paper, we replicated the tests in that paper, using cross-validation. We found that a model predicting the post-test using both average contextual slip and Bayesian Knowledge Tracing had significantly better goodness than a model using Bayesian Knowledge Tracing alone, even when cross-validating the data. However, the evidence for improvement stemming from the new features appeared to be weak. While the new-feature models achieved slightly better cross-validated performance than the original-feature models, the difference in BiC' values was small. One possible explanation for the small improvements in post-test prediction could be imprecision in the guess and slip labels. The new models with skill-difficulty features are performing more effectively at predicting the existing guess and slip labels, but any inaccuracies or lack of precision in these original labels could degrade post-test prediction performance. Another possible explanation could be that the contextual slip estimates may have reached a ceiling in their capacity to predict the post-test. In this case, additional features would be necessary to account for the remaining differences between post-test responses and model predictions.

There are several promising future directions for this work. First, continued exploration of new predictor features may yield further improvements in the accuracies of guess, slip, and moment-by-moment learning models. Second, further investigation is necessary to determine whether imprecision in guess and slip labels are responsible for the relatively modest gains observed in post-test prediction performance. This requires further analysis of students' guessing and slipping behaviors in order to assess the quality of guess, slip, and P(J) labels generated by Bayesian analysis procedures. Finally, additional work should be conducted to determine how enhanced guess, slip, and moment-by-moment learning models can be most effectively incorporated back into run-time Bayesian knowledge tracing models in order to improve the effectiveness of intelligent tutoring systems.

ACKNOWLEDGEMENTS

This research was supported via grant "Empirical Research: Emerging Research: Robust and Efficient Learning: Modeling and Remediating Students' Domain Knowledge", National Science Foundation award number #DRL-0910188, and by the Pittsburgh Science of Learning Center, NSF award number #SBE-0836012. We would also like to thank April Galyardt and Zachary Pardos for helpful suggestions at an early stage of this project.

REFERENCES

- BAKER, F.B. 2001. *The Basics of Item Response Theory*. ERIC Clearinghouse on Assessment and Evaluation, College Park, MD.
- BAKER, R.S.J.D., CORBETT, A.T., ALEVEN, V. 2008a. 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*, 406-415.
- BAKER, R.S.J.D., CORBETT, A.T., GOWDA, S.M., WAGNER, A.Z., MACLAREN, B.M., KAUFFMAN, L.R., MITCHELL, A.P., GIGUERE, S. 2010. Contextual Slip and Prediction of Student Performance After Use of an Intelligent Tutor. *Proceedings of the 18th Annual Conference on User Modeling, Adaptation, and Personalization*, 52-63.
- BAKER, R.S.J.D., CORBETT, A.T., ROLL, I., KOEDINGER, K.R. 2008b. Developing a Generalizable Detector of When Students Game the System. *User Modeling and User-Adapted Interaction*, 18, 3, 287-31
- BAKER, R.S.J.D., GOLDSTEIN, A.B., HEFFERNAN, N.T. in press. Detecting Learning Moment-by-Moment. To appear in *International Journal of Artificial Intelligence in Education*.
- BECK, J. AND CHANG, K. 2007, Identifiability: A Fundamental Problem of Student Modeling. *Proceedings of the 11th International Conference on User Modeling*. 137-146.
- CORBETT, A., KAUFFMAN, L., MACLAREN, B., WAGNER, A., JONES, E. 2010. A Cognitive Tutor for Genetics Problem Solving: Learning Gains and Student Modeling. *Journal of Educational Computing Research*, 42, 219-239.
- CORBETT, A.T., ANDERSON, J.R. 1995. Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction*, 4, 253-278.
- DE LA TORRE, J. 2004, Higher-Order Latent Trait Models for Cognitive Diagnosis. *Psychometrika*, 69, 3, 333-353.
- HAMBLETON, R.K. SWAMINATHAN, H., ROGERS, H.J. 1991. *Fundamentals of Item Response Theory*. Sage Publications, Newbury Park, CA.
- KOEDINGER, K.R. 2002. 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)*.
- MIERSWA, I., WURST, M., KLINKENBERG, R., SCHOLZ, M., EULER, T. 2006. YALE: Rapid Prototyping for Complex Data Mining Tasks. In: *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (KDD 2006), 935-940.
- PARDOS, Z. AND HEFFERNAN, N. 2010a. Modeling Individualization in a Bayesian Networks Implementation of Knowledge Tracing. *Proceedings of the 18th International Conference on User Modeling, Adaptation, and Personalization*, 255-266.
- PARDOS, Z. AND HEFFERNAN, N.T. 2010b. Using HMMs and Bagged Decision Trees to Leverage Rich Features of User and Skill from an Intelligent Tutoring System Dataset. *Proceedings of the KDD Cup 2010 Workshop held as part of the 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.
- PAVLIK, P.I., CEN, H., KOEDINGER, K.R. 2009. Learning Factors Transfer Analysis: Using Learning Curve Analysis to Automatically Generate Domain Models. *Proceedings of the 2nd International Conference on Educational Data Mining*, 121-130.
- RAFTERY, A.E. 1995. Bayesian Model Selection in Social Science Research. *Sociological Methodology*, 28, 111-163.
- SHEN, Y., CHEN, Q., FANG, M., YANG, Q., AND WU, T. 2010 Predicting Student Performance : A Solution for the KDD Cup 2010 Challenge. *Proceedings of the KDD Cup 2010 Workshop held as part of the 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.
- YU, H.-FU, LO, H.-YI, HSIEH, H.-PING, et al. 2010 Feature Engineering and Classifier Ensemble for KDD Cup 2010. *Proceedings of the KDD Cup 2010 Workshop held as part of the 16th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

SHORT PAPERS

A Method for Finding Prerequisites Within a Curriculum

Annalies Vuong, Tristan Nixon, and Brendon Towle¹, Carnegie Learning

Creating an educational curriculum is a difficult task involving many variables and constraints [Wang 2005]. In any curriculum, the order of the instructional units is partly based on which units teach prerequisite knowledge for later units. Historically, psychologists and cognitive scientists have studied the dependency structure of information in various domains [Bergan and Jeska 1980; Griffiths and Grant 1985; Chi and Koeske 1983]; however, many of these studies have been hampered by statistical issues such as the difficulty of removing instructional effects when using small samples [Horne 1983]. We hypothesize that large-scale assessment data can be analyzed to determine the dependency relationships between units in a curriculum. This structure could then be used in generating and evaluating alternate unit sequences to test whether omitting or re-ordering units would undermine necessary foundational knowledge building. Our method incorporates all possible pair-wise dependency relationships in a curriculum and, for each such candidate dependency, compares performance of students who used the potential prerequisite unit to performance of students who did not. We implemented the method on a random sample of schools from across the U.S. that use Carnegie Learning's Cognitive Tutor software and its associated curricula; a sample that far exceeds those used in previous studies both in size and in scope. The resulting structure is compared to a pre-existing list of prerequisites created by Carnegie Learning based on student skill models. We discuss extensions of this method, issues in interpreting the results, and possible applications. We hope that this work serves as a step toward developing a data-driven model of curriculum design.

Additional Key Words and Phrases: data mining, curriculum, intelligent tutor

1. INTRODUCTION

Curriculum is a fundamental part of education at every scale, from a one-day class to a four-year degree. Designing a curriculum involves balancing many competing constraints, not least of which is prerequisite knowledge. The method we present applies to any scale of curricula, though our data comes from mathematics curricula spanning one school year.

Prerequisite knowledge is here defined as the skills and information necessary to succeed in a given instructional unit within a curriculum. This knowledge can be acquired inside or outside the curriculum, giving rise to three important questions. What prerequisite knowledge is required to successfully learn each topic in the curriculum? Which units in the curriculum teach this knowledge? Finally, have students already acquired this knowledge outside of the curriculum? The third question will have an answer unique to each instructional situation. However, the first and second questions depend only on the content of the curriculum, and we believe that they can be answered empirically.

In fact, we believe they can be combined into one question. Given an instructional unit – call it Unit B – which prior units significantly influence students' success in this unit? It seems reasonable to conclude that these prior units cover some prerequisite knowledge. This paper lays out a method, given sufficient user data, for finding the prerequisite units for each instructional unit in a curriculum.

Creating a curriculum always involves defining prerequisites implicitly or explicitly. However, these definitions are usually based on expert opinion or on a theoretical model [Bergan and Jeska 1980; Griffiths and Grant 1985; Chi and Koeske 1983] and are rarely tested empirically. Even algorithms for designing optimal curricula may take prerequisite relations as a given [Wang 2005]. This lack of empirical testing is understandable, as it can be difficult to assess causal relationships or remove instruc-

¹Authors' addresses: A. Vuong, T. Nixon, and B. Towle, Carnegie Learning, Inc., 437 Grant St., Pittsburgh, PA 15219. Correspondence can be sent to avuong@carnegielearning.com.

tional effects, especially in small observational studies [Horne 1983]. Using a small sample also runs the risk of only answering the third question: do these students already have sufficient prior knowledge. A further difficulty in determining dependency structure is that a curriculum of multiple units will have a significant number of possible prerequisites to test.

Many of these problems relate to having small sample sizes. Our data are collected from students using Carnegie Learning's Cognitive Tutor, by far the most widely used intelligent tutoring system in the United States: it is currently used by over five hundred thousand students in more than twenty-five hundred schools. For high school mathematics, Carnegie Learning offers four Cognitive Tutor curricula. Each of the curricula consists of a sequence of units; each unit consists of a sequence of sections. Units cover distinct mathematical topics; sections cover distinct sets of problems on that topic, with a distinct student skill model for each section. Teachers and school administrators can create customized variations of the standard curricula by omitting units, reordering units, or adding units from another curriculum, though they cannot customize the skill models or problem sets within each unit. In school year 2008-2009, 85% of teachers used a customized curriculum. The great variety of curricula variations thus created provided us with a natural opportunity to compare student performance as they progressed through subtly different unit sequences.

That all students used the same software was very helpful in avoiding issues raised in previous studies: it reduced instructional effects in the data, reduced the chance of content-based false positives via the coverage of a large number of topics, and provided sufficiently uniform data to test almost all possible prerequisites.

2. METHOD

2.1. Data

Our data is taken from a random one-fifth sample of all schools using Cognitive Tutor in the 2008-2009 school year from whom were collected detailed logs of students' activity in the software. This sample comprises 20,577 students from 888 schools across the United States.

The standard Carnegie Learning high school curricula – Bridge to Algebra, Algebra I, Geometry, and Algebra II – contain 175 total units, including cross-listed units. Every unit prior to a given unit within each curriculum was considered a possible prerequisite; “true” prerequisites of a unit are defined to be those which have a significant effect on the success of students in completing the target unit. The list of all possible prerequisite relationships can be represented as a list of pairs of the form (Unit A, Unit B), where A is prior to B in one of the four curricula. For each pair of units, Sample A comprised all students in the data set whose curricula included both Unit A and Unit B and who attempted at least one problem in both units. Sample B contained all students whose curricula included Unit B but omitted Unit A and who attempted at least one problem in Unit B. We tested all such pairs for which there was at least one student in each sample, resulting in 3,832 tested pairs. Lacking enough information to compare a pair of units was rare: only 30 pairs were not tested. The average Sample A size was 325.5 students, with average Sample B size at 506.8 students.

2.2. Testing

For any given Unit B, student performance in the first section of the unit is more likely to be affected by prior knowledge from other units than performance in the later sections, since later sections would likely rely strongly on knowledge from earlier in the unit. To avoid this confound, we decided to evaluate success in Unit B by looking at student performance on the first section of the unit. Our hypothesis was that if

Unit A truly provides prerequisite knowledge for Unit B, we should see an increased graduation rate on the first section of Unit B for students who have had some exposure to the material in Unit A. We therefore calculated the overall graduation rate from the first section of Unit B in each of the two samples.

In the Cognitive Tutor, a student graduates from a section only if they master all of the section's skills over the course of a reasonable number of problems. The system will automatically promote them to the next section if they hit the problem limit without mastering all skills. Additionally, a student may simply stop working on a section and never return, for reasons such as leaving the class. Finally, a teacher may move a student forward out of a section if they are not keeping pace with the rest of class. Given the ways in which a student may leave a section without graduating, it is reasonable to assume that in general the performance of students who graduated from the first section of Unit B was better than the performance of those who did not and thus reasonable to use graduation rate as a performance metric.

To compare graduation rates for each pair of units in our list of possible prerequisites, we used a binomial test with $\alpha = 0.01$. The binomial test for each pair looked for a significant difference in the average graduation rate for Sample A as compared to the rate for Sample B. If a significant difference was found, Unit A was deemed a true prerequisite for Unit B.

3. RESULTS

The average number of true prerequisites for each unit was approximately 9.6 out of an average of 21.2 possible prerequisites. Overall, a little over 43% of all possible prerequisites were found to be true prerequisites.

As part of our analysis, we compared the data-driven prerequisites to the list of prerequisite relationships which is already included in the Cognitive Tutor, a list generated primarily from shared skills in the cognitive models for different units. Figure 1 shows all possible (Unit A, Unit B) pairs across the four curricula as elements in a 175 by 175 matrix, with Unit A as the row and Unit B as the column. The order of the units within each curriculum and the order of curricula (Bridge to Algebra, Algebra I, Geometry, Algebra II) was preserved in each axis of the table, hence the triangular regions indicate the standard curricula. In Figure 1, red indicates a non-significant relationship between Units A and B, with green indicating a significant relationship. Colored blocks outside the triangular regions display the results of cross-listed units, and white blocks indicate inter-curricular pairs not on our list of possible prerequisites.² Yellow blocks mark the small number of cases where there was insufficient data.

Figure 2 shows the difference between the set of empirically-derived prerequisite relationships and the set of prerequisite relationships given by the Cognitive Tutor for the same list of possible prerequisites. Dark green indicates a true prerequisite relationship found in both sets; light green a relationship deemed not prerequisite by both. Orange indicates a prerequisite relationship listed by the Cognitive Tutor that was not found in the data. Yellow shows a relationship found in the data but not in the Cognitive Tutor set. Overall, there was 56% agreement between the two methods (percentage of green blocks) and 14% agreement on true prerequisites (percentage of dark green blocks). The next section gives possible reasons for these differences.

4. DISCUSSION

Our goal was to devise a method which would empirically determine the prerequisite relationships in a given curriculum. It is important to note that this method is dis-

²Units were compared only within each curriculum. A study allowing for cross-curricular prerequisites could yield further pertinent results.

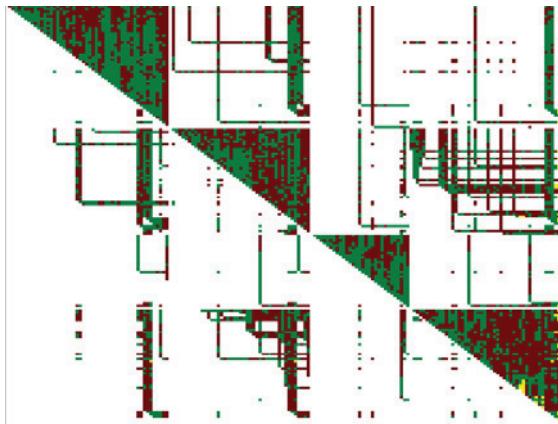


Fig. 1. Heatmap of data-based prerequisite relationships. Red indicates a non-significant relationship between Units A and B; green, a significant relationship; yellow, insufficient data to compare; and white, a pair not on our list of possible prerequisites.

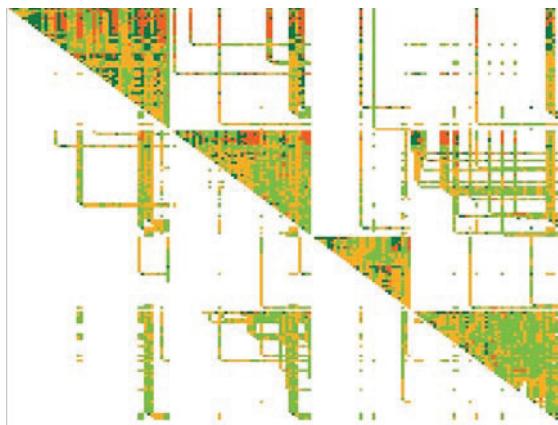


Fig. 2. Heatmap comparing data-based and cognitive model-based prerequisite relationships. Dark green indicates a true prerequisite relationship found in both sets; light green, a relationship deemed not prerequisite by both; orange, a prerequisite relationship listed by the Cognitive Tutor that was not found in the data; and yellow, a relationship found in the data but not in the Cognitive Tutor.

tinctly different from methods to derive student skill models from data (e.g. [Cen et al. 2006; Barnes et al. 2005]). We feel that our method is complimentary to such work. Our analysis takes place at a larger grain-size, comparing the relationships between units of instruction that consist of distinct skill models. The similarity or difference between these skill models may play an important role in the prerequisite relationship between the units.

It is interesting to note how the empirically derived prerequisite structure deviates from that determined by domain experts. Where the empirical evidence shows a prerequisite not identified by a domain expert, we can imagine mathematical skills practiced in both units but not part of the focus for the units, and thus easily overlooked in the expert analysis. Expert judgment is more suspect when it has identified a prerequisite that is not borne out by the data (marked in orange). However, as we see from Figure 2, this type of disagreement between expert and empirical data was confined mainly to the earlier units in each standard curriculum. It may be that a large

number of students were already sufficiently proficient on this preliminary material, to the point where any practice on prerequisite material would make no appreciable improvement to their performance. In this case, these could be true prerequisites but for our population it would make more sense to treat them as false prerequisites.

These results do raise many new questions, especially where this analysis disagrees with expert assessment, and further investigation of such units is warranted. There are many possible hypotheses for any given pre-requisite relationship. It could be that the skills practiced in unit A are truly foundational for those in B. Conversely, it could also be the case that the units share some subset of skills in common, and that it is simply the previous practice with these skills which provides the improved performance on unit B. In such a case, the curriculum structure is effectively loading the learning of those common skills onto whichever unit occurs first. It is also possible that another metric of student performance might reveal a difference not revealed by graduation rates. Although our method does not answer such questions, it does provide a useful framework for focusing attention to those unit pairs which deserve more investigation.

Since there is not already an agreed-upon way to objectively determine prerequisites, it is difficult to assess the validity of our method. A possible assessment method would be to see if different measurements of student performance yield the same prerequisite relationships. One such measure of performance is the number of problems done by students before graduating. Preliminary data exploration suggests that the result with this metric will be similar.

The dependency structure of a curriculum can be a powerful piece of information. For instance, [Ohland et al. 2004] found that removing a gateway course for their engineering major improved graduation rates for the major as a whole, suggesting that the course was not a true prerequisite. Every curriculum is subject to time constraints, and knowing which units can be safely omitted if necessary is important. Given this information, we could help teachers to customize their curricula without removing necessary prerequisite units.

Overall we feel that data-driven course design is a fruitful topic for research, which could yield relevant and useful information in many different areas of education.

ACKNOWLEDGMENTS

The authors would like to thank Dr. Steve Ritter of Carnegie Learning for his guidance and advice.

REFERENCES

- BARNES, T., BITZER, D., AND VOUK, M. 2005. Experimental analysis of the Q-matrix method in knowledge discovery. *Foundations of Intelligent Systems*, 603–611.
- BERGAN, J. AND JESKA, P. 1980. An examination of prerequisite relations, positive transfer among learning tasks, and variations in instruction for a seriation hierarchy. *Contemporary Educational Psychology* 5, 3, 203–215.
- CEN, H., KOEDINGER, K., AND JUNKER, B. 2006. Learning Factors Analysis—A general method for cognitive model evaluation and improvement. *Intelligent Tutoring Systems*, 164–175.
- CHI, M. AND KOESKE, R. 1983. Network representation of a child's dinosaur knowledge. *Developmental Psychology* 19, 1, 29–39.
- GRIFFITHS, A. AND GRANT, B. 1985. High school students' understanding of food webs: Identification of a learning hierarchy and related misconceptions. *Journal of Research in Science Teaching* 22, 5, 421–436.
- HORNE, S. 1983. Learning Hierarchies: a critique. *Educational Psychology* 3, 1, 63–77.
- OHLAND, M., YUHASZ, A., AND SILL, B. 2004. Identifying and removing a calculus prerequisite as a bottleneck in Clemson's General Engineering curriculum. *JOURNAL OF ENGINEERING EDUCATION-WASHINGTON*- 93, 253–258.
- WANG, Y. 2005. A GA-based methodology to determine an optimal curriculum for schools. *Expert Systems with Applications* 28, 1, 163–174.

Estimating Prerequisite Structure From Noisy Data

EMMA BRUNSKILL, University of California, Berkeley

There has been a long-standing interest in how to estimate the prerequisite structure among a set of skills. This prerequisite structure is important for intelligent tutoring systems and the educational data mining community, because it has important implications for student modeling, and for automatically constructing teaching policies. Here we present preliminary work towards inferring skill prerequisite structure given a set of noisy observations of student knowledge. We compare models using likelihood calculations and provide experimental results on a set of Algebra skills.

1. INTRODUCTION

At least since the work of Robert Gagné and his colleagues [1974], there has been interest in estimating the prerequisite structure among items to be learned. In this paper we will use prerequisite structure to refer to relationships among skills that provide strict constraints on the order in which these skills can be acquired.

There are two primary reasons why inferring the prerequisite structure amongst skills is important for the intelligent tutoring systems and educational data mining community. First, such structure is relevant for student modeling. In this paper we follow the popular Bayesian approach (see e.g. Corbett and Anderson [1995] and Conati et al. [2002]) to modeling student learning. Though there are a number of differences between the specific models and approaches used, broadly Bayesian models of learning involve considering the student's knowledge as a hidden state that changes stochastically as pedagogical activities are provided. Information about this hidden state may also be gleaned from the student responses to the provided activities, such as correct or incorrect quiz answers. If there is prerequisite structure among the set of skills a student is trying to learn, that provides direct constraints on the dynamics of learning, since a student will not be able to learn a new skill if its prerequisite is not yet mastered. In addition, if the student correctly answers a question involving a particular skill j , that not only provides positive evidence that the student has mastered skill j , but also positive evidence that the student has mastered all of skill j 's prerequisite skills.

Second, the skill prerequisite structure affects what constitutes a good sequence of teaching activities. Providing a student with exercises and activities on solving a linear equation will be superfluous if the student has not yet understood fractions, or other key prerequisite skills needed in order to master solving linear equations. In addition to affecting the quality of different conditional teaching strategies, recent work by Brunskill and Russell [2010] has demonstrated that prerequisite structure can be leveraged to significantly reduce the computational burden of calculating good teaching strategies in simulated teaching domains.

However, despite the importance of prerequisite structure, there are a number of challenges to estimating this structure within the context of a particular curriculum. In particular, in this paper we focus on the challenge of estimating this structure given access only to noisy observations of students' hidden knowledge. This is the most natural setting for a number of common scenarios, such as inferring the prerequisite structure among a set of skills on an exam. Equally importantly, we suggest that such data is likely to be more easily obtainable, and therefore a candidate for automatic mining and structure extraction, compared to more intensive data collection methods such as talk aloud methods, which may be able to get a better estimate of a student's true knowledge state.

As a preliminary step towards determining prerequisite structure from noisy observations of student data, in this paper we estimate the probability of the observations under alternate possible prerequisite structures. In the rest of this short paper we will briefly discuss related work, our method, and some preliminary results, before finishing with a discussion of future work.

2. RELATED WORK

Interest in inferring prerequisite structure amongst knowledge items has been present in the education community for decades. Gagné coined the term “learning hierarchy” to describe the prerequisite relationship among skills, and developed methods for extracting learning hierarchies from student data. Following this work, there were a number of studies in the education literature on estimating prerequisite structure in mathematics curriculums, including work by Miller and Phillips [1974], Uprichard and Phillips [1977], and Close and Murtagh [1986].

More recently there has been interest from the user modeling and educational data mining community. Desmarais, Maluf and Liu [1996] introduced a method for calculating partially ordered knowledge structures (POKS), and Pavlik, Cen, Wu and Koedinger [2008] have recently provided a hierarchical agglomerative clustering method for student tutoring data which computes and leverages POKS as an input to this clustering.

However, to our knowledge, none of these studies explicitly modeled the observation generation process that might distort measurements of student knowledge and affect the inferred prerequisite structure.

3. METHOD

We now discuss our preliminary data analysis towards inferring prerequisite structure (if any) among a set of N skills. The data is assumed to be a set of binary observations that indicate if a particular student got each skill correct or incorrect, collected over a set of students. For N skills there are an enormous number of potential prerequisite structures. Therefore, to start we focus on considering each pair of skills separately, and evaluate the evidence as to whether a prerequisite relation exists among the skill pair. In this initial analysis we assume all data comes from a single evaluation setting, such as a test, and focus on comparing the likelihood of the observed data under different precondition structures. We will consider alternative methods in the discussion section.

More precisely, we are interested in whether there exists a precondition relationship among each pair of skills s_i and s_j . We will consider two simple models: in the *precondition* model s_i must be mastered before s_j can be mastered, and in the *flat* model the two skills are independent. We will compare the two models by evaluating how well they model the observed student data, under the best fit of each model’s parameters. The observed data will be denoted as $z_{ik} = (0, 1)$ to represent whether student k got skill i incorrect or correct: z_{jk} denotes the observation for skill j .

We will shortly describe the different parameters employed in the two models, but first recall that we presume that a student’s mastery of a skill is not directly observed. Rather, following the knowledge tracing work of Corbett and Anderson [1995], we use a p_s slip parameter to model the probability of a student answering a question about a skill incorrectly given the student has mastered the skill, and a p_g guess parameter to model the probability of a student answering correctly even when the student has not mastered the skill in question.

In the flat model, there are two independent parameters, $p(s_i = 1)$ and $p(s_j = 1)$ which represent the probability of a student having mastered skill i and skill j respectively. These are unknown parameters. We also do not know whether or not each individual student k mastered or did not master skill i or skill j . Since our interest is in coming the likelihood of the observations z_{ik} and z_{jk} for all k , our aim is to compute the expected marginal likelihood of the observed data $z_{ik}z_{jk}$ for all students k , under the best possible fit of the hidden parameters $p(s_i = 1)$ and $p(s_j = 1)$, by summing over whether each student mastered or did not

master each skill. We estimate this using Expectation Maximization. This involves first estimating of the probability of whether or not each student mastered or did not master skill i , given whether or not they got skill i correct. For example,

$$\begin{aligned} p(s_{ik} = 1|z_{ik} = 1) &= \frac{p(z_{ik} = 1|s_{ik} = 1)p(s_i = 1)}{p(z_{ik} = 1|s_{ik} = 1)p(s_i = 1) + p(z_{ik} = 1|s_{ik} = 0)(1 - p(s_i = 1))} \\ &= \frac{(1 - p_s)p(s_i = 1)}{(1 - p_s)p(s_i = 1) + p_g(1 - p(s_i = 1))}, \end{aligned} \quad (1)$$

and similar expressions exist for $p(s_{ik} = 1|z_{ik} = 0)$, $p(s_{ik} = 0|z_{ik} = 1)$ and $p(s_{ik} = 0|z_{ik} = 0)$. Note that these probabilities can be evaluated independently for skills i and j . These probabilities allow us to compute the expected value of the log likelihood. We then calculate the values of the two model parameters $p(s_i = 1)$ and $p(s_j = 1)$ that maximize this expected log likelihood and repeat.

In the precondition model we assume skill i must be mastered before skill j . Here only 3 possible student states are allowed:

- $s_i s_j = 00$: both skills are not mastered
- $s_i s_j = 10$: skill i is mastered but skill j is not mastered
- $s_i s_j = 11$: both skills are mastered.

As the probability of these states must sum to 1, this model also only has two free parameters. We again wish to evaluate how well this model explains the observed data. Now the two skills are dependent so we must consider both when estimating a student's hidden skill mastery. For example, we can compute the probability that a student mastered both skills, given he got both skills correct, as

$$\begin{aligned} p(s_{ik}s_{jk} = 11|z_{ik}z_{jk} = 11) &= \frac{p(z_{ik}z_{jk} = 11|s_i s_j = 11)p(s_i s_j = 11)}{p(z_{ik}z_{jk} = 11|s_i s_j = 11) + p(z_{ik}z_{jk} = 11|s_i s_j = 10) + p(z_{ik}z_{jk} = 11|s_i s_j = 00)} \\ &= \frac{(1 - p_s)^2 p(s_i s_j = 11)}{(1 - p_s)^2 p(s_i s_j = 11) + (1 - p_s)p_g p(s_i s_j = 10) + p_g^2 p(s_i s_j = 00)}. \end{aligned} \quad (2)$$

We again estimated the model parameters (here, $p(s_i s_j = 00)$ and $p(s_i s_j = 10)$) using Expectation Maximization.

Our primary interest is to establish whether one model significantly better explains the observed data. We computed the expected value of the log likelihood of each model with respect to the conditional distribution of the hidden student mastery values given the estimated model parameters. As both models have the same number of parameters, we identified skill pairs i, j where the precondition model had a higher log likelihood than the flat model in one direction only (aka only for i, j or j, i but not both).

4. EXPERIMENT & RESULTS

As part of a larger study, we collected test data from 113 ninth grade students over a set of 23 linear inequality skills. Test questions were designed to include particular skills so there existed a specific mapping between each test item and skill. Each instance of each test skill was graded as correct or incorrect. As our primary interest was in estimating prerequisite structure, and not on estimating the domain parameters, we fixed $p_s = 0.3$ and $p_g = 0.2$ for all skills, and both model structures, therefore removing the issue of observation modeling.¹.

Figure 1 shows the resulting prerequisite structure induced for this particular domain. Several skills did not appear in the precondition structure. This structure should be interpreted with caution, as this analysis

¹Those particular parameter values were chosen based on expert consultation.

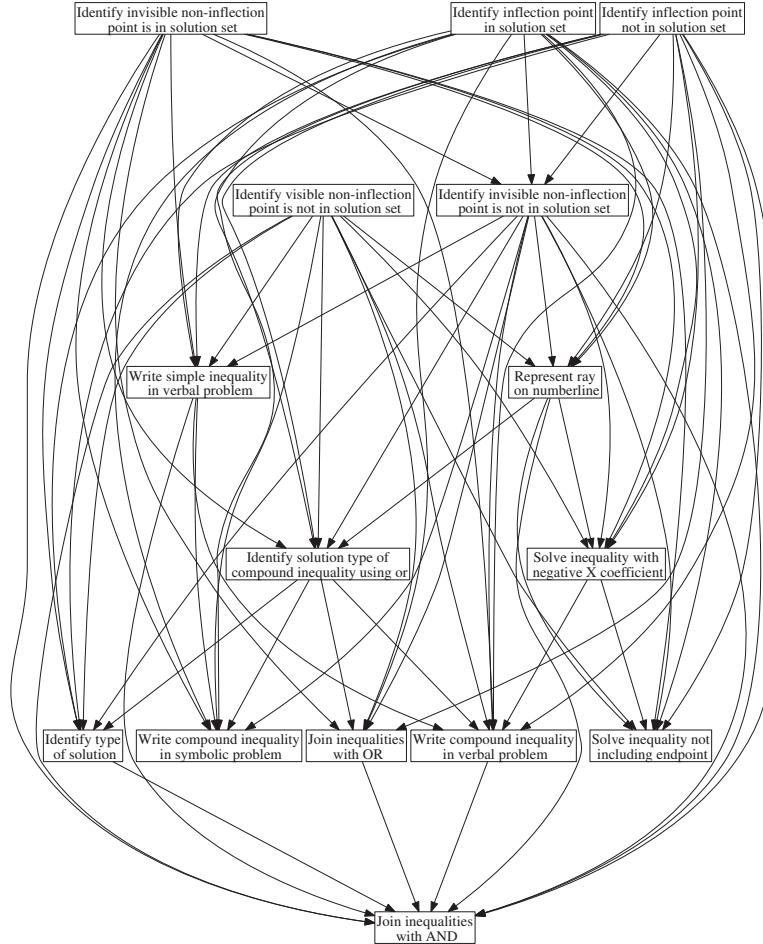


Fig. 1. Precondition structure inferred for linear inequalities curriculum. An arrow from skill i to skill j implies that i is a precondition of j .

is only preliminary and we will discuss limitations and further extensions shortly. However, some of the inferred preconditions were intuitively plausible, such as the skill of being able to write a simple inequality in a verbal problem being a precondition to the skill of writing a compound inequality in a verbal problem.

5. DISCUSSION AND FUTURE WORK

We have just described preliminary work towards inferring the prerequisite structure from noisy observations of student knowledge. In this paper we evaluated the likelihood of the observed student performance on a set

of test items given different model structures. While test data is abundant, and may be possible to obtain in settings where temporal data on student learning is not available, test data has the significant limitation that it may distort the inferred precondition structure. Ultimately we are primarily interested in the dynamics of instruction. Precondition structure could help improve automated instruction by informing whether it will be possible to successfully teach skill j if skill i has never been taught. To infer such instructional precondition structure will require analyzing temporal data of student learning. Extracting precondition structure solely from static test data can be misleading, since it both may mask a precondition between skill i and skill j (if both skills have been successfully taught, student performance on both will be very good) as well as create false preconditions (if skill i has been taught extensively, but skill j has only been recently taught, it may appear that skill i is a prerequisite for skill j , even if the two skills are independent or could have been taught in reverse order). Going forward we intend to develop approaches to computing precondition structure based on temporal data.

There are a number of other interesting issues we would like to consider. In our described approach, we assumed the observation parameters p_s and p_g were provided, and were tied to be the same across all skills. These parameters could be learned and allowed to vary amongst skill, or even problem type: for example, the probability of guessing on a multiple choice question is different than a write in answer. Another interesting issue is whether prior information about each student could be used to influence the parameters specifying the probability of a student having mastered or not mastered each skill. There exist intelligent tutoring systems that maintain a Bayesian estimate over the student knowledge state that could be used to provide such a prior. However, one limitation of leveraging such estimates would be that the Bayesian models employed by such systems make their own assumptions about the prerequisite structure, and so it might not be possible to estimate all the parameters desired. It would also be useful to compute full structures, rather than pairwise models. We are also interested in comparing to other previous approaches to inferring structure. In doing so we would need to identify an appropriate metric for comparison, which could include evaluating how well the models explain the data, or perhaps how well the models predict student performance.

Ultimately it may be impossible to infer a single correct structure. In the longer term, it is interesting to think about how one could compute teaching policies that could take as input a distribution of possible precondition structures, and select policies that were robust to uncertainty over this set of structures.

REFERENCES

- BRUNSKILL, E. AND RUSSELL, S. 2010. RAPID: A reachable anytime planner for imprecisely-sensed domains. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*.
- CLOSE, J. AND MURTAGH, F. 1986. An analysis of the relationships among computation-related skills using a hierarchical-clustering technique. *Journal for Research in Mathematics Education* 17, 2, 112–129.
- CONATI, C., GERTNER, A., AND VANLEHN, K. 2002. Using Bayesian networks to manage uncertainty in student modeling. *Journal of User Modeling and User-Adapted Interaction* 12, 371–417.
- CORBETT, A. AND ANDERSON, J. 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction* 4, 253–278.
- DESMARAIIS, M., MALUF, A., AND LIU, J. 1996. User-expertise modeling with empirically derived probabilistic implication networks. *User Modeling and User-Adapted Interaction* 5, 283–315.
- GAGNÉ, R. AND BRIGGS, L. 1974. *Principles of Instructional Design*. Holt, Rinehard, and Winston.
- MILLER, P. AND PHILLIPS, E. R. 1974. Developed of a learning hierarchy for the computational skills of fractional number subtraction. In *American Educational Research Association Meeting*.
- PAVLIK JR., P., CEN, H., WU, L., AND KOEDINGER, K. 2008. Using item-type performance covariance to improve the skill model of an existing tutor. In *Proceedings of the 1st International Conference on Educational Data Mining (EDM)*.
- UPRICHARD, A. E. AND PHILLIPS, E. 1977. An intraconcept analysis of rational number addition: A validation study. *Journal for Research in Mathematics Education* 8, 1, 7–16.

What can closed sets of students and their marks say?

Dmitry Ignatov and Serafima Mamedova and Nikita Romashkin and Ivan Shamshurin, University – Higher School of Economics, Russia

This paper presents an application of formal concept analysis to the study of student assessment data. Formal concept analysis (FCA) is an algebraic framework for data analysis and knowledge representation that has been proven useful in a wide range of application areas such as life sciences, psychology, sociology, linguistics, information technology and computer science. We use the FCA approach to represent the structure of an educational domain under consideration as a concept lattice. In this paper, we aim at building lattice-based taxonomies to represent the structure of the assessment data to identify the most stable student groups w.r.t the students achievements (and dually for courses marks) at certain periods of time and to track the changes in their state over time.

1. INTRODUCTION

Formal Concept Analysis (FCA) [Wille 1982; Ganter and Wille 1999] is an algebraic data analysis technique for building categories (formal concepts) defined as object sets sharing some attributes, irrespectively of a particular domain of application (for example, [Ignatov and Kuznetsov 2009]). FCA provides an analyst with a convenient and algebraic definition of a formal concept as a unit of human thinking. This definition is closely related to the philosophical notion of the "concept" characterized extensionally by the set of entities it covers and intensionally by the set of properties they have in common. In our study the set of objects (entities) comprises students and the set of attributes (properties) contains students' marks on different courses. The concepts form a taxonomy called a concept lattice w.r.t. specialization (generalization) relation on formal concepts. This taxonomy allows an analyst to study relationships between different groups of objects (dually, attributes) and find some interesting and potentially useful implications between their attributes. In contrast to conventional clustering techniques, FCA provides us with a kind of cluster, called formal concept, that captures similarity of objects in it by the set of shared attributes, and dually for attributes. In conventional clustering techniques we commonly have only the value of objects' similarity which comprises the cluster. The aim of this paper is to reveal some homogeneous groups of students w.r.t. their marks in term assessment data and to trace evolution of such groups in different terms of study by means of FCA. We have to add that all FCA and data mining tools such as concept stability, iceberg lattices, intensionally and extensionally related concepts were successfully used for building taxonomies of epistemic communities [Roth et al. 2006; Kuznetsov et al. 2007] and web users [Kuznetsov and Ignatov 2009]. Therefore, the main point of the paper is to show how this powerful inventory can be useful for the Educational Data Mining domain.

2. BASIC NOTIONS

Before we describe our main analyses, we briefly introduce the FCA terminology. A triple $\mathbb{K} = (G, M, I)$ is called a *formal context*, where G is a set of *objects*, M is a set of *attributes*, and the binary relation $I \subseteq G \times M$ shows which objects have which attributes. The derivation operators $(\cdot)'$ are defined for $A \subseteq G$ and $B \subseteq M$ as follows:

This work is partially supported by the Russian Foundation for Basic Research, grant # 08-07-92497-NTSNIL_a.
Corresponding author's address: D. Ignatov, Zelenaya ulitsa, 17-52, Kolomna, Russia, 140410; email: dignatov@hse.ru;

$$\begin{aligned} A' &= \{m \in M | gIm \text{ for all } g \in A\}; \\ B' &= \{g \in G | gIm \text{ for all } m \in B\}. \end{aligned}$$

In that way, A' is the set of attributes common to all objects of A and B' is the set of objects sharing all attributes of B .

The double prime operator $(\cdot)'$ forms a closure operator, i.e., $(\cdot)''$ satisfies the properties – extensity, monotony, and idempotency. So, sets A'' and B'' are said to be *closed*.

A (*formal*) concept of the context $\mathbb{K} = (G, M, I)$ is a pair (A, B) , where $A \subseteq G$, $B \subseteq M$, $A = B'$, and $B' = A$. One can see that $A'' = A$ and $B = B''$ in this case. The set A is called the *extent* and B is called the *intent* of the concept (A, B) .

A concept (A, B) is called a superconcept of (C, D) if $C \subseteq A$ (which is equivalent to $B \subseteq D$). In this case, (C, D) is a subconcept of (A, B) , and we write $(C, D) \leq (A, B)$. The set of all concepts ordered by \leq forms a lattice and called the concept lattice $\underline{\mathfrak{B}}(\mathbb{K})$ of the context \mathbb{K} .

2.1 Iceberg Lattices and Stability Indices

The main drawback of concept lattice taxonomies is their huge size even for relatively small contexts. For example, for a context $\mathbb{K} = (G, M, I)$, such that $|G| = |M| = 10$ in the worst case we have as a result $2^{10} = 1024$ concepts. However, real data are often sparse and the number of concepts is rather moderate to visualize their taxonomy. The purpose of an analyst is to reveal some interesting groups of individuals defined as concept extents. There are some remedies to cope with the huge size of concept lattices and to help find relevant concepts.

One of such well-known approaches in Data Mining is to find all frequent concepts, so called "iceberg lattices" [Stumme et al. 2002]. The *iceberg lattice* of a concept lattice $\underline{\mathfrak{B}}(\mathbb{K})$ is a set $\{(A, B) | |A|/|G| \geq \theta\}$, where $(A, B) \in \underline{\mathfrak{B}}(\mathbb{K})$ and θ is a given threshold such that $0 \leq \theta \leq 1$. An iceberg lattice is just an upper part of the concept lattice or its order filter. However, one should be careful not to overlook small but interesting groups, for example, groups not yet represented by a large number of objects, or, groups that contain "emergent" objects which are not among members of any other group. There is an additional problem: the presence of noise in data may result in many similar concepts in the concept lattice. Considering the upper part of the lattice does not solve the problem, since this part may contain a lot of such similar nodes.

To solve the problem of selecting "sound" concepts (or their intents), many FCA practitioners use the notion of concept stability[Kuznetsov 2007; Kuznetsov and Ignatov 2009]. Let $\mathbb{K} = (G, M, I)$ be a formal context and (A, B) be a formal concept of \mathbb{K} . The stability index, σ , of (A, B) is defined as follows:

$$\sigma(A, B) = \frac{\{C \subseteq A | C' = B\}}{2^{|A|}}.$$

It is obvious that $0 \leq \sigma \leq 1$. Stability indicates how much the concept intent depends on particular objects of the concept extent. Thus, a stable intent is less sensitive to noise in object descriptions. In other words, stability measures how much the group of students depends on some of its individual members. In this respect, contexts where students are objects and students' marks on courses which they study are attributes are particularly adequate: here, formal concepts represent student's communities as groups of marks' on courses along with corresponding students. Removing a few students from the context should not change drastically the well-studied (or worse-studied) courses of a student community – "real" student communities ought to be stable in spite of noisy data. In a dual manner, we can define an extensional stability index, which indicates how a concept extent depends on particular attributes. In our domain it helps to answer the question: would the students of a given concept still belong to the same category if they stop sharing the same level of achievements on some courses?

By these means we are able to reveal stable groups of high achievers and well-studied courses as well as poor students and worse-studied subjects.

2.2 Scaling

In this paper we have dealt with student marks, so, our contexts did not contain binary attributes. Such contexts are called many-valued contexts. There is a technique to transform a many-valued formal context into a conventional single-valued context. This technique is called conceptual scaling. The main idea of conceptual scaling is to represent one many-valued attribute of the initial many-valued context by some binary attributes. There are some different kinds of scaling; we prefer nominal scaling, where the attribute values are not comparable, and ordinal scaling, where attribute values are comparable to each other.

2.3 Dynamic Mappings

Let $\mathbb{K}_1 = (G, M, I)$ and $\mathbb{K}_2 = (H, N, J)$ be two contexts describing the same class of students in two different time periods (or points). How has the situation changed between these time periods? In particular, if (A, B) is a concept of \mathbb{K}_1 , what has happened to it in \mathbb{K}_2 ? Let's consider a concept $(C, D) \in \mathfrak{B}(\mathbb{K}_2)$; if the closure of $B \cap D$ is equal to $B \in \mathbb{K}_1$ and $D \in \mathbb{K}_2$, we say that (A, B) and (C, D) are *intensionally related* [Kuznetsov et al. 2007]. In the case of student assessment data, concepts intensionally related to (A, B) represent the evolution of the student's achievements in the disciplines B between two periods. Dually we can define the notion of *extensionally related* concepts. Two concepts $(A, B) \in \mathfrak{B}(\mathbb{K}_1)$ and $(C, D) \in \mathfrak{B}(\mathbb{K}_2)$ are said to be extensionally related if $(A \cap C)^{II} = A$ and $(A \cap C)^{JJ} = C$, where $(.)^{II}$ and $(.)^{JJ}$ are closure operators of the contexts \mathbb{K}_1 and \mathbb{K}_2 respectively.

Sometimes, in addition to main definition requirements for intensionally (dually extensionally) related concepts (A, B) and (C, D) , it is useful to use such constraints as follows: $|B \cap D| \geq n\% \cdot |B|$ and $|B \cap D| \geq n\% \cdot |D|$. This helps us to reduce the number of related concepts.

3. EXPERIMENTS AND RESULTS

3.1 Data

In our study we consider two assessment datasets describing students who entered the university in 2006 and 2007 respectively. By means of intensional (extensional) relatedness we are trying to find main trends in students achievements and to understand which disciplines were the most complicated.

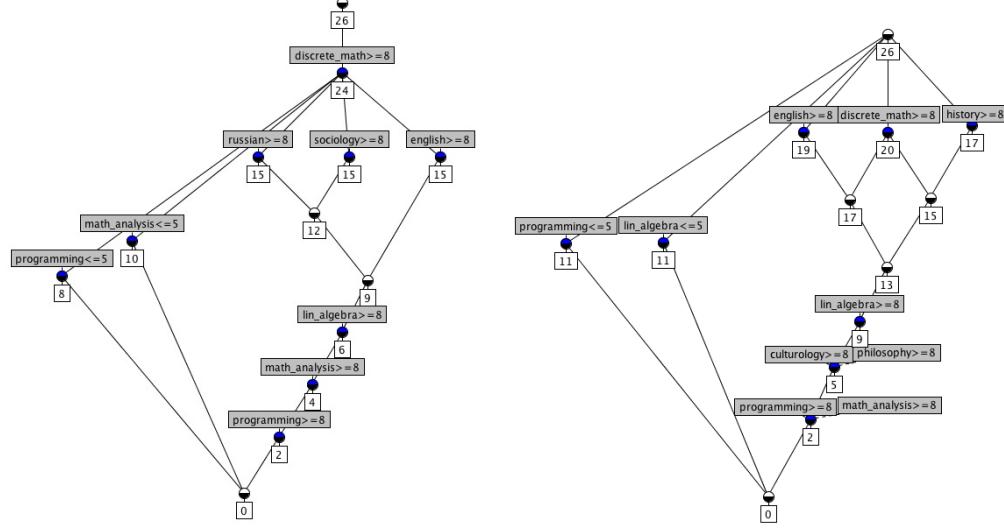
We analyse term marks of students who were studying at the department of "Applied Mathematics and Information Science" in two different academic years 2006/2007 and 2007/2008. It's worth to mention that our university maintains 10-grades system of assessment; more precisely, marks from 1 to 3 mean unsatisfactory results, 4 and 5 show satisfactory results, marks 6 and 7 indicate good results, and marks greater or equal 8 certify excellent achievements.

We compose several different contexts and conduct experiments (see subsection 3.3)

3.2 Preprocessing

We have to transform our multi-valued contexts to single-valued. To this end we introduce the following rules: for each course we consider 3 single-valued attributes, namely $[course_name] \leq 3$ (a student has failed the final exam), $[course_name] \leq 5$ (an exam is satisfactory passed or failed), $[course_name] \geq 8$ (an exam is perfectly passed). Thereby, derived contexts have exactly $3|M|$ attributes. Scaling is a matter of interpretation, but in our case, by doing so, we are able to divide successful students and those who have some difficulties in studying. Cutting off students with good marks helps us to better sort out high achievers and mediocre students.

Fig. 1. Line diagrams of 13 most stable concepts for the first term (left) and the second term (right) of 2006/2007 academic year



3.3 Experiments

For each experiment we build a taxonomy of N most stable concepts of a certain context and then compare the taxonomies' diagrams.

3.3.1 Entrants of 2006: comparison in terms of extensional relatedness between the first part of 2006/2007 and the second one. Let us compare two diagrams on figure 1. They have a similar structure; at the right top of each diagram there are three incomparable nodes with 3 disciplines that are likely to be easily passed. In the first term they are Russian Language, Sociology, and English Language, in the second term – English Language, Discrete Mathematics, and History. Our approach shows us that the concept X of those who successfully passed Discrete Mathematic, Russian Language and Sociology extensionally related to the concept Y of those students who have 8 or higher mark in Discrete Mathematics, English Language, and History. At the same time one can see that some people have moved along humanities and there are students who became more successful in mathematical disciplines. Just the thing what we need to trace dynamics is related concepts by the closure of student sets intersection. Also the students from the extent of X are related to students with best results in Discrete Mathematics, English Language, and Calculus.

In the left part of the diagram there are those disciplines that cause the most difficulty. We can also derive that not all of those students who passed programming with the mark 5 or below on the second term have had any trouble with this discipline previously. Besides programming, the courses that caused the most difficulty were Analysis in the first term (let us denote corresponding concept as Z_1) and Linear Algebra in the second term (concept Z_2). A legitimate question arising from the similarity of the two diagrams would be whether or not Z_1 and Z_2 are extensionally related (connected). In fact, they are, and thus the intent of these formal concepts is comprised of mostly the same students.

3.3.2 Entrants of 2006 and 2007: comparison in terms of intensional relatedness between the first part of 2006/2007 and 2007/2008. The other diagrams are omitted due to a lack of space. On the whole, in the first semester of 2006 student marks were higher than in first semester of 2007. The requirements in Discrete Mathematics were made stricter: if in the former time period only two students received mark 7 or lower (7.7

percent), in the latter time period the fraction of such students rose to 94 percent. In the 2006 set we see a rather large block of "humanities oriented students" (the ones who successfully passed English, Russian, Sociology, and Discrete Mathematics (because 94 percent of "humanities oriented students" did pass Discrete Mathematics with marks 8 or higher)) that consists of 9 students. It is intensionally related with the set of students in the year 2007 who successfully passed English and Russian (17 students), humanities, Linear Algebra, Analysis, and Discrete Mathematics (3 students). Taking into account that the 2007 set contains twice as many students, it has become more challenging to pass the humanities (17 students passed English and Russian with excellent marks), and it seems more likely that there will be no "humanities oriented students" (everyone who passed the humanities are also strong math students). However there are also some positive trends. It turns out that in order to be good at programming, one does not necessarily have to be good at math or humanities, therefore, more students received excellent marks for programming.

3.3.3 Entrants of 2006 and 2007: comparison in terms of intensional relatedness between the second part of 2006/2007 and 2007/2008. The formal concept "poor students of the second semester second year enrolled in 2006" contains 5 students in its intent, of which 4 students subsequently dropped out, so the students enrolled in 2007 that comprise an intensionally related concept, become the most likely drop out candidates. We also note that in the 2007 there formed a group of students who had difficulty with philosophy (40% of the class), which were at the same time successful in Linear Algebra (8 %), Analysis, and English. Among those 40% none were good at History, which makes sense. It is also interesting to note some success dependencies for Linear Algebra: whereas 2006 students had to receive an excellent grade in Linear Algebra in order to receive mark 8 or above for Analysis, students enrolled in 2007, would not receive 8 for Linear Algebra unless they had successfully passed the Analysis course.

3.3.4 Entrants of 2007: comparison in intensional terms between two student subgroups in the first part of 2007/2008. Using a concept of intensional relatedness and diagram analysis we can make the following conclusions: in general the 272 group is more successful in studying English (52% who passed with excellent marks as opposed to 38%), which is hardly surprising – it is likely that groups were formed according to the students different level of command of the English language. Comparative analysis lead us to the idea that people who were proficient in English could afford to lose a couple of points on the math entrance exam. For that reason, Linear Algebra and Analysis grades are lower in group 271. We are not going to see a stable concept "those who passed Discrete Mathematics with mark 8 or higher" on the first diagram. Also, in 272 group we don't see students that are only good at languages (and of those there are 6). These students also have good results in Linear Algebra. The corresponding formal concepts are intensionally related. A common rule for both groups holds true: those who are good at Russian have no difficulty with Sociology. The reason being that the final Sociology exam is given in a form of an essay, which is standard practice in Russian classes. If we look on the very top level, then we note there are 5 formal concepts in both cases: "Russian ≥ 8 ", "Linear Algebra ≥ 8 ", "English ≥ 8 ", "Programming ≤ 5 ", "Analysis ≤ 5 ". So the student performance structure is very similar, which is what to be expected considering that we are comparing the same class (just different groups), that have common subjects, and common teaching staff. Some internal differences are of quantitative nature. According to our hypothesis, group 272 has a more solid math background and is better organized.

3.3.5 Entrants of 2007: comparison in intensional terms between two student subgroups in the second part of 2007/2008. Let's start top down. In general, we observe some quantitative differences. Core formal concepts (first level formal concepts): common concepts – "Analysis ≥ 8 ", differences – group 272 does not have a stable concept "philosophy ≤ 5 ", on the other hand, in group 271, there are no failing History students. In group 272 there are no clear failing Linear Algebra students. It is very noticeable that the second diagram contains two formal concepts the intents of which contain both mathematics and humanities. This indicates that either success or failure in one field correlates with success or failure in the other. So students are

either well organized and therefore are successful overall, or fail everything. In 272 diagram, there is one concept that stands out – "Linear Algebra ≤ 5 " It strongly correlates with formal concepts that deal with academic failure, and does not have anything to do with English proficiency. So those students who did not get on well with programming, Culture, and Discrete Mathematics could not pass Linear Algebra. "English ≥ 8 " correlates with other disciplines ≥ 8 , whereas in the first diagram "English ≥ 8 " correlates with "programming ≤ 5 ", "Culture ≤ 5 " and "Analysis ≤ 5 ". So there is a negative correlation. It is not surprising considering that English does not present any major difficulty for 271. In general, there are no other important differences, besides the ones mentioned above.

4. CONCLUSION

In this paper we proposed to use closed sets of students and their marks to reveal some interesting patterns and implications in student assessment data, especially to trace dynamics. We restricted ourself only to a few useful techniques from FCA, but there are a lot of tools which can help an analyst to explore the assessment data. For example, to better represent and mine multi-valued numerical context we can use so called interval pattern structures [Ganter and Kuznetsov 2001]. Also, there are some alternative approaches to dynamics mappings, for example, the theory of multicontexts [Wille 1996]. We suppose that existing FCA-based techniques can be potentially useful in the other fields of EDM domain.

ACKNOWLEDGMENTS

We would like thank Jonas Poelmans, Katholieke Universiteit Leuven and Jane Ilyina, U-HSE for their help and support.

REFERENCES

- GANTER, B. AND KUZNETSOV, S. O. 2001. Pattern structures and their projections. In *ICCS*, H. S. Delugach and G. Stumme, Eds. Lecture Notes in Computer Science Series, vol. 2120. Springer, 129–142.
- GANTER, B. AND WILLE, R. 1999. *Formal concept analysis: Mathematical foundations*. Springer, Berlin-Heidelberg.
- IGNATOV, D. I. AND KUZNETSOV, S. O. 2009. Frequent itemset mining for clustering near duplicate web documents. In *ICCS*, S. Rudolph, F. Dau, and S. O. Kuznetsov, Eds. Lecture Notes in Computer Science Series, vol. 5662. Springer, 185–200.
- KUZNETSOV, S. O. 2007. On stability of a formal concept. *Ann. Math. Artif. Intell.* 49, 1–4, 101–115.
- KUZNETSOV, S. O. AND IGNATOV, D. I. 2009. Concept stability for constructing taxonomies of web-site users. In *Satellite Workshop "Social Network Analysis and Conceptual Structures: Exploring Opportunities" at the 5th International Conference Formal Concept Analysis (ICFCA'07), Clermont-Ferrand, France*. 19–24.
- KUZNETSOV, S. O., OBIEDKOV, S. A., AND ROTH, C. 2007. Reducing the representation complexity of lattice-based taxonomies. In *ICCS*, U. Priss, S. Polovina, and R. Hill, Eds. Lecture Notes in Computer Science Series, vol. 4604. Springer, 241–254.
- ROTH, C., OBIEDKOV, S. A., AND KOURIE, D. G. 2006. Towards concise representation for taxonomies of epistemic communities. In *CLA* (2008-04-15), S. B. Yahia, E. M. Nguifo, and R. Belohlavek, Eds. Lecture Notes in Computer Science Series, vol. 4923. Springer, 240–255.
- STUMME, G., TAOUIL, R., BASTIDE, Y., PASQUIER, N., AND LAKHAL, L. 2002. Computing iceberg concept lattices with titanic. *Data & Knowledge Engineering* 42, 2, 189–222.
- WILLE, R. 1982. Restructuring lattice theory: an approach based on hierarchies of concepts. In *Rival, I. (ed.): Ordered Sets*. Boston, 445–470.
- WILLE, R. 1996. Conceptual structures of multicontexts. In *ICCS*, P. W. Eklund, G. Ellis, and G. Mann, Eds. Lecture Notes in Computer Science Series, vol. 1115. Springer, 23–39.

How university entrants are choosing their department? Mining of university admission process with FCA taxonomies.

NIKITA ROMASHKIN, DMITRY IGNATOV and ELENA KOLOTOVA, National Research University – Higher School of Economics, Moscow, Russia

The aim of this paper is to present a case study in the analysis of university applications to the Higher School of Economics (U-HSE), Moscow. Our approach uses lattice-based taxonomies of entrants' decisions about undergraduate programmes. These taxonomies were built by means of Formal Concept Analysis (FCA). FCA is a well-known algebraic technique for object-attribute data analysis. Admission data as well as formalised survey data were used to reveal possibly significant factors of entrants' decisions. In this paper we argued that institutional characteristics of the admission process are highly correlated with entrants' choice. The obtained results are helpful to the university to correct the structure and positioning of its undergraduate programmes.

Key Words and Phrases: University admissions, formal concept analysis

1. INTRODUCTION

The aim of this paper is to present a case study in the analysis of university applications to the Higher School of Economics (U-HSE), Moscow.

Decision-making process of potential students is a popular topic among researchers of higher education, because of its value for understanding university positioning. Nevertheless, not many works cover speciality choice issue - much more of papers study factors influencing university choice. Many papers on speciality choice try to investigate factors influencing prospective students' decision on particular faculty selection (e.g., [Chang et al. 2006]). Some investigations are focused on speciality choice during last courses of study (e.g., [O'Herrin et al. 2003]). We found paper [Akbulut and Looney 2007] most relevant to our study - it tests a model aimed at identifying and explaining the mechanisms that shape student choice of computer science.

To analyse the data we mainly use a well-known algebraic technique called Formal Concept Analysis (FCA) [Ganter and Wille 1999]. There are applications of FCA in such fields as linguistics [Priss 2005], social networks analysis [Freeman and White 1993; Roth et al. 2006; Kuznetsov and Ignatov 2009], machine learning [Kuznetsov 2004], data mining [Poelmans et al. 2011] and software engineering [Tilley et al. 2005].

A common usage of FCA implies building of so-called concept lattices based on corresponding formal contexts. From a graph-theoretic point of view, a formal context is a bipartite graph where one part is a set of objects and the other part is a set of attributes. An edge between an object and an attribute means that an object has an attribute. Thus in most cases we represent a context as a biadjacency matrix or in terms of FCA a cross table. In terms of bipartite graphs a formal concept is a maximal biclique of a context. In a cross table a formal concept is a maximal rectangle filled with crosses with respect to any permutation of rows and columns. A formal concept consists of an extent and an intent. An extent is the set of objects

This work was partially supported by the Scientific Foundation of the State University Higher School of Economics, grant #10-04-0017 and by the Russian Foundation for Basic Researches, grant # 08-07-92497-NTSNIL-a

Author's address: N. Romashkin, email: romashkin.nikita@gmail.com; D. Ignatov, email: dmitrii.ignatov@gmail.com; E. Kolotova, email: kolotova.e@gmail.com

which have all attributes from an intent. Similarly, an intent is the set of attributes common for all objects from an extent. A set of all concepts are ordered by a generalization relation \leq defined as follows. If $A \subseteq C$ (equivalently, $D \subseteq B$) stands for a concept extent A and another concept extent C then $(A, B) \leq (C, D)$, that is the concept (C, D) is more general than (A, B) . A set of all concepts and a relation \leq forms a concept lattice. A concept lattice can be viewed as an overlapping taxonomy for underlying categories (concepts).

The rest of the paper is organized as follows. In the next section, we present our case study, describe the admission process to U-HSE and our data, explain the data preprocessing step, and discuss the results of our analysis in the end. Section 3 concludes the paper.

2. CASE STUDY: ADMISSION PROCESS TO U-HSE

2.1 Background

Assuming probable confusion of the Russian educational system, we must say a few words about the Higher School of Economics¹ (U-HSE) and its admission process.

Nowadays U-HSE is acknowledged as a leading university in the field of economics, management, sociology, business informatics, public policy and political sciences among Russian universities. Recently a number of bachelor programmes offered by U-HSE has been increased. Currently U-HSE offers 20 bachelor programmes. We consider only bachelor programmes in our investigation. In order to graduate from school and enter a university or a college every Russian student must pass a Unified State Exam (Russian transcription: EGE), similar to US SAT–ACT or UK A-Level tests. During 2010 admission to U-HSE, entrants were able to send their applications to up to three programmes simultaneously. Some school leavers (major entrants of U-HSE bachelor programmes) chose only one programme, some – two, and some – three. Then entrants had to choose only one programme to study among successful applications.

2.2 Data

2.2.1 *General Information.* We used data representing admission to U-HSE in 2010. It consists of information about 7516 entrants. We used mainly information about programmes (up to three) to which entrants apply². Exactly 3308 entrants successfully applied at least to one programme, but just 1504 become students. Along with this data we also used the data of entrants' survey (76% of entire assembly).

Further in the paper we mostly used data for the Applied Mathematics and Informatics programme to demonstrate some results. The total number of applications to the Applied Mathematics and Informatics programme was 843, of which 398 were successful but only 72 of them were actually accepted into the program. It might seem confusing only 72 out of 398 eligible prospective students decided to enroll, but since the admission process was set up in two stages, and at each stage only 72 entrants were eligible to attend the program, some of them decided to go for a different programme or university. As a result, the number of entrants whose applications were successful in any way came down to 398. Such situation is typical for all the bachelor programmes at U-HSE.

2.2.2 *Preprocessing.* FCA requires object-attribute data. In our case objects are entrants and programmes they apply to are attributes. Together they are treated as a context. A series of contexts were constructed. Namely, we built a context for every programme where objects were entrants applying to that programme and attributes were other programmes they applied to. We built a separate context for every programme because it is meaningless to consider all programmes at once as programmes are very different in size and the resulting lattice would represent only the largest of them.

¹<http://www.hse.ru/en/>

²U-HSE is a state university, thus most of student places are financed by government. In this paper we consider only such places.

Likewise, we built a context for every programme where objects were entrants and attributes were programmes to which entrants successfully applied as well as the programmes that the entrants decided to enroll into, including those at other universities.

These contexts were then used to build concept lattices. Since the resulting lattices had too complicated a structure to interpret, we filtered concepts by their extent size (extent size is the number of objects, in our case it is the number of entrants), thus remaining concepts express only some of the more common patterns in entrants decisions.

2.3 Results

2.3.1 Entrants' choice of programmes to apply. To which programmes entrants often apply simultaneously? Trying to answer this question for every programme, we built diagrams³ similar to figure 1. Such diagrams help us to reveal common patterns in entrants choices. Let us explain what the diagram in figure 1 represents. This diagram (also known as a Hasse diagram or a line diagram) is a diagram of the partial order on formal concepts. Typical applications of FCA imply building formal concept lattices discussed earlier, but here we filter concepts by extent size to avoid complexity caused by noise in the data; this reduction technique is well-known to the data mining community as so called "iceberg lattices" [Stumme et al. 2002]. Thus the order on remaining concepts is no longer a lattice, it is a partial order. Meaning of the labels on the diagram is obvious. A label above a node is a programme, a label below a node is a percent of entrants to Applied Mathematics and Informatics programme who also applied to programmes connected to a node from above. For example, the most left and bottom node on the diagram means that five percent of applied math's entrants also apply to Mathematics and Software Engineering. Then if we look at the nodes above the current node we may notice that ten percent Applied Mathematics and Informatics applicants also apply to Mathematics programme, and 70 percent also applied to Software Engineering.

Now let us try to interpret some knowledge unfolded by the diagram in figure 1. 70 percent of entrants who applied to Applied Mathematics and Informatics also apply to Software Engineering. The same diagram for Software Engineering states that 80 percent of Software Engineering applicants also apply to Applied Mathematics and Informatics. How this fact can be explained? Firstly it can easily be explained by the fact that these two programmes require to pass the same exams. Therefore there were not any additional obstacles to apply to both programmes simultaneously. Another possible explanation is that it is uneasy for entrants to distinguish these two programmes and successful application to any of them would be satisfactory result.

Analysing diagrams of other programmes' applications we found that equivalence of required exams is probably the most significant reason to apply to more than one programme.

2.3.2 Entrants' "Efficient" choice. If an entrant successfully applied to more than one bachelor programme he or she must select a programme to study. Unlike the previous case, entrants have to select exactly one programme which gives us more precise information about entrants preferences. For that reason we define this situation as an efficient choice, efficient in the sense of more expressive about true entrants preferences. Figure 2 presents the efficient choice of entrants to Applied Mathematics and Informatics programme. The meaning of diagram labels is almost the same as in figure 1. Programmes without plus sign (+) are successful applications, programmes with preceding plus sign are programmes chosen to study by entrants. Label "- Other -" means that the entrant canceled his application preferring another university or not to study this year altogether.

Together with diagram in figure 1 this diagram provides us with more precise knowledge about preferences of entrants to the Applied Mathematics and Informatics programme. More than two thirds of entrants

³As any other data mining technique FCA implies an intensive use of software. All diagrams mentioned in this paper have been produced with meud (<https://github.com/jupp/meud-wx>). Meud is mainly developed by Nikita Romashkin and currently is in far pre-released state.

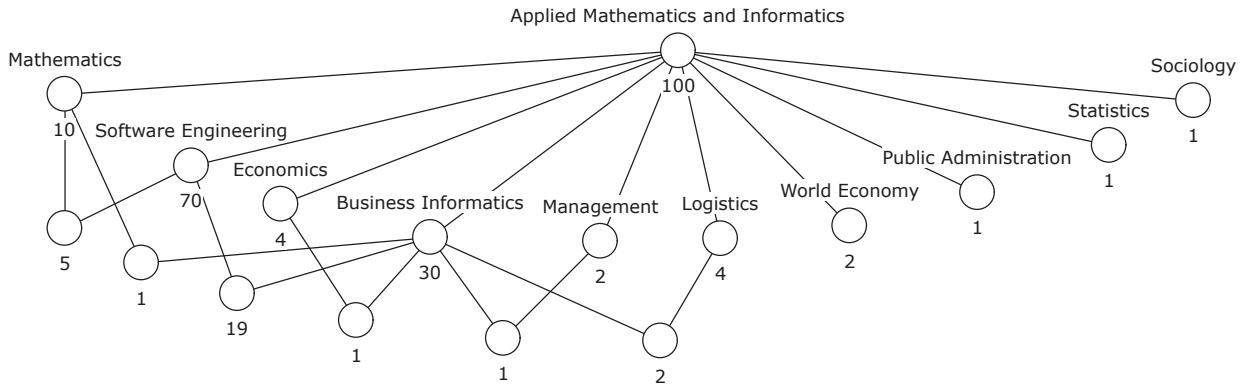


Fig. 1. Other programmes which entrants of Applied Mathematics and Informatics programme also apply.

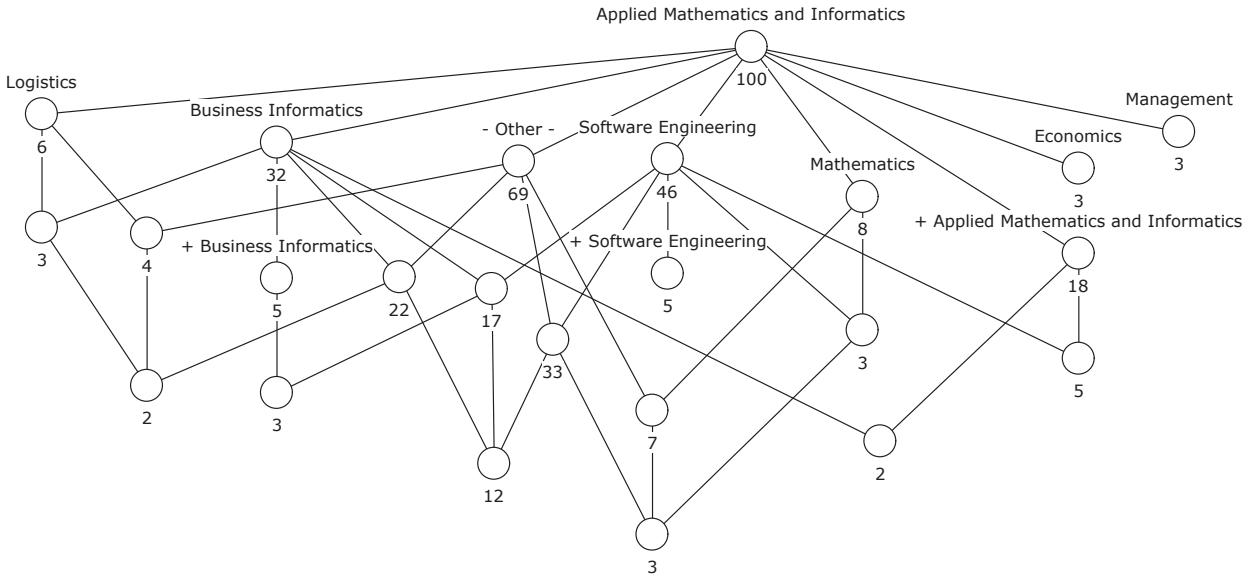


Fig. 2. "Efficient" choice of entrants to Applied Mathematics and Informatics programme.

who successfully apply to the Applied Math programme nevertheless prefer to study at another university. Whereas just 18 percent of successful applicants then become students on the Applied Mathematics and Informatics programme. Exactly 5 percent prefer to study Software Engineering and 5 percent of entrants who choose Applied Mathematics and Informatics also successfully applied to Software Engineering. It can be interpreted as equality of entrants preferences concerning these two programmes. Additionally, 5 percent prefer Business Informatics and only two percent of entrants who prefer Applied Mathematics and Informatics also successfully apply to Business Informatics, therefore in the pair Business Informatics and Applied Mathematics and Informatics the latter one is less preferable by entrants.

Here we should note that the sum of nodes percents with labels containing plus sign and node "- Other -" must equal to 100%, however here it does not because we excluded some nodes during filtering.

We built diagrams of "efficient" choice for every programme. Analysis of these diagrams helps us to recognise some relations between programmes in terms of entrants preferences. For example, some programmes in most cases is rather backup than actual entrants preference. Some programmes are close to each other by subject of study, these relations are also expressed by diagrams. With help of formalised survey data we found some possible factors of entrants' choice among some particular programmes. These knowledge can help our university to understand entrants' attitude to its undergraduate programmes and thus correct the structure and positioning of them.

3. CONCLUSION

We demonstrate a possible usage of FCA taxonomies in the field of educational data mining. After presenting some basic notions of FCA, we provide a case study of university admission process mining by means of FCA.

We present examples of diagrams obtained with help of FCA and provide a possible interpretation in two slightly different cases. Our main statement is that FCA taxonomies are a useful tool for representing object-attribute data which helps to reveal some frequent patterns and to present dependencies in data entirely at a certain level of details. Some interesting patterns then can be analysed separately and more carefully with help of other supportive data, for example, formalised survey data.

ACKNOWLEDGMENTS

We would like to thank Jonas Poelmans, Katholieke Universiteit Leuven and Galina Makarova for their help and support.

REFERENCES

- AKBULUT, A. Y. AND LOONEY, C. A. 2007. Inspiring students to pursue computing degrees. *Commun. ACM* 50, 67–71.
- CHANG, P.-Y., HUNG, C.-Y., LNG WANG, K., HUANG, Y.-H., AND CHANG, K.-J. 2006. Factors influencing medical students' choice of specialty. *Journal of the Formosan Medical Association* 105, 6, 489 – 496.
- FREEMAN, L. AND WHITE, D. 1993. Using galois lattices to represent network data. *Sociological Methodology* Volule 23, 1, 127–146.
- GANTER, B. AND WILLE, R. 1999. *Formal concept analysis: Mathematical foundations*. Springer, Berlin-Heidelberg.
- KUZNETSOV, S. O. 2004. Machine learning and formal concept analysis. In *ICFCA* (2004-02-19), P. W. Eklund, Ed. Lecture Notes in Computer Science Series, vol. 2961. Springer, 287–312.
- KUZNETSOV, S. O. AND IGNATOV, D. I. 2009. Concept stability for constructing taxonomies of web-site users. In *Satellite Workshop "Social Network Analysis and Conceptual Structures: Exploring Opportunities" at the 5th International Conference Formal Concept Analysis (ICFCA'07), Clermont-Ferrand, France*. 19–24.
- O'HERRIN, J. K., BECKER, Y. T., LEWIS, B., AND CHEN, H. 2003. Why do students choose careers in surgery? *The Journal of surgical research* 114, 2, 260–.
- POELMANS, J., ELZINGA, P., VIAENE, S., AND DEDENE, G. 2011. Formally analysing the concepts of domestic violence. *Expert Syst. Appl.* 38, 3116–3130.
- PRISS, U. 2005. Linguistic applications of formal concept analysis. In *Formal Concept Analysis* (2005-07-20). 149–160.
- ROTH, C., OBIEDKOV, S. A., AND KOURIE, D. G. 2006. Towards concise representation for taxonomies of epistemic communities. In *CLA* (2008-04-15), S. B. Yahia, E. M. Nguifo, and R. Belohlavek, Eds. Lecture Notes in Computer Science Series, vol. 4923. Springer, 240–255.
- STUMME, G., TAOUIL, R., BASTIDE, Y., PASQUIER, N., AND LAKHAL, L. 2002. Computing iceberg concept lattices with titanic. *Data & Knowledge Engineering* 42, 2, 189–222.
- TILLEY, T. A., COLE, R. J., BECKER, P., AND EKLUND, P. W. 2005. *A Survey of Formal Concept Analysis Support for Software Engineering Activities*. LNAI Series, vol. 3626. Springer-Verlag, 250–271.
- VALTCHEV, P., MISSAOUI, R., AND GODIN, R. 2004. Formal concept analysis for knowledge discovery and data mining: The new challenges. In *Concept Lattices*, P. W. Eklund, Ed. Lecture Notes in Computer Science Series, vol. 2961. Springer, 352–371.

What's an Expert? Using learning analytics to identify emergent markers of expertise through automated speech, sentiment and sketch analysis.

I. WORSLEY

Stanford University, U.S.A.

AND

II.BLIKSTEIN

Stanford University, U.S.A.

Assessing student learning across a variety of environments and tasks continues to be a crucial educational concern. This task is of particular difficulty in non-traditional learning environments where students endeavor to design their own projects and engage in a hands-on educational experience. In order to improve our ability to recognize learning in these constructionist environments, this paper reports on an exploratory analysis of learning through multiple modalities: speech, sentiment and drawing. A rich set of features is automatically extracted from the data and used to identify emergent markers of expertise. Some of the most prominent markers of expertise include: user certainty, the ability to describe things efficiently and a disinclination to use unnecessary descriptors or qualifiers. Experts also displayed better organization and used less detail in their drawings. While many of these are things one would expect of an expert, there were areas in which experts looked very similar to novices. To explain this we report on learning theories that can reconcile these seemingly odd findings, and expound on how these domain-independent markers can be useful for identifying student learning over a series of activities.

Key Words and Phrases: Learning Analytics, Multi-modal, Assessment, Speech

1. INTRODUCTION

The call to improve assessment of student learning is being raised from various fronts. National education policy mandates that schools demonstrate student advancement on a regular basis. At the same time, corporations and public institutions call for students to learn 21st century skills: creativity, collaboration and innovation. Educators, who scramble to satisfy these competing demands, find themselves at an irreconcilable crux. One solution may lie in the development of automatic natural assessment tools. Such tools can provide the automaticity needed to allow testing to be more open-ended and also offer innovative teachers new ways for assessing how their students are learning during hands-on, student-designed learning. With this in mind, our primary research question is: How can we use informal student speech and drawings to decipher meaningful “markers of expertise” (Blikstein 2011) in an automated and natural fashion?

2. PRIOR WORK

This research builds on a growing tradition in artificial intelligence in education that uses various techniques to uncover correlations between student artifacts and efficacious learning. Previous work includes a variety of examples from intelligent tutoring systems that leverage: discourse analysis (Litman et al 2009, Forbes-Riley et al 2009), content word extraction (Chi et al 2010, Litman et al 2009), uncertainty detection (Liscombe et al 2005), sentiment analysis (Craig et al 2008, D'Mello et al 2008, Conati 2009), linguistic

analysis, prosodic and spectral analysis, and multi-modal analysis (Litman et al 2009, Forbes-Riley and Litman 2010). Other examples from the education context include automatic essay grading (Chen et al 2010, Rus et al 2009) and educational robots. The present study also extends our previous work (Blikstein and Worsley 2011) and explores the salience of the aforementioned analysis techniques in characterizing open-ended learning.

3. DATA

The data for this study comes from interviews with 15 students from a tier-1 research university. Of the 15 students, 8 were women, 7 were men; 7 were from technical majors, 3 were undergraduates, and 12 graduate students. There were 3 novices, 9 intermediates, and 3 experts and each interview took approximately 30 minutes. Participants were asked to draw and think aloud about how to build various electronic and mechanical devices. The questions were posed in a semi-structured clinical interview format. Question 1, the control question, asked the student to construct a temperature control system, while question 2 challenged the student to design a device to automatically separate, glass, paper, plastic and metal. Student speech was transcribed by graduate and undergraduate students. Prior to the interviews, the subjects were labeled as being experts, intermediates or novices in engineering and robotics. This classification was based on previous formal technical training either through a degree program or through a lab course on physical computing. This classification is in accordance with theory that suggests that experts are those that have had extended time practicing their skill. The data consisted of audio files, transcriptions of the interviews, and digitized drawings that the students produced during the interview.

4. DATA ANALYSIS

In accordance with previous literature, this study utilized the following techniques for feature extraction: crowd-sourcing using Mechanical Turk to determine human ratings of each transcript; prosodic analysis - pitch, intensity and duration – and spectral analysis – the first three formants - using the Praat software (Boersma and Weenick, 2010); linguistic analysis - pauses, filled pauses, restarts – using the Python Natural Language Toolkit; sentiment analysis using the Linguistic Inquiry and Word Count (LIWC) and the Harvard Inquirer; content word analysis, using web-mined lexicons from chemistry, mathematics, computer science, material science and general science; dependency parsing using the Stanford Parser (Klein and Manning, 2003); n-gram analysis; and human coded drawing analysis based on the work of Song and Agogino 2004, Shah et al. 2003 and Anderson 2000.

The data was analyzed using expectation maximization (EM) with an intra-cluster Euclidean distance objective function. Before running EM, each feature value was modified to have unit variance and zero mean. Additionally, t-tests were performed to check statistical significance.

5. RESULTS

The complete analysis involved nearly 200 features, excluding n-grams. For the sake of brevity, we will only report on a subset of features. From the drawing analysis data in Table 1, we see little variation across the classes. Moreover, the only statistically significant class differences exist between novices and non-novices. These differences are observed for ‘space used’ and dimensions.

Table 1 - A comparison of the average drawing features scores across expertise types. Text Annotation ranges from 0 to 1, while, Space Used, Is 3-D and dimensions range from 1 to 3. Finally, the remaining scores were rated on a 10 point scale.

Class	Text Annotation	Space Used	Abstraction	Is 3-D	Detail	Organization	Dimensions
Novice	1.00	2.47	5.97	1.64	5.83	5.94	2.11
Intermediate	0.78	1.85	4.60	1.63	4.31	5.46	2.05
Expert	1.00	1.92	6.27	1.10	4.85	8.25	1.55

Similar class-based statistics are reported for significant linguistic, prosodic and sentiment features. Table 2 presents the linguistic and prosodic results, while Table 3 presents the sentiment analysis results.

Table 2 – The normalized average duration, pitch, intensity and number of disfluencies, among the different classes.

Expertise	Duration	Pitch	Disfluencies	Intensity
Novice	1.16	0.7	1.06	-0.39
Intermediate	-0.2	-0.09	-0.52	0.02
Expert	-0.56	-0.42	0.5	0.32

To provide additional clarification about the linguistic and prosodic data trends, consider that the average duration of a novice answer was nearly twice as long as that of an expert.

Table 3 – Average normalized word count for various sentiment words from LIWC and the Harvard Inquirer

Class	Positive	Strong	Weak	Understate	Quality	Quantity	Certainty
Novice	0.063	0.058	0.032	0.053	0.020	0.065	0.014
Intermediate	0.041	0.068	0.033	0.039	0.028	0.068	0.022
Expert	0.039	0.077	0.024	0.036	0.012	0.079	0.022

Finally, we present the centroids that were obtained from doing clustering analysis with both sentiment and speech features.

Table 4 - EM cluster centroid values for the features included in a combined sentiment and speech analysis. Values have been normalized to unit variance and zero mean. Duration is in seconds, while the other values are in words per transcript.

	Novice	Intermediate	Expert
Duration (s)	0.97	-0.30	-1.23
Filled Pauses	-0.13	-0.51	1.78
LIWC Neg	0.32	-0.52	1.3
SureLw	-0.68	0.26	0.65
Quality	-0.55	0.62	-1.1

6. DISCUSSION

Of particular interest to this study is the presence of several features that accurately predict expertise based on certainty. Though not presented at length, preliminary analysis of this data involved extracting n-grams from each transcript and looking for patterns across the different classes. Not surprisingly, n-grams that indicated uncertainty, eg. “don’t know”, “well, you know” were more common among novices than among non-novices. These initial results confirm a theory previously presented by Beck, in Bruer (1993) which indicates that increasing expertise tends to increase student self-confidence. These results were further corroborated in our later analysis through the certainty (SureLw) and understatement (Undrst) features. Certainty was much more common among experts, while understatements were more frequently employed by novices. Furthermore, we saw subtle leanings towards certainty through the “strong” and “weak” features, which were more prevalent among experts and novice, respectively.

The observed results concerning the decrease in duration for intermediate and expert participants, as compared to novices, also suggests that more advanced users are more certain in their approach. However, the decrease in duration is also in accord with work by Anderson and Schunn’s ACT-R theory, which describes how experts have greater facility in accessing the necessary declarative and procedural skills needed to solve complex problems, simply due to their increased exposure to them. More specifically, Anderson and Schunn (2000) completed a similar study in which they observed a substantial decrease in time needed to complete geometry proofs as students spent more time working on them.

Somewhat unexpected was the lack of meaningful results from the drawing analysis and content word analysis. The initial hypothesis for the drawing analysis assumed that more expert individuals would be capable of providing superior drawings of the system because of an improved mental representation of the required components (Anderson 2000). Instead we found little to no correlation between our features and expertise. Even in the case of our organization metric which showed a statistically significant difference between classes, the correlation coefficient was 0.13. We attribute some of this ambiguity to the drawings having a different audience for different research participants. Certain participants viewed the drawings as artifacts that they were making for the researchers, whereas others viewed the drawing space as a place for them to take notes, and simply get their thoughts on paper.

Similarly, the content word analysis failed to provide meaningful features for distinguishing experts from novices. While this may suggest that the task was not sufficiently difficult, a more likely explanation may be related to the informal nature of the interaction. According to Brown and Spang (2008) the language of science and mathematics are decidedly different from the language of everyday conversation. Because of this, it is unlikely that students will employ noticeably different levels of science and mathematics terminology in informal settings. Additionally, the nature of this open-ended design space is that people will bring previous knowledge from a variety of backgrounds and use that to solve problems. As such, it could be perfectly conceivable for a computer scientist, chemical engineer and mechanical engineer to all come up with expert solutions to a problem using completely different nomenclature.

Taken together, these results provide additional validation for the need to develop novel assessment techniques that leverage natural student artifacts: speech and drawings.

7. CONCLUSION

This study has explored a set of domain-independent markers of expertise that can allow educators and researchers to recognize student learning through analyzing student

speech, and, to a lesser extent, drawings. Using speech as a form of assessment certainly presents some challenges, but has the potential to introduce innovative ways for understanding and predicting learning in open-ended learning environments. This ability to assess non-traditional learning should help open the door to more widespread adoption of experiential learning practices, and an associated increase in 21st century competencies. Thus far our work has been exploratory. We performed in-depth analysis on a small sample size in order to better inform the types of features that we need to be looking for in future work. This initial work points to user uncertainty, as perceived through various modalities, as an influential indicator of student development. In future work we plan to further validate our findings through larger scale, longitudinal studies in constructionist learning environments.

REFERENCES

- Harvard Inquirer. http://www.wjh.harvard.edu/~inquirer/spreadsheet_guide.htm
- Linguistic Inquiry and Word Count <http://liwc.net/liwcdescription.php>
- ANDERSON, J.R. 2000. Cognitive Psychology and Its Implications, Fifth Edition. Worth Publishing.
- ANDERSON, J.R. & SCHUNN, C.D. 2000. Implications of the ACT-R Learning Theory: No Magic Bullets in R. Glaser (Ed.), *Advances in instructional psychology* (Vol. 5). Mahwah, NJ.
- BLIKSTEIN, P. 2011. Using learning analytics to assess students' behavior in open-ended programming tasks. *Paper presented at the I Learning Analytics and Knowledge Conference*, Banff, Canada.
- BLIKSTEIN, P. & WORSLEY, M. 2011. Learning Analytics: Assessing Constructionist Learning Using Machine Learning. *Paper presented at the American Educational Research Association Annual Meeting*, New Orleans, USA.
- BOERSMA, P., AND WEENINK, D. 2010. Praat: doing phonetics by computer [Computer program]. Version 5.2.03, <http://www.praat.org/>. Design Studies 19: 431-453
- BROWN, B. A., AND SPANG, E. 2008, Double talk: Synthesizing everyday and science language in the classroom. *Science Education*, 92: 708–732.
- BRUER, J.T. 1993. Schools for Thought: A science of learning in the classroom. MIT Press.
- CHEN, Y., LIU, C., LEE, C., AND CHANG, T. 2010, "An Unsupervised Automated Essay Scoring System," *Intelligent Systems, IEEE* , vol.25, no.5, pp.61-67, Sept.-Oct. 2010
- CHI, M., VANLEHN, K., LITMAN, D., AND JORDAN, P. 2010. Inducing Effective Pedagogical Strategies Using Learning Context Features. In: *Proc. of the 18th Int. Conference on User Modeling*.
- CONATI, C. AND MACLAREN, H. 2009. Empirically building and evaluating a probabilistic model of user affect. *User Modeling and User-Adapted Interaction*, August, 2009 267-303.
- CRAIG, S. D., D'MELLO,S., WITHERSPOON, A. AND GRAESSER, A. 2008. 'Emote aloud during learning with AutoTutor: Applying the Facial Action Coding System to cognitive-affective states during learning', *Cognition & Emotion*, 22: 5, 777 — 788.
- D'MELLO, S. K., CRAIG, S. D., WITHERSPOON, A., MCDANIEL, B., AND GRAESSER, A. 2008. Automatic detection of learner's affect from conversational cues. *User Modeling and User-Adapted Interaction* 18, 1-2 (Feb. 2008), 45-80.
- FORBES-RILEY, K., AND LITMAN, D. 2010. Metacognition and Learning in Spoken Dialogue Computer Tutoring. *Proceedings 10th International Conference on Intelligent Tutoring Systems (ITS)*, Pittsburgh, PA.
- FORBES-RILEY, K., ROTARU, M., AND LITMAN, J. 2009. The Relative Impact of Student Affect on Performance Models in a Spoken Dialogue Tutoring System. *User Modeling and User-Adapted Interaction (Special Issue on Affective Modeling and Adaptation)*, 18(1-2), February, 11-43.
- KLEIN, D AND MANNING, C. 2003. Accurate Unlexicalized Parsing. *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, pp. 423-430.
- LISCOMBE, J., HIRSCHBERG, J., AND VENDITTI, J. 2005. Detecting Certainty in Spoken Tutorial Dialogues. In *Proceedings of Interspeech 2005—Eurospeech*, Lisbon, Portugal.
- LITMAN, D., MOORE, J., DZIKOVSKA, M., AND FARROW. E. 2009. Using Natural Language Processing to Analyze Tutorial Dialogue Corpora Across Domains and Modalities. *Proceedings 14th International Conference on Artificial Intelligence in Education (AIED)*, Brighton, UK, July.
- LITMAN, D., AND FORBES-RILEY, K. 2009. Spoken Tutorial Dialogue and the Feeling of Another's Knowing. *Proceedings 10th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, London, UK, September.
- RUS, V., LINTEAN, M.,, AND AZEVEDO, R.. 2009. Automatic Detection of Student Mental Models During Prior Knowledge Activation in MetaTutor. In *Proceedings of the 2nd International Conference on Educational Data Mining* (Jul. 1-3, 2009). Pages 161-170
- SHAH ,J., VARGAS-HERNANDEZ, N., SMITH, S.M. (2003) Metrics for measuring ideation effectiveness. *Design Studies* 24: 111-134
- SONG, S., AGOGINO, A.M. 2004 Insights on Designers' Sketching Activities in Product Design Teams. 2004

Using Logistic Regression to Trace Multiple Subskills in a Dynamic Bayes Net

YANBO XU and JACK MOSTOW
Carnegie Mellon University, United States

A challenge in estimating students' changing knowledge from sequential observations of their performance arises when each observed step involves multiple subskills. To overcome this mismatch in grain size between modelled skills and observed actions, we use logistic regression over each step's subskills in a dynamic Bayes net (LR-DBN) to model transition probabilities for the overall knowledge required by the step. Unlike previous methods, LR-DBN can trace knowledge of the individual subskills without assuming they are independent. We evaluate how well it fits children's oral reading fluency data logged by Project LISTEN's Reading Tutor, compared to other methods.

Key terms: dynamic Bayes net, logistic regression, knowledge tracing, multiple subskills, oral reading fluency

1. INTRODUCTION

Dynamic Bayes nets are often used to model skill acquisition, e.g., in Knowledge Tracing (KT) [Corbett and Anderson, 1995]. They model a skill as a hidden state of knowledge, and estimate the changing probability of this state by observing successive attempts to use the skill. However, KT does not model multiple subskills used in such an attempt.

Previous research has explored various approaches to this problem. One approach is to use a conjunctive model [Cen et al., 2008; Gong et al., 2010], which assumes that a student must master all of the subskills in order to perform the step correctly. If the subskills are independent, the probability of knowing them all can be estimated by multiplying the estimated probabilities of knowing the individual subskills. However, this product typically underestimates the probability of knowing all the subskills. The minimum of their estimated probability provides a less pessimistic estimate based on the assumption that the likelihood of a correct answer is dominated by the student's weakest subskill [Gong et al., 2010]. Alternatively, Koedinger et al. [2011] use techniques from Bayesian nets to avoid blaming each subskill equally in conjunctive models. By pre-specifying the relationship between a step and its subskills, all of these approaches make strong assumptions about independence of subskills. Performance Factors Analysis [Pavlik Jr. et al., 2009b] and Learning Factors Analysis [Cen et al., 2006; Pavlik Jr. et al., 2009a] use non-linear regression to estimate multiple subskills without making such assumptions, but statically: that is, they do not trace subskills over time.

This paper presents LR-DBN, a method that uses logistic regression to trace multiple subskills in a dynamic Bayes net student model without assuming they are independent. Section 2 explains how LR-DBN works, Section 3 evaluates it, and Section 4 concludes.

2. LOGISTIC REGRESSION IN A DYNAMIC BAYES NET

In a KT model, and in other dynamic Bayes net models for student modelling [Chang et al., 2006], we estimate the probability of a student knowing, learning, or forgetting the skill(s) required to perform each observed step. So, we use a latent variable to model a hidden knowledge state that changes over time, and infer it from sequential observations of the student's performance. If we know (or assume) which set of subskills a step

Authors' addresses: Y. Xu, E-mail: yanbox@cs.cmu.edu; J. Mostow, E-mail: mostow@cs.cmu.edu, Project LISTEN, RI-NSH 4103, 5000 Forbes Avenue, Carnegie Mellon University, Pittsburgh, PA 15213-3890, USA. 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.

requires, it makes sense to estimate overall knowledge of the step as some function of the estimated knowledge of each individual subskill it requires.

Accordingly, we propose to model the probabilities of transitions between successive knowledge states using logistic regression over all of the subskills. We later prove that this approach is equivalent to modeling the knowledge probabilities themselves using logistic regression, but it provides a more convenient way to trace the individual subskills.

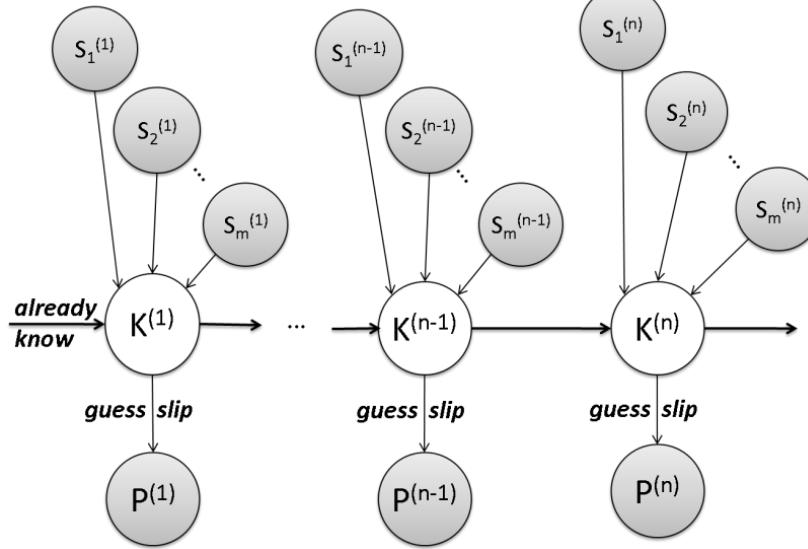


Fig. 1. A dynamic Bayes Architecture with Logistic Regression

Fig. 1 shows a dynamic Bayes architecture of KT framework model with binary variables:

- $S_j^{(n)}$: known indicator variable; 1 if step n requires subskill j , 0 otherwise.
- $K^{(n)}$: hidden; true iff the student has the knowledge step n requires.
- $P^{(n)}$: observed; true iff the student performs step n correctly.

Besides, we denote *already know* at initial state as $P(K^{(0)} = \text{true})$. Then we use logistic regression to model the $(1 - \text{already know})$ and transition probabilities from the knowledge state $K^{(n-1)}$ at step $n-1$ to the knowledge state $K^{(n)}$ at step n over m subskills:

$$1 - \text{already know} = P(K^{(0)} = \text{false}) = \frac{\exp(-\sum_{j=1}^m \beta_j^{(0)} S_j^{(0)})}{1 + \exp(-\sum_{j=1}^m \beta_j^{(0)} S_j^{(0)})} \quad (1)$$

$$P(K^{(n)} = \text{false} \mid K^{(n-1)} = \text{false}) = \frac{\exp(-\sum_{j=1}^m \beta_j S_j^{(n)})}{1 + \exp(-\sum_{j=1}^m \beta_j S_j^{(n)})} \quad (2)$$

$$P(K^{(n)} = \text{false} \mid K^{(n-1)} = \text{true}) = \frac{\exp(-\sum_{j=1}^m \gamma_j S_j^{(n)})}{1 + \exp(-\sum_{j=1}^m \gamma_j S_j^{(n)})} \quad (3)$$

Here $\beta_j^{(0)}$ is the coefficient fit for skill j at the initial state, where $S_j^{(0)} = 1$, and β_j and γ_j represent skill j 's respective contributions (when involved) to the transition probabilities in (2) and (3) from $K^{(n-1)}$ to $K^{(n)}$. We assume that β_j and γ_j do not vary over time. These equations imply that the more subskills a step requires, or the harder they are to learn, the lower the probability of knowing or learning the step. We now show how the model can trace individual subskills, first in a simple case and then in the general case.

Tracing subskills in a simple case: Consider a simple scenario in which a student repeatedly practices a single step that involves multiple subskills. The transition probabilities in equations (1), (2), and (3) correspond to $(1 - \text{already know})$, $(1 - \text{learn})$, and forget in KT. For now we assume forget is 0, a standard assumption in KT. Thus:

$$\text{already know} = P(K^{(0)} = \text{true}) = 1 - P(K^{(0)} = \text{false}) = \frac{1}{1 + \exp(-\sum_{j=1}^m \beta_j^{(0)} S_j)} \quad (4)$$

$$\begin{aligned} \text{learn} &= P(K^{(n)} = \text{true} \mid K^{(n-1)} = \text{false}) = 1 - P(K^{(n)} = \text{false} \mid K^{(n-1)} = \text{false}) \\ &= \frac{1}{1 + \exp(-\sum_{j=1}^m \beta_j S_j)} \end{aligned} \quad (5)$$

$$\text{forget} = P(K^{(n)} = \text{false} \mid K^{(n-1)} = \text{true}) = 0 \quad (6)$$

Note that the variables S_j 's that indicate which subskills are used in a step do not have superscripts of n because their assignments are determined by the step, so they remain constant for repeated practice of the same step. Now we compute $P(K^{(1)} = \text{true})$:

$$\begin{aligned} P(K^{(1)} = \text{true}) &= 1 - P(K^{(1)} = \text{false}) \\ &= 1 - [P(K^{(1)} = \text{false} \mid K^{(0)} = \text{true}) \times P(K^{(0)} = \text{true})] \quad (\text{By Bayes rule}) \\ &\quad + P(K^{(1)} = \text{false} \mid K^{(0)} = \text{false}) \times P(K^{(0)} = \text{false})] \\ &= 1 - 0 - \frac{\exp(-\sum_{j=1}^m \beta_j S_j)}{1 + \exp(-\sum_{j=1}^m \beta_j S_j)} \times \frac{\exp(-\sum_{j=1}^m \beta_j^{(0)} S_j)}{1 + \exp(-\sum_{j=1}^m \beta_j^{(0)} S_j)} \quad (\text{By eq. 6, 4, and 5}) \\ &= 1 - \frac{\exp(-\sum_{j=1}^m (\beta_j^{(0)} + \beta_j) S_j)}{1 + \exp(-\sum_{j=1}^m \beta_j S_j) + \exp(-\sum_{j=1}^m \beta_j^{(0)} S_j) + \exp(-\sum_{j=1}^m (\beta_j^{(0)} + \beta_j) S_j)} \end{aligned} \quad (7)$$

Define $C = \exp(-\sum_{j=1}^m \beta_j S_j) + \exp(-\sum_{j=1}^m \beta_j^{(0)} S_j)$. We can always find some $\delta_j^{(1)}$'s such that $\frac{1}{1+C} = \exp(-\sum_{j=1}^m \delta_j^{(1)} S_j)$, e.g., by choosing $\delta_j^{(1)}$'s to be equal, based on the assumption that the subskills change equally. Then:

$$\begin{aligned} P(K^{(1)} = \text{true}) &= 1 - \frac{\exp(-\sum_{j=1}^m (\beta_j^{(0)} + \beta_j) S_j)}{1 + C + \exp(-\sum_{j=1}^m (\beta_j^{(0)} + \beta_j) S_j)} \\ &= 1 - \frac{\frac{1}{1+C} \exp(-\sum_{j=1}^m (\beta_j^{(0)} + \beta_j) S_j)}{1 + \frac{1}{1+C} \exp(-\sum_{j=1}^m (\beta_j^{(0)} + \beta_j) S_j)} \\ &= 1 - \frac{\exp(-\sum_{j=1}^m (\beta_j^{(0)} + \beta_j + \delta_j^{(1)}) S_j)}{1 + \exp(-\sum_{j=1}^m (\beta_j^{(0)} + \beta_j + \delta_j^{(1)}) S_j)} \\ &= \frac{1}{1 + \exp(-\sum_{j=1}^m (\beta_j^{(0)} + \beta_j + \delta_j^{(1)}) S_j)} \\ &= \frac{1}{1 + \exp(-\sum_{j=1}^m (\beta_j^{(0)} + \Delta\beta_j^{(1)}) S_j)} \end{aligned} \quad (8)$$

Here we define $\Delta\beta_j^{(1)} = \beta_j + \delta_j^{(1)}$. More generally, at step n we have:

$$P(K^{(n)} = \text{true}) = \frac{1}{1 + \exp(-\sum_{j=1}^m (\beta_j^{(0)} + \Delta\beta_j^{(1)} + \dots + \Delta\beta_j^{(n)}) S_j)} \quad (9)$$

Therefore, we can use the proposed model to trace subskill j to step n by using $\Delta\beta_j^{(1)}, \dots, \Delta\beta_j^{(n)}$. Meanwhile, we also showed that the probability of having all the knowledge for step n is a logistic regression over coefficients $\beta_j^{(0)} + \Delta\beta_j^{(1)} + \dots + \Delta\beta_j^{(n)}$ for each subskill $j = 1, \dots, m$.

Tracing subskills in the general case: What if steps require different sets of subskills?

$$\begin{aligned} P(K^{(1)} = \text{true}) &= P(K^{(1)} = \text{true} \mid K^{(0)} = \text{true}) \times P(K^{(0)} = \text{true}) \\ &\quad + P(K^{(1)} = \text{true} \mid K^{(0)} = \text{false}) \times P(K^{(0)} = \text{false}) \quad (\text{By Bayes rule}) \\ &= \frac{1}{1 + \exp(-\sum_{j=1}^m \gamma_j S_j^{(1)})} \frac{1}{1 + \exp(-\sum_{j=1}^m \beta_j^{(0)} S_j^{(0)})} \quad (\text{By eq. 1, 2 and 3}) \end{aligned}$$

$$+ \frac{1}{1+\exp(-\sum_{j=1}^m \beta_j S_j^{(1)})} \frac{\exp(-\sum_{j=1}^m \beta_j^{(0)} S_j^{(0)})}{1+\exp(-\sum_{j=1}^m \beta_j^{(0)} S_j^{(0)})} \quad (10)$$

(Note that *forget* is no longer 0 in the general case.)

Using the same trick as for (8), let $C' = \exp(-\sum_{j=1}^m \beta_j S_j^{(1)}) (1 + \exp(-\sum_{j=1}^m \beta_j^{(0)} S_j^{(0)}))$, and let $C'' = \exp(\sum_{j=1}^m \beta_j^{(0)} S_j^{(0)}) (1 + \exp(\sum_{j=1}^m \beta_j S_j^{(1)})) + \exp(\sum_{j=1}^m \gamma_j S_j^{(1)})$. We choose $\delta_j^{(1)}$ and $\varepsilon_j^{(1)}$ such that $\frac{1}{1+C'} = \exp(-\sum_{j=1}^m \delta_j^{(1)} S_j^{(0)} \bar{S}_j^{(1)})$ and $1+C'' = \exp(-\sum_{j=1}^m \varepsilon_j^{(1)} \bar{S}_j^{(0)} S_j^{(1)})$, where $\bar{S} = 1 - S$. We then rewrite (10) as:

$$P(K^{(1)} = \text{true}) = \frac{1}{1+\exp(-\sum_{j=1}^m (\beta_j^{(0)} S_j^{(0)} + (\beta_j + \gamma_j) S_j^{(0)} S_j^{(1)} + \Delta\beta_j^{(1)} S_j^{(0)} \bar{S}_j^{(1)} + \Delta\gamma_j^{(1)} S_j^{(0)} S_j^{(1)}))} \quad (11)$$

Here $\Delta\beta_j^{(1)} = \beta_j + \gamma_j + \delta_j^{(1)}$ and $\Delta\gamma_j^{(1)} = \beta_j + \gamma_j + \varepsilon_j^{(1)}$. More generally, at step n :

$$P(K^{(n)} = \text{true}) = \frac{1}{1+\exp(-\sum_{j=1}^m (\beta_j^{(0)} S_j^{(0)} + \sum_{k=1}^n (\beta_j + \gamma_j) S_j^{(k-1)} S_j^{(k)} + \Delta\beta_j^{(k)} S_j^{(k-1)} \bar{S}_j^{(k)} + \Delta\gamma_j^{(k)} \bar{S}_j^{(k-1)} S_j^{(k)}))} \quad (12)$$

Thus we can trace subskill j at step n by using $\Delta\beta_j^{(n)}$ if $S_j^{(n-1)} = 0$ and $S_j^{(n)} = 1$, $\Delta\gamma_j^{(n)}$ if $S_j^{(n-1)} = 1$ and $S_j^{(n)} = 0$, and $\beta_j + \gamma_j$ if $S_j^{(n-1)} = 1$ and $S_j^{(n)} = 1$. To save space, we only showed here how to update subskill knowledge independent of observed performance on the step. To condition on performance, we can further derive $P(K^{(2)} = \text{true} | P^{(1)})$, and therefore derive $P(K^{(n)} = \text{true} | P^{(1)}, \dots, P^{(n-1)})$ in (9) and (12).

3. EXPERIMENTS

We implemented LR-DBN in the Bayes Net Toolbox for Matlab [Murphy, 2006]. Specifically, we defined the knowledge K given m subskills S_j as a “softmax” node in the toolbox. We used LR-DBN to model the growth of children’s oral reading fluency, where performance P denotes whether the student read a word fluently. Our data was recorded by Project LISTEN’s Reading Tutor [Mostow and Aist, 2001] during the 2005-2006 school year. We scored each word as fluent if read without help or hesitation and accepted by the automated speech recognizer.

We assume that whether a student read a word fluently depended on whether the student knew the grapheme-to-phoneme mappings in the word. So in our experiment, the subskills required in a student’s reading word step are the word’s grapheme-to-phoneme mappings. We modeled 27 children who read a total of 5,078 distinct word types with 332 unique grapheme-phoneme mappings. To evaluate our models, we fit them separately for each student on the first half of all of that student’s data, tested on the second half, and averaged the test results across students. The test set contains a total of 32,122 read words, out of which 23,222 were fluent. For comparison, we also applied the original KT model and estimated the probability of knowing a word as the minimum probability of knowing all of its grapheme-to-phoneme mappings, based on assuming that the student’s weakest subskill determined whether he read the word fluently.

Table I shows the results. The values in parentheses show 95% confidence intervals based on standard error calculated from the unbiased weighted sample variance of individual students’ accuracies. Since the data is unbalanced (72.3% of the words were fluent), we also report within-class accuracies. Table I shows that LR-DBN significantly outperformed the weakest-subskill KT model, especially on non-fluent words. High within-class accuracy on unbalanced data is often hard for KT [Zhang et al., 2008].

Table I. LR-DBN vs. KT Models of Children's Reading Fluency Growth

	Accuracy	Accuracy on fluent words	Accuracy on non-fluent words
LR-DBN	88.8% ($\pm 1.7\%$)	92.0% ($\pm 2.7\%$)	80.5% ($\pm 9.5\%$)
KT of weakest skill	72.6% ($\pm 3.6\%$)	94.5% ($\pm 5.8\%$)	19.3% ($\pm 8.4\%$)

4. CONCLUSION

This paper describes and evaluates LR-DBN, a novel student modeling method to trace hidden subskills by using logistic regression within a dynamic Bayes net. We used oral reading data from 27 children to compare LR-DBN to conventional knowledge tracing of weakest subskills. LR-DBN performed significantly better overall, thanks to similar accuracy ($92.0\% \pm 2.7\%$ vs. $94.5\% \pm 5.8\%$) on fluent words combined with 4-fold higher accuracy on disfluent words ($80.5\% \pm 9.5\%$ vs. $19.3\% \pm 8.4\%$). We later [Xu and Mostow, 2011] tested both models on a published data set [Koedinger et al., 2010] from 123 students working on a geometry area unit of the Bridge to Algebra Cognitive Tutor®. LR-DBN fit this data significantly better too, with only half as many prediction errors on unseen data. Future tests should use data from more students and tasks, and compare LR-DBN to other baselines, such as conjunctive modeling [Koedinger et al., 2011].

REFERENCES

- CEN, H., KOEDINGER, K. and JUNKER, B. 2006. Learning Factors Analysis – A General Method for Cognitive Model Evaluation and Improvement. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, Jhongli, Taiwan, June 26-30, 2006, 164-175.
- CEN, H., KOEDINGER, K.R. and JUNKER, B. 2008. Comparing Two IRT Models for Conjunctive Skills. In *Ninth International Conference on Intelligent Tutoring Systems*, Montreal, 2008, 796-798.
- CHANG, K.-M., BECK, J.E., MOSTOW, J. and CORBETT, A. 2006. A Bayes Net Toolkit for Student Modeling in Intelligent Tutoring Systems. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, Jhongli, Taiwan, June 26-30, 2006, K. ASHLEY and M. IKEDA, Eds., 104-113.
- CORBETT, A. and ANDERSON, J. 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction* 4, 253-278.
- GONG, Y., BECK, J. and HEFFERNAN, N.T. 2010. Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting Procedures. In *Proceedings of the 10th International Conference on Intelligent Tutoring Systems*, Pittsburgh, PA, 2010, V. ALEVEN, J. KAY and J. MOSTOW, Eds. Springer Berlin / Heidelberg, 35-44.
- KOEDINGER, K., PAVLIK, P.I., STAMPER, J., NIXON, T. and RITTER, S. 2011. Fair Blame Assignment in Student Modeling. In *Fourth International Conference on Educational Data Mining*, Eindhoven, NL, July 6-8, 2011.
- KOEDINGER, K.R., BAKER, R.S.J.D., CUNNINGHAM, K., SKOGSHOLM, A., LEBER, B. and STAMPER, J. 2010. A Data Repository for the EDM community: The PSLC DataShop. In *Handbook of Educational Data Mining*, C. ROMERO, S. VENTURA, M. PECHENIZKIY and R.S.J.D. BAKER, Eds. CRC Press, Boca Raton, FL, 43-55.
- MOSTOW, J. and AIST, G. 2001. Evaluating tutors that listen: An overview of Project LISTEN. In *Smart Machines in Education*, K. FORBUS and P. FELTOVICH, Eds. MIT/AAAI Press, Menlo Park, CA, 169-234.
- MURPHY, K. 2006. Bayes Net Toolbox for Matlab.
- PAVLIK JR., P.I., CEN, H. and KOEDINGER, K.R. 2009a. Learning factors transfer analysis: Using learning curve analysis to automatically generate domain models. In *Proceedings of the 2nd International Conference on Educational Data Mining*, Cordoba, Spain, 2009a, T. BARNES, M. DESMARAIS, C. ROMERO and S. VENTURA, Eds., 121-130.
- PAVLIK JR., P.I., CEN, H. and KOEDINGER, K.R. 2009b. Performance Factors Analysis - A New Alternative to Knowledge Tracing. In *Proceedings of the 14th International Conference on Artificial Intelligence in Education (AIED09)*, 2009b, 531-538.
- XU, Y. and MOSTOW, J. 2011. Logistic Regression in a Dynamic Bayes Net Models Multiple Subskills Better! In *4th International Conference on Educational Data Mining* Eindhoven, Netherlands, July 6-8, 2011.
- ZHANG, X., MOSTOW, J. and BECK, J.E. 2008. A Case Study Empirical Comparison of Three Methods to Evaluate Tutorial Behaviors. In *9th International Conference on Intelligent Tutoring Systems*, Montreal, June 23-27, 2008, B. WOOLF, Ed., Springer-Verlag, 122-131.

Monitoring Learners' Proficiency: Weight Adaptation in the Elo Rating System

K. WAUTERS

Katholieke Universiteit Leuven, Belgium

P. DESMET

Katholieke Universiteit Leuven, Belgium

AND

W. VAN DEN NOORTGATE

Katholieke Universiteit Leuven, Belgium

Adaptive item sequencing is a well-established adaptation technique for personalizing learning environments and can be achieved through an intense reciprocity between the item difficulty level and the learner's proficiency. Consequently, the need to monitor learners' proficiency level is of great importance. On that account, researchers have brought forward the Elo rating system. While the Elo rating system has its origin in chess, it has proven its value during its short history in the educational setting. Elo's algorithm implies that the rating after an event is function of the pre-event rating, the weight given to the new observation and the difference between the new observed score and the expected score. It seems reasonable to adapt the weight of Elo's algorithm as function of the number of observations: the more previous observations we have, the more certain we are about the learner's proficiency estimate, and the less this estimate should be affected by a new observation. The aim of this paper is to search for weights as a function of the number of previous observations that results in optimally accurate proficiency estimates, making use of a real data set. Results indicate that the Elo algorithm with a logistic weight function better approximates the parameter estimates obtained with the item response theory than the Elo algorithm with a fixed weight.

Key Words and Phrases: IRT, proficiency and Elo rating

1. INTRODUCTION

The research on e-learning environments has long been focused on delivering information online without taking into account the characteristics of the particular learner, course material and/or the context. It is only recently that research attention is drawn to dynamic or adaptive e-learning environments. An adaptive learning environment creates a personalized learning opportunity by incorporating one or more adaptation techniques to meet the learners' needs and preferences (Brusilovsky 1999). One of those adaptation techniques is adaptive item sequencing, in which the sequencing of the learning material is adapted to learner-, item-, and/or context characteristics (Wauters, Desmet & Van den Noortgate 2010). Hence, adaptive item sequencing can be established by matching the difficulty of the item to the proficiency level of the learner. Recently, the interest in adaptive item sequencing has grown, as it is found that excessively difficult items can frustrate learners, while excessively easy items can cause learners to lack any sense of challenge (e.g. Pérez-Marín, Alfonseca & Rodriguez 2006, Leung & Li 2007). Learners prefer learning environments where the item selection procedure is adapted to their proficiency. This is already accomplished to a certain extent in computerized adaptive tests (CATs; Wainer 2000).

A prerequisite for efficient adaptive item sequencing is to be able to estimate the learner's proficiency level at an early stage of the learning process and follow the learning curve adequately. Hence, the problem of the proficiency estimation is twofold (Wauters et al. 2010). On the one hand, we need to estimate the learner's proficiency level when little information is provided, which is referred to as the cold start problem (Masthoff 2004). On the other hand, we

Authors' addresses: K. Wauters, ITEC/IBBT, Faculty of Psychology and Educational Sciences, Katholieke Universiteit Leuven, Kortrijk, Belgium. E-mail: kelly.wauters@kuleuven-kortrijk.be; P. Desmet, ITEC/IBBT, Faculty of Arts, Katholieke Universiteit Leuven, Kortrijk, Belgium. E-mail: pietdesmet@kuleuven-kortrijk.be; W. Van den Noortgate, ITEC/IBBT, Faculty of Psychology and Educational Sciences, Katholieke Universiteit Leuven, Kortrijk, Belgium. E-mail: wim.vandennoortgate@kuleuven-kortrijk.be.

need to be able to follow the progression of the learner as learners are presumed to learn while they are working in the learning environment. As most learning environments are created to learn more than one skill, Bayesian networks are the dominant method of student modeling (Corbett & Anderson 1995). However, when no relationship exists between the skills to be learned, other methods can be applied. One method to model the learner's proficiency progress is by assessing while working in the learning environment, which can be done by means of progress testing. In progress testing, tests are frequently administered to allow for a quick intervention when atypical growth patterns are observed. In order to estimate the proficiency level of the learner more precisely, these tests can be made adaptive (Wainer 2000). Such CATs often make use of the item response theory (IRT; Van der Linden & Hambleton 1997) to estimate the proficiency level of the person, based on the results of prior calibration in which the item difficulty parameters are estimated. Hence, progress testing on the basis of CAT and IRT has the advantage that it makes more precise estimation of the person's proficiency level possible, but it requires costly prior calibration and it is computationally demanding. Furthermore, in order to be able to recover the atypical growth pattern, the proficiency assessment not only needs to be long enough to be accurate, but should also occur on a regular basis. Hence, progress testing is less appropriate for learning environments as it interrupts the learning process. Another approach, one that is well suited for learning environments, is the tracking of the learner's proficiency level. This can be achieved by updating the proficiency level of the learner after each item administration. One method that allows the tracking of the learner's proficiency level is the Elo rating system (Elo 1978). The Elo rating system was originally developed for rating chess performances and is recently implemented in the educational field (Brinkhuis & Maris 2010). When applied in chess, a chess player competes with his opponent, which results in a win, a loss or a draw. This data is known as paired comparison data, for which the Elo rating system was developed. In the educational field, we have a similar kind of paired comparison, where the learner is seen as a player and the item is seen as its opponent. In the formula of Brinkhuis and Maris (2010), the new proficiency rating after an item administration is function of the pre-administration rating, a weight given to the new observation and the difference between the actual score on the new observation and the expected score on the new observation. This expected score is calculated by means of the Rasch model. This formula implies that when the difference between the expected score and the observed score is high, the change in the estimate of the proficiency level is high. This algorithm enables continuous measurement, since the rating is updated after every event. This implies that the Elo rating system is order-sensitive. The formula for updating the proficiency level (the item difficulty is updated at the same time and in a similar way), is given by:

$$\theta_n = \theta_0 + W(X - X_e)$$

where θ_n is the new learner's proficiency level rating after the learner has answered an item, θ_0 is the pre-event rating, W is the weight given to the new observation, X is the actual observation (score 1 for correct, 0 for incorrect), and X_e is the expected observation which is estimated based on the Rasch model. Hence, the formula for updating the learner's proficiency level after a response ($X=0$ or 1) becomes:

$$\theta_n = \theta_0 + W \left[X - \frac{\exp(\theta_0 - \beta_0)}{1 + \exp(\theta_0 - \beta_0)} \right],$$

where β_0 is the estimated item difficulty level before that item is answered by this specific person. Because the estimation of the proficiency level becomes more stable when more items are answered by this person, the weight given to the new observation should decrease when the rating of the learner's proficiency level is based on more observations. In this study we will focus on this weight adaptation by searching for a function that can describe the decrease in weight. More specifically, the aim of this paper is to compare the proficiency levels obtained by means of IRT estimation with those estimated on the basis of the Elo rating system with various weight functions. The focus is not on tracking the learner's proficiency level, but on promptly estimating the learner's proficiency level. In future research, the focus will go to monitoring the learner's proficiency level.

2. EXPERIMENT

2.1 Learners' proficiency Estimation Methods

2.1.1 Item Response Theory. To estimate the learners' proficiency level, the IRT model with a single item parameter proposed by Rasch (Van der Linden & Hambleton 1997) is used. The Rasch model models the probability of answering an item correctly as a logistic function of the difference between the person's proficiency level (θ) and the item difficulty level (β):

$$\text{Prob}(X_{pi} = 1) = \frac{\exp(\theta_p - \beta_i)}{1 + \exp(\theta_p - \beta_i)}$$

In this study, the proficiency level of the learners is estimated with known item difficulty levels. The difficulty level of the items was estimated in advance by Selor, the selection agency of the Belgian government. Selor used examinee data to estimate the item difficulty parameters by means of IRT.

2.1.2 Elo Rating. In this study, the Elo rating systems implemented by Brinkhuis and Maris (2010) was used to estimate learners' proficiency level.

$$\theta_n = \theta_0 + W \left[X - \frac{\exp(\theta_0 - \beta_0)}{1 + \exp(\theta_0 - \beta_0)} \right]$$

The weight function should have two crucial characteristics: (1) it should slowly decrease when the number of items answered by that learner increases; (2) it should never become zero or negative. A function that satisfies these requirements is the logistic function. Hence, we propose a weight function of the logistic family in which three parameters will be varied.

$$W = \frac{W_0}{1 + a * \exp(b * N_{ip})}$$

where N_{ip} is the number of items answered by learner p before answering item i . The three parameters that will be varied in this equation are W_0 , which is the starting weight, a , and b . The a parameter influences the slope of the logistic curve (figure 1a), the b -parameter influences at what point the weight is reduced from W_0 to $W_0/2$ (figure 1b). A b -parameter set to 50 indicates that the weight given to the new observation is halved by the time the fiftieth item is presented.

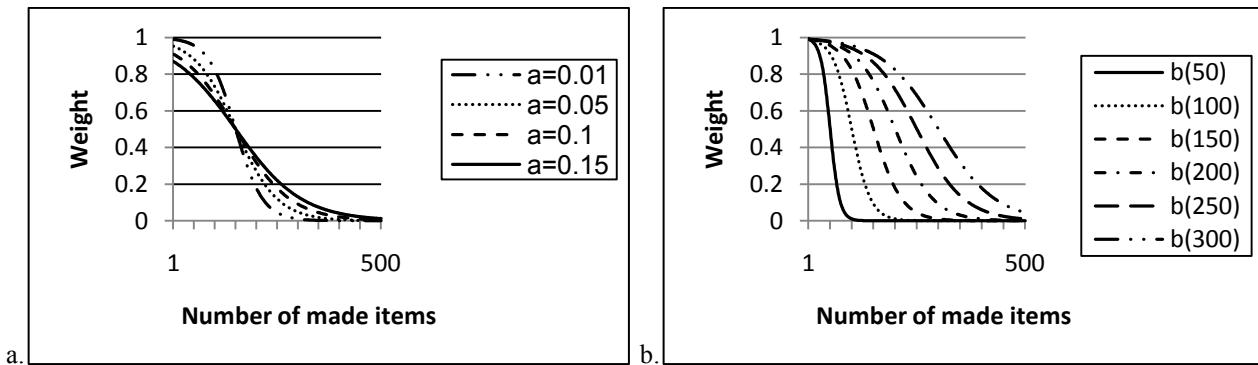


Fig. 1. Logistic weight function with $W_0=1$, $b=150$ (i.e. the weight is halved by the time the 150th item is presented) and several a -values (1a); Logistic weight function with $W_0=1$, $a=.01$ and several b -values (1b).

We hypothesize that the Elo rating formula will be more efficient when the starting weight is high, because when little information is known about the learner's proficiency level all the information needs to be extracted from the new observations. Furthermore, we hypothesize that the weight should decrease as the number of prior observations increases. When the estimation of the learner's proficiency level is based on many observations, the estimate is fairly reliable and the new observation should not give much weight to the updating of the proficiency level.

2.2 Method

2.2.1 Participants. Students from ten educational programs in the Flemish part of Belgium (1st and 2nd Bachelor Linguistics and Literature – K.U.Leuven; 1st, 2nd and 3rd Bachelor Teacher-Training for primary education – Katho Tielt; 1st and 2nd Bachelor Teacher-Training for secondary education – Katho Reno; 1st and 2nd Bachelor of Applied Linguistics – HUB and Lessius; and 1st Bachelor Educational Science – K.U.Leuven) were contacted to participate in the experiment. Two hundred thirty two students completed the whole experiment.

2.2.1 Material and Procedure. The study consisted of two sessions, each taking approximately half an hour. The learning material consisted of items on French verb conjugation, supposedly measuring one single skill. The instructions, comprising information on the login procedure for the learning environment and on the proceedings of the experimental study were sent to the participants by email. When students were logged on, they were given an informed consent. Subsequently, they completed 25 items. After two weeks, students completed the next 25 items. The difficulty of both tests of 25 items can be considered equal as the composing items are of equal difficulty level. IRT estimation and the Elo rating system both combine the answers of the learner on the two tests, resulting in 50 items.

2.3 Results

Learners score significantly higher on the posttest ($M=66.64$, $SD=17.66$) compared to the pretest ($M=63.86$, $SD=18.15$), $t(231)=-3.09$, $p<.001$. For each of the two tests, a logistic regression analysis with the raw score as dependent variable and the item order and item difficulty parameter as independent variables is performed to assess the within-test gains. Difficulty significantly predicted the odds of a correct response ($\beta=-0.78$, $t(1)=705.29$, $p<.001$ for the pretest, and $\beta=-0.67$, $t(1)=541.57$, $p<.001$ for the posttest). However, no significant effect was found of item order, indicating no within-test learning gains.

The Pearson correlation between the estimated learners' proficiency level on the basis of IRT and the estimated learners' proficiency level on the basis of the Elo rating system was the criterion used to evaluate the efficacy of the logistic weight function. Detailed correlation results for the learners' proficiency level estimates are shown in table I.

Table I. Pearson correlation matrix of the learners' proficiency level estimates for the different parameter values of the logistic weight function included in the Elo rating system.

a	b	W_0				
		1	0.8	0.6	0.4	0.2
0	0	.79	.82	.85	.88	.92
0.01	50	.85	.87	.89	.91	.94
0.01	100	.80	.83	.86	.89	.92
0.01	150	.80	.82	.85	.89	.92
0.01	200	.80	.82	.85	.88	.92
0.01	250	.79	.82	.85	.88	.92
0.01	300	.79	.82	.85	.88	.92
0.05	50	.86	.88	.90	.92	.94
0.05	100	.81	.84	.87	.90	.93
0.05	150	.81	.83	.86	.89	.93
0.05	200	.80	.83	.86	.89	.92
0.05	250	.80	.83	.86	.89	.92
0.05	300	.80	.83	.85	.89	.92
0.10	50	.86	.88	.90	.92	.94
0.10	100	.82	.85	.87	.90	.93
0.10	150	.81	.84	.87	.90	.93
0.10	200	.81	.84	.86	.89	.93
0.10	250	.81	.83	.86	.89	.93
0.10	300	.81	.83	.86	.89	.93
0.15	50	.86	.88	.90	.92	.94
0.15	100	.83	.85	.88	.90	.93
0.15	150	.82	.84	.87	.90	.93
0.15	200	.82	.84	.87	.90	.93
0.15	250	.82	.84	.87	.90	.93
0.15	300	.81	.84	.86	.89	.93

Note: all correlations are statistically significant at the .001 significance level.

The Pearson correlation coefficient between IRT-based proficiency estimates and the proficiency estimates obtained by means of the Elo rating system shows that the Elo rating system with an initial weight of 0.2 (last column) outperformed the Elo rating system with an initial weight of 0.4 ($t(229)=-10.93$, $p<.001$) or higher. The results further indicate that the correlation between IRT-based proficiency estimates and the proficiency estimates obtained by means of the Elo rating system with a logistic weight function ($a>0$) is higher than the correlation between IRT-based proficiency estimates and the proficiency parameter estimates obtained through the Elo rating system with a fixed weight ($a=0$; for example, the difference between the correlation coefficients in case $W_0=0.2$ and $a=0$, and the correlation coefficient in case $W_0=0.2$, $a=0.01$ and the b-parameter is set so the weight is halved by the time the fiftieth item is presented to the learner, i.e. $b=50$ in table I is significant, $t(229)=-6.90$, $p<.001$). Furthermore, results indicate that the Pearson correlation coefficient is the highest when the a-parameter has a value above 0.01 and the b-parameter is set so the weight is halved by the time the fiftieth item is presented to the learner, i.e. $b=50$ in table I.

3. DISCUSSION

Monitoring the learner's progress is an important component in the creation of an adaptive learning environment by means of adaptive item sequencing. Because IRT is computationally demanding and the requirement of prior item difficulty estimation is costly, the Elo rating system has been put forward as an alternative by Brinkhuis and Maris (2010). The Elo rating system allows for dynamic parameter evaluation without prior item difficulty estimation. As it is already shown that the Elo rating system has its value in the educational field for tracking the learner's proficiency level, this article further builds on the Elo rating system and extends the formula of Brinkhuis and Maris (2010) by including a logistic weight function that describes the decrease in weight as the number of observations increases. Based on the response data of two hundred thirty two learners on fifty items, the proficiency level was estimated by means of IRT, the Elo rating system with a fixed weight value, and the Elo rating system with a logistic weight function in which three parameters were varied: (1) the initial weight, (2) the a-parameter, and (3) the b-parameter.

The findings indicate that, at least for data as the ones collected, the weight given to the new observation should be small at the beginning. This means that, even though the estimation of the learner's proficiency level is based on a small amount of observations and therefore unreliable, the outcome of a new observation only gives small weight to the update of the proficiency level. This result is inconsistent with our hypothesis and could be due to the simultaneous estimation of the item difficulty parameter and the learner's proficiency parameter in the Elo rating system. In this study, both the item difficulty level and the learner's proficiency level were estimated. Therefore, not only the estimation of the learner's proficiency level was unreliable at the beginning, but also the estimation of the item difficulty level and hence, of the expected outcome of the new observation. In such a situation, the small weight given to a new observation could be advocated.

Results also indicate that the weight should decrease as the number of observations increases. This finding can be deduced from the lower correlation of IRT with the Elo rating estimates obtained with a fixed weight compared to the correlation of IRT with the Elo rating estimates with a decreasing weight. This result is in line with our hypotheses arguing that as the estimation of the learner's proficiency level is based on more observations, the estimation becomes more stable and reliable. Furthermore, the results indicate that this decrease in weight should be steep as the best correlation with IRT-based estimation is found when the weight is halved by the time the fiftieth item is presented to the learner. This could also be explained by the simultaneous estimation of the item difficulty parameter and the learner's proficiency parameter in the Elo rating system. As the expected score on a new observation is unreliable due to the unreliable item difficulty estimate, the information provided by this new observation is small and little weight should be given to this new observation.

Even though this study gives an indication of the parameter values resulting in a good estimation of the learner's proficiency level, it should be considered that these results depend on this particular data set. We recognize that results might depend on the learning rate of the student, the skill difficulty level, the variance in item difficulty, etc.

Future research will review the learner's proficiency parameter estimates obtained on the basis of the Elo rating system with starting item difficulty parameter values estimated by means of prior IRT-based calibration. This should allow us to evaluate the logistic weight function within the Elo rating system for estimating the learner's proficiency level when items difficulty parameters, and hence the expected score on a new observation are more reliable. Another research question will address the relationship of the weight function with other variables, such as time between two learning sessions and hint usage.

REFERENCES

- CORBETT, A., AND ANDERSON, J. 1995. Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. *User Modeling and User-Adapted Interaction*, 4, 253-278.
- BRINKHUIS, M.J.S., AND MARIS, G. 2010. Adaptive Estimation: How to Hit a Moving Target. Report No.2010-1, Measurement and Research Department, Cito, Arnhem
- BRUSILOVSKY, P. 1999. Adaptive and Intelligent Technologies for Web-Based Education. *Künstliche Intelligenz*, 4, 19-25.
- ELO, A.E. 1978. *The Rating of Chess Players, Past and Present*, B.T. Batsford Ltd., London.
- LEUNG, E.W.C., AND LI, Q. 2007. An Experimental Study of a Personalized Learning Environment Through Open-Source Software Tools. *IEEE Transaction on Education*, 50, 331-337.
- MASTHOFF, J. 2004. Group modeling: Selecting a Sequence of Television Items to Suit a Group of Viewers. *User Modeling and User-Adapted Interaction*, 14, 37-85.
- PÉREZ-MARÍN, D., ALFONSECA, E., AND RODRIGUEZ, P. 2006. On the Dynamic Adaptation of Computer Assisted Assessment of Free-Text Answers. *Adaptive Hypermedia and Adaptive Web-Based Systems, Proceedings*, 4018, 374-377.
- VAN DER LINDEN, W.J., AND HAMBLETON, R.K. 1997. *Handbook of Modern Item Response Theory*, Springer, New York.
- WAINER, H. 200. *Computerized Adaptive Testing: a Primer*, Erlbaum, London.
- WAUTERS, K., DESMET, P., AND VAN DEN NOORTGATE, W. 2010. Adaptive Item-Based Learning Environments Based on the Item Response Theory: Possibilities and Challenges. *Journal of Computer Assisted Learning*, 26(6), 549-562.

Modeling students' activity in online discussion forums: a strategy based on time series and agglomerative hierarchical clustering

G. COBO, D. GARCÍA, E. SANTAMARÍA, J.A. MORÁN,
J. MELENCHÓN, C. MONZO

Universitat Oberta de Catalunya (UOC), Spain

Online discussion forums (or discussion boards) are one of the most common tools in web-based teaching-learning environments. Students' activity in discussion threads can be a relevant source of information that facilitates the monitoring tasks during the course by providing teachers with relevant indicators of their students' needs and lacks. In the present paper, the use of time series and an agglomerative hierarchical clustering algorithm is proposed with the aim of determining what different behavior patterns are adopted by students in online discussion forums. To this end, the actions carried out by students along the threads (e.g., writing and reading) are used to represent their activity in times series form. The use of an agglomerative hierarchical clustering algorithm is proposed in order to group similar students according to their activity profile. Some strategies on how to cut the obtained dendograms are discussed and preliminary experimental results are presented.

Key Words and Phrases: Online discussion forums, data mining, clustering educational data, agglomerative hierarchical clustering, modeling students' activity

1. INTRODUCTION

Online discussion forums (or discussion boards) are emerging as one of the most common tools in web-based teaching-learning environments. On the one hand, an online discussion forum, being an asynchronous tool, allows students to maintain discussions related with their learning processes at any time. On the other hand, it also allows teachers to develop monitoring and even assessment tasks. In fact, a high level of interaction among students is desirable and increases the effectiveness of distance education courses (Fulford & Zhang, 1993). Then, all the activity carried out in the discussions threads appears to be a relevant source of information that can provide teachers with relevant indicators of their students' needs and lacks.

The purpose of the present work is to propose a strategy in order to model students' activity in online discussion forums. In such a context, clustering students seems to be a proper way to find similar learning behaviors (Vellido *et al.*, 2009). Thus, the proposed strategy is based on grouping students according to their activity in the online discussion threads, in order to identify what similar behavior profiles can be found in the virtual classroom. Due to the number of profiles is a priori unknown (in fact, it can depend on several factors: total number of students, kind of teaching-learning strategies promoted by the teacher, kind of subject, etc.) an agglomerative hierarchical clustering algorithm is used in order to group the students and find the behavior profiles.

This paper is structured as follows. Firstly, the working framework is introduced in Section 2. Next, the proposal of a novel strategy to model students' activity in online discussion forums is addressed in Section 3. Preliminary experimental results are shown in Section 4. Finally, conclusions and further work are presented in Section 5.

Authors' addresses: G. Cobo, D. García, E. Santamaría, J.A. Morán, J. Melenchón and C. Monzo, Informatics, Multimedia and Telecommunications Department, Universitat Oberta de Catalunya (UOC), Barcelona, Spain. E-mails: {gcobo, dgarciaso, esantamaria, jmoranm, jmelenchonm, cmonzo}@uoc.edu

2. WORKING FRAMEWORK

The application of data mining techniques in web-based teaching-learning environments is an incipient area which can solve many problems or support teachers in their decisions (Romero *et al.*, 2006). Regarding student modeling field, different approaches are possible depending on the source of the analyzed data (logs, assessment and performance, tailored questionnaires, discussion threads, etc.), the kind of data mining techniques used in the modeling process (classification techniques, clustering algorithms, association rules, etc.) and the final application (predicting performance, identification of gifted students, implementing adaptive learning systems, etc.) (Romero *et al.*, 2010).

This paper is focused on modeling students' activity in online discussion forums. In this regard, relevant contributions can be found in literature. For instance, an analysis of different styles of posting is made in (Sackville and Sherratt, 2006) from both quantitative and qualitative perspectives. For such purpose, four different types of messages are considered: statement, limited response, questioning response and dialogue postings. In a similar way, a system that identifies discussion threads that may have unanswered questions and need instruction attention is built in (Ravi and Kim, 2007), where messages are classified in four different categories: questions, answers, elaborations and corrections. In more qualitative terms, a taxonomy of participation in online discussions based in two variables –interpersonal interaction and interaction with content– is proposed in (Bento, 2005).

Moreover, in terms of modeling students' behaviors in online asynchronous environments, (Beaudoin, 2002) deals with identification of lurkers (in an online discussion board, a lurker is the one who reads but never writes). In online teaching-learning environments, this kind of behavior makes impossible a visible and active interaction with other students and teacher. In order to investigate lurking, (Nonnecke and Preece, 2001) carried out a study on lurking using in-depth semi-structured interviews with members of online groups. The analysis reveals that lurking is a strategic activity involving more than just reading posts. Reasons for lurking are categorized and a model is proposed to explain lurker behavior. Worker, lurker and shirker students are identified in (Taylor, 2002), after finding three significant participation patterns –proactive, peripheral and parsimonious participation groups, respectively– in accessing and contributing to a discussion board.

In this scenario, a novel data mining-based strategy to model students' activity in online discussion forums is proposed in the next section of this paper.

3. MODELING STUDENT'S ACTIVITY IN ONLINE DISCUSSION FORUMS

Regardless of the different types of posting one can consider (Sackville and Sherratt, 2006) (Ravi and Kim, 2007), two main actions are carried out in an online discussion forum: writing and reading. Students can adopt different attitudes in terms of these two main actions, thus defining different behavior profiles (active learners, lurkers, etc.). Modeling students' activity in online forums consists on finding groups of students with similar behaviors in terms of the actions they carry out all along the discussion threads.

Then, on the one hand, the first question is: what representation of students' activity can properly reflect their behavior profile? One common approach would be just counting the amount of actions carried out by students (Taylor, 2002). Our proposal is to represent students' activity in times series form. Thereby, both amounts and tendencies in students' activity can be identified (e.g. it's not the same reading thirty posts in one single day than along one week). A time series that represents a single student's activity would be:

$$\mathbf{x}^{(i)} = \left(x_1^{(i)}, x_2^{(i)}, \dots, x_n^{(i)}, \dots, x_N^{(i)} \right)$$

where $\mathbf{x}^{(i)}$ is the time series that represents the i -th student, $x_n^{(i)}$ is the value $\mathbf{x}^{(i)}$ adopts at temporal sample n and N is the total number of temporal samples the time series includes (N weeks, N days, N hours...). In a virtual classroom with M students, several strategies can be carried out in order to define the value of $x_n^{(i)}$, e.g.:

$$x_n^{(i)} = \frac{rd_n^{(i)}}{rd^M}, \quad x_n^{(i)} = \frac{wr_n^{(i)}}{wr^M}, \quad x_n^{(i)} = \begin{cases} 1 & \text{if } rd_n^{(i)} \neq 0 \\ 0 & \text{else} \end{cases}, \quad x_n^{(i)} = \begin{cases} 1 & \text{if } wr_n^{(i)} \neq 0 \\ 0 & \text{else} \end{cases}$$

where $rd_n^{(i)}$ and $wr_n^{(i)}$ are respectively the number of reading and writing actions carried out by the i -th student at sample n and rd^M and wr^M are respectively the total amount of reading actions carried out by all M students (the whole virtual classroom).

And, on the other hand, another question arises: how to group students with similar behaviors (i.e. similar time series)? The result will be a certain number of clusters which contain learners with similar behavior profiles, but the number of clusters (i.e. the number of profiles) is a priori unknown. Besides, it seems logical setting up the groups by linking first the most similar couple of students, then linking the next most similar couple and so on. Given these premises, our proposal is to use an agglomerative hierarchical clustering algorithm, a well-used algorithm in educational data mining (Vellido *et al.*, 2009).

This kind of clustering algorithm presents significant advantages in this scenario. Firstly, the number of clusters is not required a priori, since the outcome of the algorithm is a hierarchical tree called dendrogram (Jain and Dubes, 1988) that shows how data joins along a tree structure of link points (nodes). Secondly, algorithm's behavior is completely deterministic, since requires no initialization. Thirdly, several labels for the data can be obtained by cutting the dendrogram according to different criteria. The most common one fits a certain distance threshold (the nodes under the threshold remain joined). Even so, criteria based on fitting inconsistency thresholds (the most consistent links remain joined now) can provide with more interesting results (Zahn, 1971). And fourthly, the dendrogram is an excellent exploratory data tool: its hierarchical structure gives interesting information on how students have been grouped and where isolated clusters can be found. Furthermore, it allows analyzing relations between students joined under the same node (this can be very useful in order to define behavior sub-profiles).

Hierarchical clustering algorithms need to fix two parameters in order to be properly configured. The first one is the distance function used to compare data. In this regard, Euclidean, Minkowski and Cosine are the most common distance functions when comparing time series (Gan *et al.*, 2007). It has to be taken into consideration that certain distance functions may require of certain pre-processing of the time series (normalization, noise removing, etc.). And the second one is the linkage method used to measure distance

Table I. Five-step strategy to modeling students' activity in online discussion forums

Steps	Parameters
1. Defining the data	<ul style="list-style-type: none"> Defining set of M students Defining type(s) of messages (all type, questions, answers...) Defining type(s) of actions (writing, reading...)
2. Constructing time series	<ul style="list-style-type: none"> Defining type and number of samples (N weeks, N days, N hours...) Defining the value of the times series data
3. Pre-processing time series	<ul style="list-style-type: none"> Normalization, denoising, linear trend removing... Time domain, frequency domain, feature extraction...
4. Agglomerative Hierarchical Clustering of time series	<ul style="list-style-type: none"> Defining distance function: Euclidean, Minkowski, Cosine... Defining linkage method: Single Link, Complete Link...
5. Identifying clusters (behavior profiles)	<ul style="list-style-type: none"> Visual identification of clusters By cutting dendrogram: distance threshold, inconsistency threshold...

between clusters. Single Link (nearest neighbor method) and Complete Link (farthest neighbor method) are the most usual methods (Gan *et al.*, 2007).

4. PRELIMINARY EXPERIMENTAL RESULTS

In order to illustrate the proposed strategy, preliminary results are presented. The experiment involves a virtual classroom of 55 online students in an Electronic Circuits Theory subject and covers an entire semester (369 writing and 14142 reading actions throughout 119 days). Students' writing and reading actions are separately characterized by two different time series, without distinguishing among different types of messages:

$$\mathbf{x}^{(i)} = \left(x_1^{(i)}, x_2^{(i)}, \dots, x_n^{(i)}, \dots, x_{119}^{(i)} \right) \rightarrow x_n^{(i)} = \begin{cases} 1 & \text{if } wr_n^{(i)} \neq 0 \\ 0 & \text{else} \end{cases} \quad \forall i \in [1, 55]$$

$$\mathbf{y}^{(i)} = \left(y_1^{(i)}, y_2^{(i)}, \dots, y_n^{(i)}, \dots, y_{119}^{(i)} \right) \rightarrow y_n^{(i)} = \begin{cases} 1 & \text{if } rd_n^{(i)} \neq 0 \\ 0 & \text{else} \end{cases}$$

Furthermore, Cosine distance (seeking for different activity paces) and Complete Link algorithm (emphasizing clusters compactness) have been used to cluster the times series data in time domain. The results of the experiment are shown in figure 1:

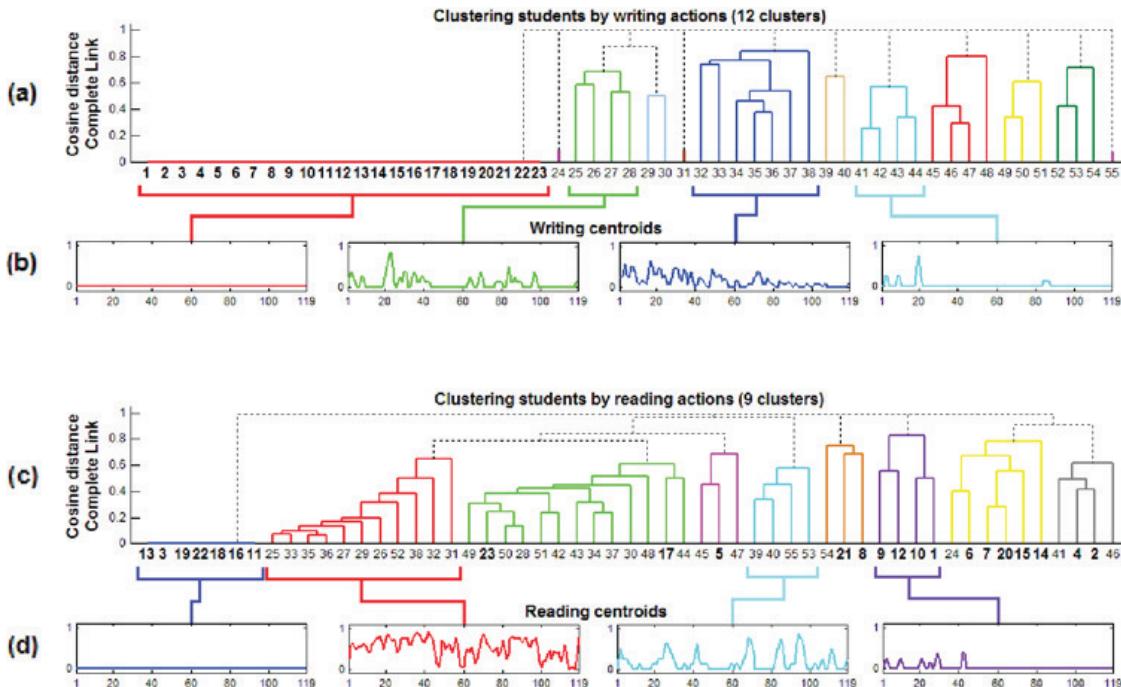


Fig. 1. (a): Writing clusters. (c): Reading clusters. (a) and (c): Bold-labeled students [1,23] are possible lurkers (no writing action). (b) and (d): Some representative profiles (centroids of some clusters) of both writing and reading throughout time are shown.

Some interesting remarks can be made from these results. Firstly, natural clusters (colored lines in Fig. 1 (a) and (c)) can be obtained by cutting dendrograms through suitable inconsistency thresholds (calculated according to (Zahn, 1971)), since these clusters are just hanging from the most inconsistent links (dotted lines in Fig. 1 (a) and (c)). In fact, the 9 clusters in Fig. 1 (c) cannot be obtained by cutting the dendrogram through any distance threshold, since there is no horizontal cut of the dendrogram that can provide with the same 9 clusters.

Secondly, resultant clusters centroids, both in terms of writing and reading (see Fig. 1 (b) and (d), respectively), indicate the most representative behavior profiles. Both total inactivity and regular activity profiles can be observed, as well as different profiles of more sporadic activity in different periods of time, for instance. This distinction based on different kinds of pace is possible because the Cosine distance tends to join profiles with coincident variations of activity throughout time.

And thirdly, students are grouped in different clusters depending on whether writing or reading actions are considered, which means that different behaviors can be associated to both kinds of activity. In fact, this strategy allows to easily distinguish between lurkers and inactive students: students #1, 2, 4, 5, 6, 7, 8, 9, 10, 12, 14, 15, 17, 20, 21 and 23 perform a lurking behavior, since they don't write at all but they do have active reading profiles; whereas #3, 11, 13, 16, 18, 19 and 22 are truly inactive students, since they neither write nor read (see bold-labeled students in Fig. 1 (a) and (c)).

5. CONCLUSIONS AND FURTHER WORK

A novel strategy to model students' activity in online discussion forums has been proposed. This strategy is based on characterizing students' activity in time series form and grouping similar students through an agglomerative hierarchical clustering algorithm.

Time series allow representing different kinds of activity (e.g., writing and reading) and thus identify several activity profiles throughout time. Different groups of students can be obtained by means of agglomerative hierarchical clustering without knowing the number of clusters a priori. Furthermore, dendograms are excellent exploratory data tools thanks to the rich information provided by their hierarchical tree structure.

Results obtained in a preliminary experiment point out that cutting dendograms through a robust criterion based on links' inconsistency is the most proper way to obtain natural clusters. The obtained activity profiles also depend on the chosen distance function and on how the values of times series data are set. Furthermore, the proposed strategy seems to be really suitable to identify lurking behaviors.

In this sense, the work started in this paper will be continued. More experiments will be carried out in order to improve the modeling of students' activity. Moreover, student's actions will be labeled by adding qualitative information about threaded posts (questions, answers, corrections, etc.), in order to obtain more relevant behavior profiles.

REFERENCES

- BEAUDOIN, M.F. 2002. Learning or lurking? Tracking the 'invisible' online student. *The Internet and Higher Education*, 2002, 5, 2, 147-155.
- FULFORD, C.P. AND ZHANG, S. 1993. Perceptions of interaction: The critical predictor in distance education. *The American Journal of Distance Education*, 1993, 7(3), 8-21.
- GAN, G., MA, C. AND WU, J. 2007. Data clustering. Theory, algorithms and applications. *ASIA-SIAM Series on Statistics and Applied Probability*, 2007.
- JAIN, A., AND DUBES, R. 1988. Algorithms for Clustering Data. *Prentice-Hall*, 1988.
- NONNECKE, B. AND PREECE, J. 2001 . Why lurkers lurk. *Americas Conf. on Information Systems*, 2001.
- RAVI, S. AND KIM, J. 2007. Profiling Student Interactions in Threaded Discussions with Speech Act Classifiers. In *Proceedings of the Conference on Artificial Intelligence in Education*. IOS Press, 357-364.
- ROMERO, C., VENTURA, S. AND HERVÁS, C. 2006. Educational data mining: A survey from 1995 to 2005. *Expert Systems with Applications*, 2006, 33, 135-146.
- ROMERO, C., VENTURA, S. PECHENIZKIY, M. AND BAKER, R. 2010. Handbook of educational data mining. *Data Mining and Knowledge Discovery Series*, Chapman & Hall / CRC Press, 2010.
- SACKVILLE, A. AND SHERRATT, C. 2006. Styles of Discussion: Online Facilitation Factors. In *Proceedings of the Fifth International Conference - Networked Learning*, Lancaster University, 2006.
- TAYLOR, J.C. 2002. Teaching and learning online: the workers, the lurkers and the shirkers. *Journal of Chinese Distance Education*, CCRTVU Press, Beijing, n. 9, 2002, 188, 31-37.
- VELLIDO, A., CASTRO, F. AND NEBOT, A. 2010. Clustering educational data. In *Handbook of educational data mining*, CRC Press, 2010, 75-92.
- ZAHN, C.T. 1971. Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, 1971, 20, 1.

Prediction of Perceived Disorientation in Online Learning Environment with Random Forest Regression

I. AKÇAPINAR

Hacettepe University, Turkey

II. COŞGUN

Hacettepe University, Turkey

AND

III. ALTUN

Hacettepe University, Turkey

In this study, it is aimed to predict users' perceived disorientation by using a data mining technique: Random Forest Regression (RFR). Two RFR models are designed to predict users' disorientation scores. The models were generalized with 10-fold Cross Validation (CV). In the first model, log based metrics were used as explanatory variables. In the second, log based metrics, eye metrics and self-report metrics were included in the model. According to findings, our RFR models predict perceived disorientation score with high accuracy. First model's R^2 is 57.8 %, second model's R^2 is 63.5 %. These results showed that adding eye metrics and self-report metrics to the model increased the predictive performance.

Key Words and Phrases: Disorientation, random forest, data mining, navigation log, eye tracking.

1. INTRODUCTION

In a hypermedia environment, information is presented nonlinearly and connectively unlike from written text. While this flexible structure provides learners with great browsing freedom, still there is a risk for some learners to become "lost in hyperspace" (Botafogo, Rivlin, & Shneiderman, 1992). Getting disoriented is one of the major difficulties that users experience while navigating (Conklin, 1987).

1.1 Disorientation

Conklin (1987) defined disorientation as the tendency to lose one's sense of location and direction in a non-linear document. Disorientation, or the tendency to lose one's sense of location in a Web site, can cause users to become frustrated, lose interest, and experience a measurable decline in efficiency (McDonald & Stevenson, 1998). While disoriented users feel more cognitive load, their learning performance is also affected negatively (Madrida, Oostendorpb, & Melguizob, 2009). For those reasons, much hypermedia research has been devoted to this matter (Otter & Johnson, 2000). There are two major approaches to measure user disorientation levels: objective or subjective measurements (Gwizdka & Spence , 2007). Disorientation in objective measurements is usually addressed with the term "lostness".

Smith (1996) proposed an objective measure of lostness based on the ratios of visited and optimal node counts. Otter and Johnson (2000) added a link's weight to Smith's

Authors' addresses: G. Akçapınar, Department of Computer Education and Instructional Technologies, Hacettepe University, Ankara, Turkey. E-mail: gokhana@hacettepe.edu.tr; E. Coşgun, Department of Biostatistics, Hacettepe University, Ankara, Turkey; E-mail : erdal.cosgun@hacettepe.edu.tr; A. Altun, Department of Computer Education and Instructional Technologies, Hacettepe University, Ankara, Turkey. E-mail: altunar@hacettepe.edu.tr

formula, proposing a weighted lostness formula. They also proposed another measure which is concerned with the accuracy of users' mental models of websites. Dias and Sousa (1997) introduced the orientation ratio which measures the degree of disorientation as the indicator of the accuracy of information retrieval.

On the other hand, Ahuja and Webster (2001) claimed that a better way to assess users' disorientation is asking users directly how they feel after navigating in a website. For this purpose, the authors validated a questionnaire on self-perceived disorientation and found that their questionnaire predicted performance on web information retrieval tasks better than user actions.

Although disorientation is a common problem for the internet users, it is difficult to measure it (Herder E. , 2003). In this study, we combined different kinds of metrics including eye movement data to determine the most important predictors of disorientation with the help of a data mining technique: Random Forest Regression (RFR).

1.2 Random Forest Regression

RFR is an effective nonparametric statistical technique for high-dimensional analysis. Random Forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest (Cosgun, Limdi, & W. Duarte1, 2011). The tree methods exhaustively break down cases into a branched, tree-like form until the splitting of the data is statistically meaningful, with unnecessary branches pruned using other test cases to avoid over-fitting (Choi & Lee, 2010). The generalization error for forests converges to a limit as the number of trees in the forest becomes large, and depends on the strength of the individual trees in the forest and the correlation between them (Breiman, 2001). Each tree in the forest is grown with following steps:

1. Draw a bootstrap sample from the data. Call those not in the bootstrap sample the "out-of-bag" data.
2. Grow a "random" tree, where at each node, the best split is chosen among randomly selected variables (m). The tree is grown to maximum size and not pruned back.
3. Use the tree to predict out-of-bag data.
4. Use the predictions on out-of-bag data to form majority votes.
5. Repeat, N times and collected an ensemble of N trees. Prediction of test data is done by majority votes from predictions from the ensemble of trees (Emir & Cabrera, 2009).

1.2.1 Variable importance

RFR determines the relative importance of each variable, through various methods, such as the calculation of the Gini Index, which assesses the importance of the variable and carries out accurate variable selection (Torri, Beretta, Ranghetti, Granucci, Ricciardi-Castagnoli, 2010). In every tree grown in the forest, put down the out-of-bag (oob) cases and count the number of votes cast for the correct class. After that randomly permute the values of variable m in the oob cases and put these cases down the tree. Subtract the number of votes for the correct class in the variable- m -permuted oob data from the number of votes for the correct class in the untouched oob data. The average of this number over all trees in the forest is the raw importance score for variable m (Breiman & Cutler, 2004).

2. METHOD

Thirty prospective high school computer teachers from Computer Education and Instructional Technologies Department of Hacettepe University (8 females and 22 males) participated in this study.

Data was collected on a web site that was designed by one of the researchers with the purpose of learning a given task. The content of the task included the basic SQL queries (Select, Update, Delete, Insert) and their use within the PHP language. The web site was exploratory in nature and hyperlinked across concepts. This topic was selected because it was new information for the participating students. They have already learned basic PHP language. A networked structure was used to present information consisting of 22 web pages (about 1000 words) and 57 cross-reference links between pages.

Table 1 shows the descriptions of metrics. Students' perceived disorientation was measured using the Turkish version of the Ahuja and Webster's (2001) self-report instrument. This instrument adapted to Turkish by Cangöz and Altun (2010). There were 10 statements and subjects were asked to indicate whether they agreed or disagreed with the statements on a 5-point Likert scale. Disorientation scores were calculated by the sum of user ratings of these statements. Log metrics were gathered through log data, which were collected from ASP (Active server page) coded learning environment. Time spent on a web site (duration), revisited page counts (revisited), unvisited page counts (unvisited) and unique page counts (unique) were extracted from these log files. Eye movement data were collected by Tobii T120 eye tracker. Fixation count and fixation duration metrics were selected for the study. Students' exam scores related to PHP language are taken as the prior knowledge score.

Table I. Description of metrics

	Description
Eye metrics	
FDonTitle	Total fixation duration on titles
FConTitle	Total fixation count on titles
FDonContent	Total fixation duration on contents
FConContent	Total fixation count on contents
FDonSample	Total fixation duration on code samples
FConSample	Total fixation count on code samples
FDonLink	Total fixation duration on links
FConLink	Total fixation count on links
Log metrics	
Duration	Total time spent while performing learning task
UniqueNav	Total number of unique web pages
Revisited	Total number of revisited web pages
Unvisited	Total number of unvisited web pages
Self-report metric	
PK	Learners' prior knowledge tests score

2.1 Design and Procedure

All of the learners performed a learning task in network structured hypermedia. While they were performing the task, their navigation data were logged and their eye movements were recorded by the eye tracker device. They were given a maximum 10 minutes to study the material. After they completed the task, they were requested to complete the perceived disorientation scale.

Random Forest Regression (RFR), were implemented using R-2.11.0 which is an open source software. We have used the Random Forest package for RFR, the ModelMap package for data manipulation.

3. RESULTS

Two different RFR regression models were developed to predict users' perceived disorientation. The first model included users' navigation log data. For the second model, users' eye metrics and prior knowledge scores were added to the model. Users' perceived disorientation was the dependent variable in both of the models.

According to findings related to Model 1, time spent on a web site, revisited page count and unvisited page count were the most important predictor of perceived disorientation (Table II). First model's R^2 is 57.8 %.

According to findings related to Model 2, fixation duration on samples, fixation count on contents, time spent on a web site, fixation count on samples, prior knowledge, unique navigation and fixation duration on links were more important predictors of perceived disorientation among others (Table II). Second model's R^2 is 63.5 %.

Table II. Predictor importance for Model 1 and Model 2

	Importance
Model 1	
Duration	1.00*
Revisited	0.50*
Unvisited	0.49*
UniqueNav	0.40
Model 2	
FDonSample	1.00*
FConContent	0.93*
Duration	0.93*
FConSample	0.76*
PK	0.68*
UniqueNav	0.63*
FDonLink	0.52*
FDonContent	0.48
FConTitle	0.48
Revisited	0.40
FConLink	0.38
FDonTitle	0.38
Unvisited	0.35

Important variables marked with asterisks (*)

4. CONCLUSION

In this study, two different RFR models were proposed, and these models found to be a predictor of perceived disorientation with high accuracy. In eye movement research, when users produce more fixations on certain fields on the screen, their attention is thought to be on that field rather than the other areas on screen (Poole, Ball, & Phillips, 2004). This could indicate that the viewer is either having difficulty in extracting information or the object is more engaging in some way (Just & Carpenter, 1976). When eye metrics were included in the model, predictive performance was increased, indicating that eye metrics are important predictors for perceived disorientation. If researchers who

study disorientation have a chance to collect data with eye tracker, they must use it; otherwise, log based metrics can also be applied with reported accuracy.

Unvisited and revisited page counts were found to be good predictors in a log based model. When other metrics (prior knowledge and eye metrics) were included in the model, their (unvisited and revisited page counts) importance decreased. Herder (2003) also reported that no correlation was found between users' perceived disorientation and percentage of revisits. This finding can be interpreted as users may use revisits as a navigation strategy.

Time spent on a specific web page is an important predictor of disorientation in both models. Amadieu, Gog, Paas, Tricot, and Mariné (2009) found that low prior knowledge learners experienced higher disorientation than their higher counterparts during learning with a network structured concept map. In both models, prior knowledge was found to be an important predictor of user perceived disorientation.

We have a limitation on sample sizes. Because, in nature of these kind of educational data, number of users are limited. But data mining methods are very useful for determine the best solutions. In this study we have tried to implement Random Forest Regression algorithm for finding best model on "disorientation data". For discard the disadvantages of sample size, our entire process was contained within a 10-fold cross validation structure. This approach is helpful for generalized our findings. In future studies we are going to use this "base model" and try to increase our prediction performance.

REFERENCES

- Ahuja, J., & Webster, J. (2001). Perceived disorientation: an examination of a new measure to assess web design effectiveness. *Interacting with Computers*, 14, 15-29.
- Amadieu, F., Gog, T. V., Paas, F., Tricot, A., & Mariné, C. (2009). Effects of prior knowledge and concept-map structure on disorientation, cognitive load, and learning. *Learning and Instruction*(19), 376-386.
- Botafogo, R. A., Rivlin, E., & Shneiderman, B. (1992). Structural analysis of hypertexts: identifying hierarchies and useful metrics. *ACM Transactions on Information Systems*, 10, 142-180.
- Breiman, L. (2001). Random Forests. *Machine Learning*, 5-32.
- Breiman, L., & Cutler, A. (2004). *Random Forests*. Retrieved from Random Forests: http://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm#varimp
- Cangöz, B., & Altun, A. (2010). Bilgisayar geziniminde örtük bellek ve algılanan oryantasyon kaybının rolü. (The role of implicit memory and perceived disorientation in navigation). *16. Ulusal Psikoloji Kongresi*. Mersin: Mersin Üniversitesi.
- Choi, M., & Lee, G. (2010). Decision tree for selecting retaining wall systems based on logistic regression analysis. *Automation in Construction*, 917-928.
- Conklin, J. (1987). Hypertext: An introduction and survey. *IEEE Computer*, 20(7), 17-41.
- Cosgun, E., Limdi, N., & W. Duarte1, C. (2011). High Dimensional Pharmacogenetic Prediction of a Continuous Trait using Machine Learning Techniques with Application to Warfarin Dose Prediction in African Americans. *Bioinformatics Advance Access*, 1-6.
- Dias, P., & Sousa, A. P. (1997). Understanding navigation and disorientation in hypermedia learning environments. *Journal of Educational Multimedia and Hypermedia*, 173-185.
- Emir, B., & Cabrera, J. (2009, May 10). Course Notes of "Exploring/Data Mining Pharmaceutical Data". İstanbul, Turkey.
- Gwizdka, J., & Spence , I. (2007). Implicit measures of lostness and success in web navigation. *Interacting with Computers.*, 19(3), 357-369.
- Herder, E. (2003). Revisitation Patterns and Disorientation. In *Proceedings of the German Workshop on Adaptivity and User Modeling in Interactive Systems ABIS 2003* (pp. 291-294). Karlsruhe: University of Karlsruhe.
- Just, M. A., & Carpenter, P. A. (1976). Eye fixations and cognitive processes. *Cognitive Psychology*, 8, 441-480.
- Madrida, R., Oostendorpb, H., & Melguizob, M. (2009). The effects of the number of links and navigation support on cognitive load and learning with hypertext: The mediating role of reading order. *Computers in Human Behavior*, 66-75.
- McDonald, S., & Stevenson, R. J. (1998). Effects of text structure and priorknowledge of the learner on navigation in hypertext. *Human Factors*, 18-27.
- Poole, A., Ball, L. J., & Phillips, P. (2004). In search of salience: A response time and eye movement analysis of bookmark recognition. In P. M. S. Fincher, *People and Computers XVIII-Design for Life: Proceedings of HCI 2004* (pp. 363-378). London: Springer-Verlag.
- Smith, P. A. (1996). Towards a practical measure of hypertext usability. *Interacting with Computers*, 365-38.

Analysing Student Spatial Deployment in a Computer Laboratory

VLADIMIR IVANČEVIĆ,
University of Novi Sad, Serbia

MILAN ČELIKOVIĆ,
University of Novi Sad, Serbia

AND

IVAN LUKOVIĆ
University of Novi Sad, Serbia

Although data mining techniques are constantly improved and made more easy to use, there is still little application of related methods in the investigation of student seating choices and respective implications. Therefore, in this paper we analyse characteristics and effects of student spatial deployment in a computer laboratory by using data mostly gathered through electronic student testing system. For a first semester programming course, we examine relations between student assessment results and student choices of seating locations in a laboratory, as well as search for regularities in the spatial deployment, such as students' test location changes throughout the semester and the connection they share with the success on tests. Obtained results demonstrate the existence of patterns, whose meaning is also discussed, and emphasise the importance of considering student spatial choices in the study of understanding and improving student behaviour and learning.

Key Words and Phrases: student spatial deployment, student seating, mining assessment data

1. INTRODUCTION

Research on seating arrangements of students in classrooms has always been a familiar topic within the scientific community. Although the latest data analysis and investigation techniques have found their way into the field known as educational data mining (EDM), that has not led to substantial improvements in the study of student spatial deployment in a classroom, which may seem unusual since computers can contribute much to the automation of data gathering and shortening of the duration of the study.

In this paper, we propose an approach which focuses on the usage of data mining (DM) techniques in the examination of student spatial deployment in a classroom. Our main goal is to demonstrate how to prepare data and extract information necessary for the modern research on student seating locations. Analyses were conducted using data from a basic programming course organized at the Faculty of Technical Sciences (FTS) of University of Novi Sad and they include: investigation of student assessment results with respect to the student seating choices in the classroom; and investigation of migration and changes in student seating choices in the classroom with respect to assessment results.

2. RELATED WORK

Different studies concerning implications of choosing different seating locations in a classroom at the university level often have seemingly contradicting results. Some claim that seating location does not influence grades (Kalinowski et al, 2007) or that the impact

Authors' addresses: V. Ivančević, Department of Computing and Control, Faculty of Technical Sciences, University of Novi Sad, Novi Sad, Serbia. E-mail: dragoman@uns.ac.rs; M. Čeliković, Department of Computing and Control, Faculty of Technical Sciences, University of Novi Sad, Novi Sad, Serbia. E-mail: milancel@uns.ac.rs; I. Luković, Department of Computing and Control, Faculty of Technical Sciences, University of Novi Sad, Novi Sad, Serbia; E-mail : ivan@uns.ac.rs

is so weak that it can be considered unimportant (Montello, 1988). Others state that the seating location is linked to a better willingness to participate in classes (Çinar, 2010) or that the seating in the back leads to lower grades when compared to the front (Gossard et al, 2006). Moreover, even with the initially randomized seating arrangement, there was a positive impact of the seating in the front of the lecture hall (Perkins et al, 2005).

Unfortunately, the use of computers and data mining in those circumstances has been scarce, which could be attributed to the fact that an infrastructure for automatically tracking student seating locations is still not available in most classrooms, so the process of recording seating choices is impractical and labourious. In order to avoid the manual data collection, we relied on the automatic logging system available in the laboratory.

3. THE ENVIRONMENT

3.1 The Laboratory

The analysed course was held in a laboratory (see Fig. 1) specifically designed for computer science education at FTS (Rakić et al, 2007). It features 32 student work locations, each with a single personal computer uniquely identified by a number in the range of 200 to 231, and a separate computer for the instructor. In the same figure, the typical positions of teaching assistants are marked by stars: the leading teaching assistant (LTA) standing at the front of the classroom and presenting a topic; and the supporting teaching assistant (STA) moving around the room. Student evaluation in the course was conducted using a software for computer-based student testing developed at the same laboratory (Živanov et al, 2008), and monitored by teaching assistants, both directly and with the help of a surveillance system. In order to be analysed, the student work locations were divided according to their distances from the LTA in 3 location groups (see Fig. 1): *Front-Zone* (11 locations: 200-202, 210-212, 220-224); *Mid-Zone* (11 locations: 203-206, 213-216, 225-227); and *Back-Zone* (10 locations: 207-209, 217-219, 228-231).

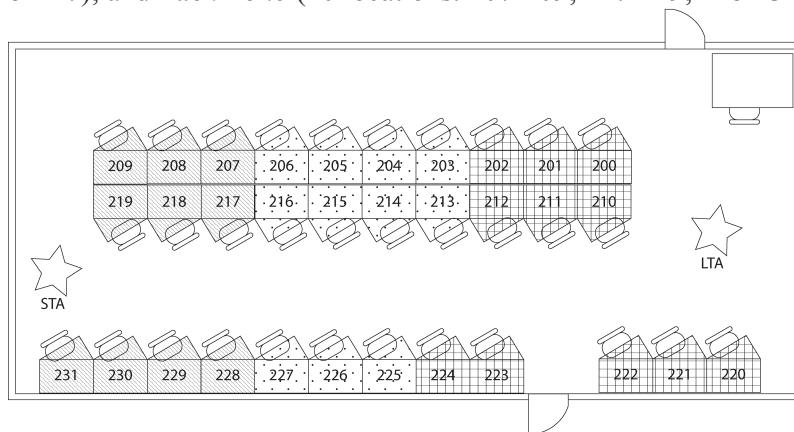


Fig. 1. An overview of the laboratory

3.2 Data Description

We examined students' results achieved on tests conducted as a part of the *Programming Languages and Data Structures* course, the first year mandatory course of *Computer and Control* curriculum carried out at FTS in the winter semester of academic year 2010/2011. The data encompassed 3 tests with 153 students involved and 430 individual assessments. These data were extracted from the testing system logs and include: course, test, and student group identification; basic student data (id, first name, and last name); and test data (achieved and maximum score, computer IP address, and completion time).

4. ANALYSING SPATIAL DEPLOYMENT

Further in this section we present our analyses conducted through *Oracle Data Miner* software tool. As we attempt to make general conclusions relying on 3 tests data, we need to make one assumption about the environment: *student seating choices at tests are representative of their seating choices during the non-test classes*. This claim is supported by the practical observations made during the course and by the findings presented later showing that many students chose the same location or the one nearby.

4.1 Analysing Student Scores with Respect to Location

For each location in the laboratory, we determined the number of students who sat at that location during the assessments (*SCN*), as well as the sum of all the test scores made by those students when they sat there (*SCSUM*). Next, for each location we calculated its corresponding average test score *SCAVG* as the quotient of *SCSUM* and *SCN*. Since FTS uses a six-point grade scale from 5 (unsatisfactory) to 10 (excellent), these single test scores (integers from the range of -5 to 10) can also be interpreted as grades. We clustered the locations by their *SCAVG* and *SCN* values using an enhanced version of the *k*-Means algorithm (Oracle, 2008) and obtained 4 groups: *s1* (characterised by wide score range and low *SCN*), *s2* (low *SCAVG* and high *SCN*), *s3* (high *SCAVG* and moderate *SCN*), and *s4* (moderate *SCAVG* and moderate *SCN*). Their properties are shown in Table 1, together with member distributions among 3 location zones (*Front*, *Mid*, and *Back*).

Table I. Locations clustered by average test score and individual test count

Clust.	Size	Confidence[%]	Support Count	Front[%]	Mid[%]	Back[%]
<i>s1</i>	6	100	6	17	33	50
	Rule: $4.295 \leq SCAVG \leq 8.21$ and $10.0 \leq SCN \leq 12.5$					
<i>s2</i>	6	83.3	5	17	50	33
	Rule: $4.73 \leq SCAVG \leq 5.60$ and $14.0 \leq SCN \leq 14.5$					
<i>s3</i>	5	80.00	4	60	20	20
	Rule: $6.91 \leq SCAVG \leq 7.34$ and $12.5 \leq SCN \leq 15.0$					
<i>s4</i>	15	86.67	13	40	33	27
	Rule: $5.6 \leq SCAVG \leq 6.91$ and $12.5 \leq SCN \leq 14.5$					

The highest scores (*s3*) are mostly present in the *Front*, while the lower scores (*s2*) are featured more in the *Mid* and less in the *Back*. It is interesting to note that the number of locations with the low *SCN* increases when going from the front to the back (*s1*). Cluster with the moderate *SCAVG* and moderate *SCN* (*s4*) is the largest, as well as the most balanced among the zones. This may suggest that the test scores are dependent on other factors, although the link to location LTA distance is observable in smaller clusters.

Additionally, we utilised the *Anomaly Detection* technique (Oracle, 2008) which employs the single-class Support Vector Machine (SVM) method (Boser, 1992) to detect atypical locations with respect to their *SCAVG* and *SCN* values. We found 4 anomalous locations: 216, 219, 223, and 231. Location 219 was anomalous since it has a high score with low test count, while the other seats got labelled because of the associated low scores. One common trait is that three of them (219, 223, and 231) are border locations (only one direct neighbouring location). This spatial information is not an inherent part of *SCAVG*, nor *SCN*, but it emerges from the findings. The farthest locations (219 and 231) also stand out because of their low *SCN*, possibly owing to lower visibility and audibility.

If we arrange the locations into 8 equal-size bins by their distance from LTA, we can plot the dependency between the average test scores expressed in points and the location LTA distance in centimetres (see Fig. 2) which shows that the highest test scores are linked to front locations, although the farthest seats also have above average values.

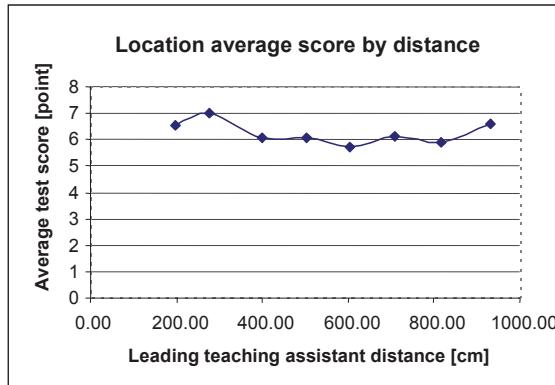


Fig. 2. Location average test score by leading teaching assistant distance for all three tests

4.2 Analysing Student Migration within the Laboratory

For each of the 131 students who had done all three tests in the course, we calculated the average test score and the number of different seats occupied during the tests. Next, we divided students into 3 groups according to the associated number of different seats. Group size, as well as average test score, is shown in Table 3. The findings demonstrate a one level higher grade of the group of students who never changed the seat when compared to those who changed the seat each test. Furthermore, as the number of associated seats increases, both the number of students and average score decrease. This shows that nearly half of the students (47%) stay at the same location, while the rise in the frequency of seat change corresponds to the drop in test scores.

Table III. Student average scores and count by the number of chosen locations

Number of chosen locations	Student count	Average test score
1	61	6.72
2	46	6.06
3	24	5.63

In addition to that, we calculated location positions in the laboratory, as well as the distance (d) that student covered during n analysed tests ($n=3$):

$$d = \sqrt{\sum_{i=1}^n \left(\frac{\sum_{j=1}^n x_j}{n} - x_i \right)^2 + \left(\frac{\sum_{j=1}^n y_j}{n} - y_i \right)^2}$$

where: x_i, y_i – x and y coordinates of the location where student sat during the test i . Next, using the enhanced k -Means algorithm, we clustered the students according to the respective distance covered and number of different locations associated with the student, thus obtaining *distance clusters*: $d1$ (1 location, no distance covered), $d2$ (3 locations, any distance), $d3$ (2 locations, longer distance), and $d4$ (2 locations, shorter distance). Moreover, we clustered the same students according to their summed test scores and formed 4 *student test score clusters* (given from lowest to highest scores): $gLow$, $gMid$, $gMid2$, and $gHigh$. Finally, for each distance cluster we determined how many of its members belong to each of the four test score clusters (see Table 4).

Obtained results support previous findings. Many students who did not change the location a single time ($d1$) are present in the two groups with highest test scores ($gMid2$ and $gHigh$). Those who changed their location for each test ($d2$) are best represented by

the groups with lower scores (*gLow* and *gMid1*). Students who changed the location only once have more pronounced medium scores (*d3*) or scores more balanced among the whole score range (*d4*). The last finding could represent all those students whose typical location was occupied by someone else, so they had to select another seat close to the original one. As this could happen to any student, the differences between the percentages in this cluster are the smallest - all score ranges are more or less equally represented. Moreover, clusters *d1* and *d4* represent students who always keep the same seat (*d1*) or do not move far from their preferred seat (*d4*). Since they make 71% of the population, the student location switch inertia is further confirmed.

Table IV. Covered distance clusters by student test score clusters

Distance cluster	Size	<i>gLow</i> [%]	<i>gMid1</i> [%]	<i>gMid2</i> [%]	<i>gHigh</i> [%]
d1	61	21.31	18.03	24.59	36.07
d2	24	33.33	33.33	20.83	12.5
d3	14	14.29	28.57	35.71	21.43
d4	32	28.12	25	18.75	28.12

5. CONCLUSION

In the paper, we discuss the application of DM techniques for the investigation of student seating arrangements in the classroom. The proposed approach uncovered practical results which show that with the increased location distance from the instructor, scores tend to drop to a stable level and that border locations are outliers in terms of associated test scores and occupancy. Furthermore, the analyses of classroom migration demonstrate that students who do not change the seating location have, on average, a one grade level higher score than the others. Also, many students gravitate towards such behaviour.

Information acquired through these means could be used for the development of an algorithm for the identification of optimal seating arrangements which would diminish negative effects of a bad seating choice on student learning. Further work could also include the examination of factors influencing student seating choices, as well as. the integration of the proposed analyses with the existing student testing system.

ACKNOWLEDGEMENTS

The research was supported by Ministry of Science and Technological Development of Republic of Serbia, Grant III-44010.

REFERENCES

- BOSER, B. E., GUYON, I. M., AND VAPNIK, V.N. 1992. A training algorithm for optimal margin classifiers, In *Proceedings of the 5th Annual ACM Workshop on COLT*, 144-152
- ÇINAR, İ. 2010. Classroom geography: Who sit where in the traditional classrooms?, *The Journal of International Social Research*, 3(10), 200-212
- GOSSARD, M. H., JESSUP, E., AND CASAVANT, K. 2006. Anatomy of a classroom: An exploratory analysis of elements influencing academic performance, *NACTA Journal*, 6, 36-39
- KALINOWSKI, S., AND TAPER, M. L. 2007. The effect of seat location on exam grades and student perceptions in an introductory biology class, *Journal of College Science Teaching*, 1, 54-57
- MONTELLO, D. R. 1988. Classroom seating location and its effect on course achievement, participation, and attitudes, *Journal of Environmental Psychology*, 8, 149-157
- ORACLE 2008. *Oracle Data Mining Concepts, 11g Release 1 (11.1)*, B28129-04, Oracle
- PERKINS, K. K. AND WIEMAN C. E. 2005. The surprising impact of seat location on student performance, *The Physics Teacher*, 43, 30-33
- RAKIĆ, P., STRIČEVIĆ, L., ŽIVANOV, Ž., SUVAJDŽIN, Z., AND HAJDUKOVIĆ, M. 2007. Computer classroom: Deployment and exploitation, *Info M*, 6(21), 9-13.
- ŽIVANOV, Ž., RAKIĆ, P., STRIČEVIĆ, L., PUŠIĆ, B., SUVAJDŽIN, Z., AND HAJDUKOVIĆ, M. 2008. Computer aided student examination, *Info M*, 7(25), 45-53.

Predicting School Failure Using Data Mining

C. MÁRQUEZ-VERA

Autonomous University of Zacatecas, Mexico

C. ROMERO AND S. VENTURA

University of Cordoba, Spain

This paper proposes to apply data mining techniques to predict school failure. We have used real data about 670 middle-school students from Zacatecas, México. Several experiments have been carried out in an attempt to improve accuracy in the prediction of final student performance and, specifically, of which students might fail. In the first experiment the best 15 attributes has been selected. Then two different approaches have been applied in order to resolve the problem of classifying unbalanced data by rebalancing data and using cost sensitive classification. The outcomes of each one of these approaches using the 10 classification algorithms and 10 fold-cross validation are shown and compared in order to select the best approach to our problem.

Key Words and Phrases: School failure, Educational Data Mining, Prediction, Classification.

1. INTRODUCTION

Recent years have shown a growing interest and concern in many countries about the problem of school failure and the determination of its main contributing factors. This problem is known as the “the one hundred factors problem” and a great deal of research has been done on identifying the factors that affect the low performance of students (school failure and dropout) at different educational levels (primary, secondary and higher) (Araque et al., 2009). A very promising solution to resolve this problem is the use of Data Mining (DM) that is called Educational Data Mining (EDM) when applied to an educational context (Romero and Ventura, 2010). There are examples about how to apply EDM techniques for predicting drop out and school failure (Kotsiantis, 2009). These works have shown promising results with respect to those sociological, economic or educational characteristics that may be more relevant in the prediction of low academic performance. It is also important to notice that most of this research on EDM applied to resolve this problems have been applied primarily to the specific case of higher education (Kotsiantis, 2009) and more specifically to online or distance education (Lykourentzou et al., 2009). However, very little information has been found in specific research on elementary and secondary education, and what has been found only uses statistical methods, not DM techniques (Parker, 1999).

2. DESCRIPTION OF THE DATA USED

This paper uses data from students (of about 15 years of age) admitted to the Academic Program 2 of UAPUAZ. It must be pointed out that a very important task in this work was information gathering and data pre-processing due to the quality and reliability of available information which directly affects the results obtained. All the information used in this study has been gathered from three different sources during the period from August to December 2010:

Authors' addresses: C. Márquez-Vera, Autonomous University of Zacatecas, Mexico. E-mail: cmarquezvera@hotmail.com; S. Ventura, C. Romero, Department of Computer Science, University of Cordoba, Spain; E-mail: sventura@uco.es, cromero@uco.es;

- a) A specific survey was designed and administered to all students in the middle of the course. Its purpose was to obtain personal and family information to identify some important factors that could affect school performance.
- b) A general survey from the National Evaluation Center (CENEVAL) for admission to many institutions of secondary and higher education. When students register for the admission exam (EXANI I), this Center also carries out a socioeconomic study to obtain this information.
- c) The final scores obtained by students in different subjects in the course provided by the Department School Services of the UAPUAZ at the end of the semester.

Finally, the output variable/attribute or class to be predicted in our problem is the academic status or final student performance that has two possible values: PASS (student who pass the course) or FAIL (student who has to repeat the course). This attribute has been provided by the Academic Program of UAPUAZ at the end of the course.

Starting from all this information we have created a dataset (split in 10 folds) with 77 attributes/variables about 670 students of whom 610 passed and 60 failed the course.

3. EXPERIMENTATION AND RESULTS

To do all the experiments, ten classification algorithms have been used that are available in the well-known Weka DM software (Witten et al., 2011): five rule induction algorithms such as JRip, NNge, OneR, Prism and Ridor; and five decision tree algorithms such as J48, SimpleCart, ADTree, RandomTree and REPTree. These algorithms have been selected because they are considered as “white box” classification model, that is, they provide an explanation for the classification result and can be used directly for decision making.

In the first experiment, the 10 classification algorithms have been executed using 10 fold-cross validation and all the available information, that is, the original data file with 77 attributes of 670 students. The results with the test files (an average of 10 executions) of classification algorithms are shown in Table I (A). This table shows the overall accuracy rate (Accuracy), the rates or percentages of correct classification for each of the two classes: Pass (TP rate) and Fail (TN rate) and the Geometric Mean (GM) that is a measure of the central tendency used with unbalanced datasets. It can be seen in Table II (A) that the values generally obtained are high in accuracy (greater than 91.5%) and in the TP rate (greater than 95.7%), but not so high in the TN rate (greater than 25%) and the Geometric mean (greater than 49.9%). The best algorithm in the TP rate and Accuracy was ADTree (99.7% and 97.6% respectively), in the TN rate and Geometric mean was Jrip (78.3% and 87.5% respectively).

After this first experiment using all available attributes, we have notice that in all the obtained model only few of the large number of attributes used (77) appear. So, we decided to do carry out a study of feature selection in order to try to identify which of them has the greatest effect on our output variable (academic status). The objective is to reduce the number of attributes without losing reliability in classification. Weka provides several feature selection algorithms from which we have selected the following ten (Witten et al., 2011): CfsSubsetEval, ChiSquaredAttributeEval, ConsistencySubsetEval, FilteredAttributeEval, OneRAttributeEval, FilteredSubsetEval, GainRatioAttributeEval, InfoGainAttributeEval, ReliefFAttributeEval, SymmetricalUncertAttributeEval. The results obtained have been ranked by these ten algorithms to select the best attributes using our 77 available attributes. In order to find the ranking of the attributes, we have counted the number of times each attribute was selected by one of the algorithms (see

Table I) and only those with a frequency greater than 2 have been selected as the best attributes.

TABLE I. MOST INFLUENTIAL ATTRIBUTES RANKED BY FREQUENCY OF APPEARANCE

Attribute	Frequency
Scores in Humanities 1, and in English 1	10
Scores in Social Science 1, Math 1, Reading and Writing 1, Physics 1, and Computer 1	9
Level of motivation	5
Grade Point Average in secondary	3
Age, number of brothers/sisters, classroom/group, smoking habits, and average score in EXANI I	2
Studying in group, marital status, time spent doing exercises, and score in History	1

The selection of the attributes with a frequency greater than 2 has reduced the dimensionality of our dataset from the original 77 attributes to only the best 15 attributes. Starting from these 15 attributes a second experiment has been carried. Table I (B) shows the results with the test files (the average of 10 executions) using only the best 15 attributes. When comparing the results obtained versus the previous one using all the attributes, that is, Table II (A) versus (B), we can see in general that all the algorithms have improved in some measures (TN Rate and Geometric mean). And about the other measures (TP rate and Accuracy) there are some algorithms that obtain a bit worse or a bit better values, but very similar in general to the previous ones. In fact, the maximum values obtained now are better than the previous ones obtained using all attributes in two evaluation measures (TN rate and Geometric mean). The algorithm that obtains these maximum values is Jrip (81.7% TN rate and 89% Geometric Mean). However, although these results are better than the previous one; they are still very lower than the obtained by TP rate (greater than 95.6% and a maximum value of 99.2%) and Accuracy (greater than 93.1% and a maximum value of 97.3%). This is because our data are imbalanced.

TABLE I. COMPARISON OF RESULTS: (A) USING ALL ATTRIBUTES, (B) USING THE BEST ATTRIBUTES.

Algorithm	A				B			
	TP rate	TN rate	Accuracy	Geometric Mean	TP rate	TN rate	Accuracy	Geometric Mean
Jrip	97,7	78,3	96	87,5	97	81,7	95,7	89
NNge	98,5	73,3	96,3	85	98	76,7	96,1	86,7
OneR	98,9	41,7	93,7	64,2	98,9	41,7	93,7	64,2
Prism	99,5	25	93,1	49,9	99,2	44,2	94,7	66,2
Ridor	96,6	65	93,7	79,2	95,6	68,3	93,1	80,8
ADTree	99,7	76,7	97,6	87,4	99,2	78,3	97,3	88,1
J48	97,4	53,3	93,4	72,1	97,7	55,5	93,9	73,6
RandomTree	95,7	48,3	91,5	68	98	63,3	94,9	78,8
REPTree	98	56,7	94,3	74,5	97,9	60	94,5	76,6
SimpleCart	97,7	65	94,8	79,7	98	65	95,1	79,8

The problem of imbalanced data classification occurs when the number of instances in one class is much smaller than the number of instances in another class or other classes (Gu et al., 2008). Traditional classification algorithms have been developed to maximize the overall accuracy rate, which is independent of class distribution; this causes majority class classifiers in the training stage, which leads to low sensitivity classification of minority class elements at the test stage. One way to solve this problem is to act during the pre-processing of data by making a sampling of or a balance of class distribution. There are several data balancing or rebalancing algorithms and one that is widely used and available in Weka as a supervised data filter is SMOTE (Synthetic Minority Over-sampling Technique). In general, SMOTE (Nitesh et al., 2002) introduces minority class elements synthetically, considering the nearest neighbor elements of the same class. In our case, only the 10 training files (with the best 15 attributes) have been

rebalanced using the SMOTE algorithm, obtaining 50% Pass students and 50% Failed students and not rebalancing the test files. The results obtained after re-executing the 10 classification algorithms using 10 fold-cross validation are summarized in Table III (A). If we analyze and compare this Table versus the previous Table II, we can observe that slightly over half of the algorithms have increased the values obtained in all the evaluation measures, and some of them also obtain the new best maximum values in almost all measures except accuracy: Prism (99.8% TR rate), OneR (88.3% TN rate) and ADTree (97.2% Accuracy and 92.1% Geometric Mean).

TABLE III. COMPARISON OF RESULTS: (A) USING DATA-BALANCING, (B) USING COST-SENSITIVE

Algorithm	A				B			
	TP rate	TN rate	Accuracy	Geometric Mean	TP rate	TN rate	Accuracy	Geometric Mean
Jrip	97.7	65	94.8	78.8	96.2	93.3	96	94.6
Nnge	98.7	78.3	96.9	87.1	98.2	71.7	95.8	83
OneR	88.8	88.3	88.8	88.3	96.1	70	93.7	80.5
Prism	99.8	37.1	94.7	59	99.5	39.7	94.4	54
Ridor	97.9	70	95.4	81.4	96.9	58.3	93.4	74
ADTree	98.2	86.7	97.2	92.1	98.1	81.7	96.6	89
J48	96.7	75	94.8	84.8	95.7	80	94.3	87.1
RandomTree	96.1	68.3	93.6	79.6	96.6	68.3	94	80.4
REPTree	96.5	75	94.6	84.6	95.4	65	92.7	78.1
SimpleCart	96.4	76.7	94.6	85.5	97.2	90.5	96.6	93.6

A different approach to solving the problem of imbalanced data classification is to apply cost-sensitive classification (Elkan, 2001). Optimizing the classification rate without taking into consideration the cost of errors can often lead to suboptimal results because high costs can result from the misclassification of a minority instance. In fact, in our particular problem, we are much more interested in the classification of Fail students (the minority class) than Pass students (the majority class). These costs can be incorporated into the algorithm and considered during classification. In the case of 2 classes, costs can be put into a 2x2 matrix in which diagonal elements represent the two types of correct classification and the off-diagonal elements represent the two types of errors. Weka allows any classification algorithm to be made cost sensitive by using the meta-classification algorithm CostSensitiveClassifier and setting its base classifier as the desired algorithm. In fact, the CostSensitiveClassifier and our 10 classification algorithms have been applied as base classifiers using the original test and training files with the best 15 attributes. We have also selected (0, 1; 4, 0) as the cost matrix because it obtained the best results. This matrix indicates that performing the classification takes into consideration that it is 4 times more important to correctly classify Fail students than Pass students. Table III (B) shows the results with test files obtained after applying 10 fold-cross validation. On analyzing and comparing Table III (B) versus Table III (A), some algorithms can be seen to obtain better values in some evaluation measures while other algorithms obtain worse values. So, there is no clear general improvement. However, one algorithm (Jrip) does obtain the current best maximum values on the TN rate (93.3%) and Geometric mean (94.6), which is very important in our problem.

4. DISCUSSION AND FUTURE WORK

Regarding the different used approaches and the classification results obtained, the main conclusions are:

- We have shown the utility of feature selection techniques when we have a great number of attributes. In our case, we have reduced the number of attributes used from the 77 to 15 attributes, without losing classification performance.
- We have shown two ways to address the problem of imbalanced data classification by rebalancing the data and considering different classification costs. These approaches have been able to improve the classification results obtained in one or several evaluation measures.
- We can select cost-sensitive classification as the best approach because it obtains not only very good classification results in the minority class (Fail students), but also in the majority class (Pass students).

Regarding the specific knowledge extracted from the classification models obtained, the main conclusions are:

- White box classification algorithms obtain models that can explain their predictions at a higher level of abstraction by IF-THEN rules. These types of rules are easily understood and interpreted by non-expert DM users. In this way a non-expert user of DM such as a teacher or instructor can directly use the output obtained by these algorithms to detect students with problems (classified as fail) and to make decisions about how to help them and prevent their possible school failure.
- There are some factors/attributes and specific values that appear more in the models when predicting the students who will fail in the classification models obtained. For example, the scores/grades that most appear in the obtained classification rules are the values of "Deficient" or "Not Presented" in the subjects of Physics, Humanities, Math and English. Other factors frequently associated with failing are: to be over 15 years of age, to have more than one brother/sister, to be attending an evening classroom/group, and to have a low level of motivation to study.

Finally, as the next step in our research, we want to develop our own classification algorithm using grammar-based genetic programming and cost sensitive classification for comparison versus other classification algorithms.

ACKNOWLEDGMENTS

This research is subsidized by P08-TIC-3720 and TIN2008-06681-C06-03 projects, and FEDER funds.

REFERENCES

- ARAQUE F., ROLDAN C., SALGUERO A. 2009. Factors Influencing University Drop Out Rates. *Computers & Education*, 53, 563–574.
- ELKAN, C. 2001. The Foundations of Cost-Sensitive Learning. International Joint Conference on Artificial Intelligence, 1-6.
- Gu, Q., Cai, Z., Zhu, L., Huang, B. 2008. Data Mining on Imbalanced Data Sets. Proceedings of International Conference on Advanced Computer Theory and Engineering, Phuket, 1020-1024.
- Kotsiantis S. 2009. Educational Data Mining: A Case Study for Predicting Dropout – Prone Students. *Int. J. Knowledge Engineering and Soft Data Paradigms*, 1(2), 101–111.
- LYKORENTZOU I., GIANNOUKOS I., NIKOPOULOS V., MPARDIS G. AND LOUMOS V. 2009. Dropout Prediction in e-learning Courses through the Combination of Machine Learning Techniques. *Computers & Education*, 53, 950–965.
- NITESH V. CHAWLA ET. AL. 2002. Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, 16, 321-357.
- PARKER A. 1999. A Study of Variables That Predict Dropout From Distance Education. *International Journal of Educational Technology*, 1(2) 1-11.
- ROMERO C. AND VENTURA S. 2010. Educational Data mining: A Review of the State of the Art. *IEEE Transactions on Systems, Man, and Cybernetics*. 40(6), 601-618.
- WITTEN I. H., EIBE F. AND HALL, M.A. 2011. Data Mining, practical Machine Learning Tools and Techniques. Third Edition. Morgan Kaufman Publishers.

A Dynamical System Model of Microgenetic Changes in Performance, Efficacy, Strategy Use and Value during Vocabulary Learning

P. PAVLIK JR. AND S. WU

Carnegie Mellon University, United States

This paper describes the development of a dynamical systems model of self-regulated learning, which explains the practically and theoretically important dynamic relationships among three student-engagement constructs and performance during learning. This work mined data from computerized adaptive flashcard learning to create this dynamical systems model. This flashcard practice included pop-up survey questions on the student's experience of recent easiness, strategy use, and usefulness, in addition to the correctness performance data for the practice. Using this dynamical systems model, we were then able to simulate various user profiles to predict how they would experience the flashcard system. These simulations show how strategy use in this task is crucial because of the ways it influences performance, particularly over time. In the model, this result is shown by a bifurcation into two different regions for higher and lower strategy use, where the higher strategy-use equilibrium state is accompanied by performance predictions suggesting learning that is more efficient.

Key Words and Phrases: Dynamical system model, language learning, motivation, metacognition, efficacy, and utility

1. INTRODUCTION

Many sources agree that motivation is a dynamic construct that changes from moment to moment with the stream of events students experience [e.g. Witherspoon et al. 2007]. Of course, while each period marks changes in student attitudes, perceptions and actions, there is also continuity across time as attitude states shift not randomly, but as a function of prior attitudes, perceptions and actions. If these different variables cause change in each other, in addition to this continuity, we have described a dynamical system. This perspective on dynamic motivation is similar to Csikszentmihalyi's definition of the state of flow that occurs for people during activities that balance skills and challenges [Csikszentmihalyi 1991]. This paper attempts to add richness to the discussion, by introducing a flow-like mathematical model that supposes there is more than just skill and challenge feeding into the changing experience of motivation in a student.

This work resonates well with the general project of dynamical systems research in the social and cognitive sciences [Vallacher and Nowak 2007; Ward 2002], by providing a tangible example of a dynamical system. However, unlike prior work, it appears to be more focused on a generalizable model with educational implications. In the one instance we were able to find of an educationally relevant dynamical model that was fit to data [Guastello et al. 1999], the authors chose to fit the system to each user, a procedure that produces a distribution of dynamical systems. While this procedure produces interesting results, it is more difficult to use for making general predictions.

In this paper, we data mine for our general dynamical system in a dataset from a Chinese vocabulary tutoring system. This task is superficially straightforward, because students see an “optimal” schedule of drill practice (test with a review if incorrect) and have to respond to each drill practice by typing the matching Chinese pinyin phonetic representation or English meaning. Additionally, because we were pursuing the hypothesis that strategies and motivation might influence this practice, we asked three

Authors' addresses: P. Pavlik Jr, Human Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, PA, USA. Email: ppavlik@andrew.cmu.edu; S. Wu, Carnegie Mellon University, Pittsburgh, PA, USA. Email: suemei@andrew.cmu.edu

questions, which cycled for each user (using individual random orders for each subject, i.e. Q1, Q2, Q3 or Q2, Q1, Q3, etc....). One of these questions popped up every 2 minutes during the flashcards, and required only a single key response to minimize student inconvenience. These questions were as follows:

- “How easy was the recent practice?”, with Likert like items ranging from “too hard” to “too easy”, on a 5 point scale.
- “How useful for learning was the recent practice?”, with Likert like items ranging from “not useful” to “very useful”, on a 5-point scale.
- “Were you able to use any learning strategies during the recent practice?” with Likert like items ranging from “mostly used repetition” to “mostly used strategies”, on a 5 point scale.

These questions were meant to be unthreatening to the student, since they are only admissions about recent practice, and so should not carry strong social norms for responding one way or another. These questions attempt to measure 3 crucial theoretical determinants of performance and motivation that have been suggested over the last 50 years in the literature on behavior, motivation, and metacognition.

2. DATA COLLECTION

The data was collected in the Elementary Chinese II course at Carnegie Mellon University (CMU). All data used to create the model have been archived in the Pittsburgh Science of Learning Center (PSLC) DataShop web application, and a copy of the model and data are available to registered users (free). The R code posted at this location functions by downloading the data using DataShop web services (requires a quick set-up) before finding the model using R functions as described below.

The data itself comes from Lessons 15-18 in Elementary Chinese II at Carnegie Mellon University, which correspond to chapters 15-18 in the *Chinese Link (Zhongwen Tiandi)* textbook [Wu et al. 2006]. The intervention was a computerized adaptive practice drill system with several different types of Chinese vocabulary flashcards including practice of the pinyin phonetic system, English meaning, radical character components of Chinese, and flashcards where the student filled in a missing vocabulary word in a sentence (Cloze fill in the blank) [Pavlik Jr. et al. 2008]. We had 65 unique subjects (of which only 61 produced complete practice data) and 198 unique lesson runs. Average lesson-run length was about 20 minutes (averaging 9.34 blocks of 2 minutes of practice per lesson started). Students were allowed to take breaks (by hitting a pause button) at any time, the dynamical systems model ignores any breaks students took and treats each lesson as continuous. Since students were asked to practice for only 20 minutes, many students completed the work without any break. While there was a between-subject manipulation of initial practice difficulty (either wider or narrower distribution of initial practice), this did not appear to cause detectable effects, though it may have produced more varying data, thus providing more variability for the model to capture, improving the detection of patterns by encouraging the occurrence of these patterns in the data.

3. DYNAMICAL SYSTEM MODEL

After doing some preliminary correlations of the t and t-1 data values (using only one predictor and one predicted variable) to make sure there were significant relationships between the variables, the data was organized into a set of data points for each user for each lesson. There were 4 data vectors per student per lesson, including vectors for easiness, usefulness, strategy use, and performance. Since each of the 3 survey items was measured sparsely, about 2/3 of the data values in each survey vector were left empty. Probability correct for the previous 2 minutes (previous epoch) however, was always

calculable, so these vectors were not sparse. Further, since probability (performance) is a 0 to 1 value (tending to be greater than 50% in the data), whereas our readings from the other measures varied from 1-5, we scaled probability by using the logit of probability as data (constrained so that $\text{logit} < -5 = -5$ and $\text{logit} > 5 = 5$). For this reason, the model predicts logit values for performance, and the simulator code converts these to probabilities for the simulation graph plotting. Because the model is undefined unless it has a full set of 4 prior values to begin computations, we used the neutral values of 3 easiness, 3 usefulness and 3 strategy use to initialize the model for each lesson for each user.

A dynamical systems model assumes some current state of nature and then describes an evolution rule about how each state is transformed to the future state. Assuming the evolution rule is correct; by iteration, we can predict the future states of nature given some start state. Our evolution rule was

$$\begin{aligned} e_{t+1} &= x_1 e_t + x_2 e_t^2 + x_3 u_t + x_4 u_t^2 + x_5 s_t + x_6 s_t^2 + x_7 p_t + x_8 p_t^2 + \\ &\quad x_9 e_t u_t + x_{10} e_t s_t + x_{11} e_t p_t + x_{12} u_t s_t + x_{13} u_t p_t + x_{14} s_t p_t + x_{15} \\ u_{t+1} &= x_{16} e_t + x_{17} e_t^2 + x_{18} u_t + x_{19} u_t^2 + x_{20} s_t + x_{21} s_t^2 + x_{22} p_t + x_{23} p_t^2 + \\ &\quad x_{24} e_t u_t + x_{25} e_t s_t + x_{26} e_t p_t + x_{27} u_t s_t + x_{28} u_t p_t + x_{29} s_t p_t + x_{30} \\ s_{t+1} &= x_{31} e_t + x_{32} e_t^2 + x_{33} u_t + x_{34} u_t^2 + x_{35} s_t + x_{36} s_t^2 + x_{37} p_t + x_{38} p_t^2 + \\ &\quad x_{39} e_t u_t + x_{40} e_t s_t + x_{41} e_t p_t + x_{42} u_t s_t + x_{43} u_t p_t + x_{44} s_t p_t + x_{45} \\ p_{t+1} &= x_{46} e_t + x_{47} e_t^2 + x_{48} u_t + x_{49} u_t^2 + x_{50} s_t + x_{51} s_t^2 + x_{52} p_t + x_{53} p_t^2 + \\ &\quad x_{54} e_t u_t + x_{55} e_t s_t + x_{56} e_t p_t + x_{57} u_t s_t + x_{58} u_t p_t + x_{59} s_t p_t + x_{60} \end{aligned}$$

where t – Indexes the observation time epoch ($t+1$ values are predicted values)

e – Likert-like self-report on recent easiness

u – Likert-like self-report on recent usefulness

s – Likert-like self-report on recent strategy use

p – logit of probability correct since previous self-report (max 5 min -5)

This rule was applied for each of the 4 state space variables to compute the next epoch prediction. The rule for each variable required 15 parameters to capture the 4 first order (linear) effects, 10 second order (quadratic) effects, and the single fixed (intercept) effect. This meant we had 60 parameters for the non-linear deterministic model.

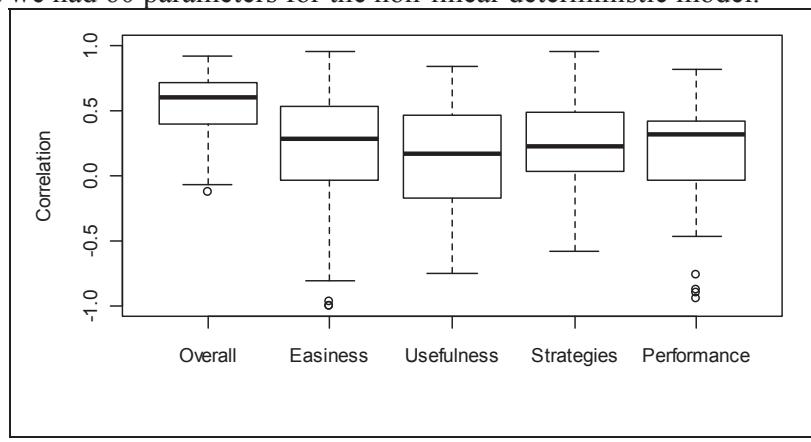


Fig. 1. Boxplots of correlations between model and data for each individual student, for the overall model and by construct ($N=61$ students).

The model parameters were found by fitting the model to the data by minimizing the sum of squared error. To fit the model we used the BFGS optimizer implementation in the R “optim” command. Because there was not sufficient time to complete some sort of model validation, these results should be considered preliminary. However, to provide some support for validity, Figure 1 shows boxplots of the correlations of the model and data for each individual student, for each of the 4 variables. These correlations show the model fit is not achieved by fitting individual student magnitudes, but rather the model is capturing a general overall model of the dynamical system within the bulk of students. In

particular, it is encouraging that only 2 students negatively correlated with the overall model. It is also clear the model is not succeeding by just fitting one or some of the constructs while neglecting others. In fact, we see the overall pattern is better modeled than any individual construct.

5. SIMULATIONS

We wanted to use the model as a simulation tool to show us how to understand the student experiences in the tutor and how these experiences are reflected in the constructs we tracked.

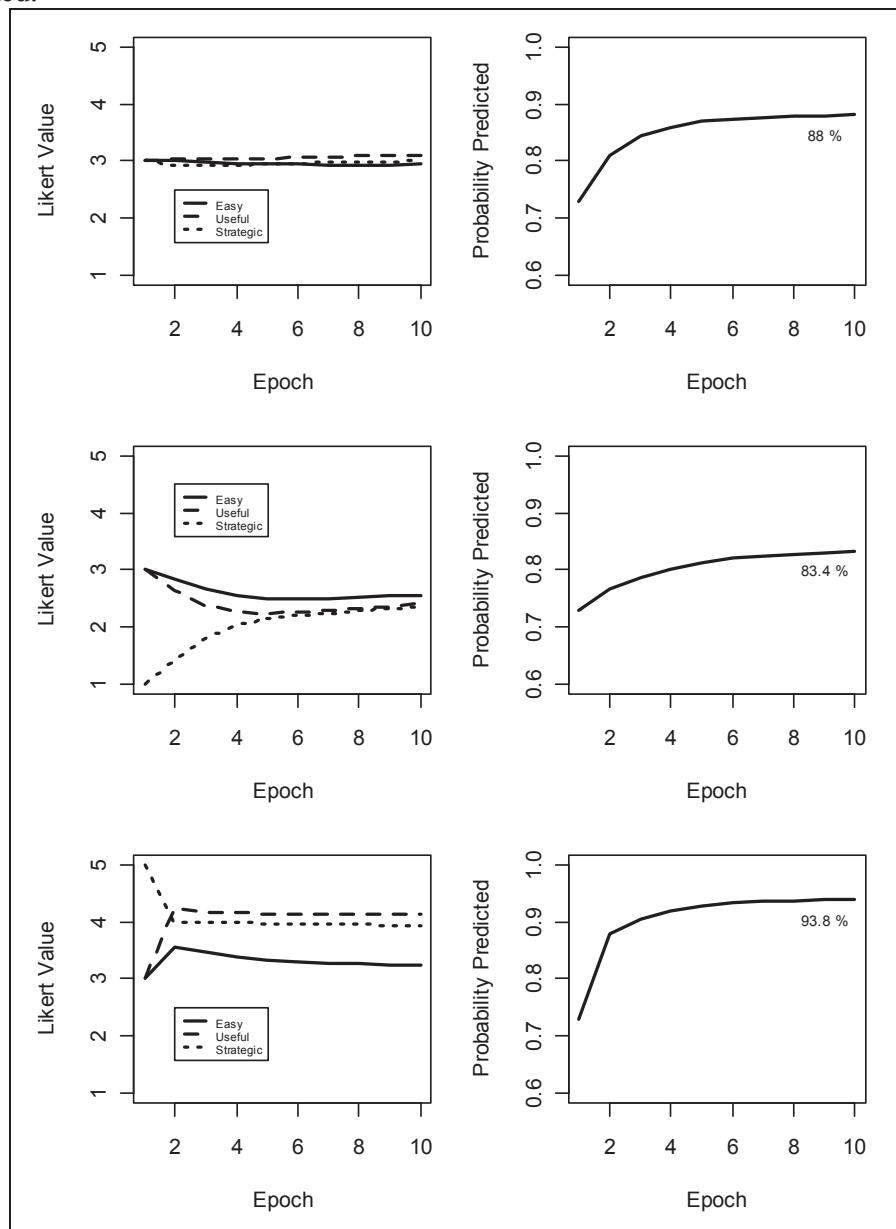


Fig. 2. Pinned simulations of an average student, a rote learner, and a strategic learner.

To begin, Figure 2 top shows a pinned (initialized at a certain value) simulation starting from values very close to average (we used 3,3,3 for simplicity and because these were clearly the medians of each distribution). For this example, the model was initialized with 3 easiness, 3 usefulness, 3 strategy-use, and a starting performance of 73% (equivalent to a logit of 1, which was almost exactly the true average). Figure 2 top demonstrates this

“average” model reaches a performance level of 88.0% after 20 minutes (10 epochs) of practice, and suggests visually this level of performance has reached a stable plateau.

Figure 2 middle for a rote learner contrasts with Figure 2 bottom, for a strategic learner, each pinned to begin at the minimum and maximum strategy values. This strategic simulated student has a much different experience and we can see after 10 epochs, it is already becoming clear they have arrived at a different steady state with strategy use and usefulness ratings stabilizing at about 4. Interestingly, easiness is not improved, in large part because (according to our parameter analysis, not shown) high strategy use and high usefulness combined are predicted to reduce easiness. This makes sense if we suppose producing strategies in a useful fashion might be difficult for the student, even if effective. This effectiveness is highlighted by the reduced error rate in this steady state high strategy model. Comparing the error rates ($1-.938=.062$) for pinned high and pinned low ($1-.834=.166$) reveals pinning to the low strategy for the first epoch leads to the prediction of $.166/.062=2.7$ times more errors during the epoch 16 minutes later. Because errors greatly reduce practice efficiency in flashcard type learning [Pavlik Jr. and Anderson 2008], such an effect shows how strategy use may be an important component of optimal practice.

6. CONCLUSIONS

The main contribution of this work is to provide a general model to help us understand the dynamic motivational system of students. The dynamical systems model we produced is valuable because of what it implies for optimal use of the flashcard system and optimal learning in general. As we showed in the simulations, the model provides evidence of the mutually supportive feedback between usefulness, strategy use, and performance. By observing the simulations we see how high strategy use has a powerful effect on students’ perception of usefulness, but this is contrasted by the different behaviour of easiness ratings, which are depressed when strategies and usefulness are high. This seems to be a natural result of people finding it less easy to use strategies despite the acknowledgment of their usefulness for learning.

ACKNOWLEDGEMENTS

This research was supported in part by a grant from the Pittsburgh Science of Learning Center which is funded by the National Science Foundation award number SBE-0836012 and Ronald Zdrojkowski for educational research.

REFERENCES

- CSIKSZENTMIHALYI, M. 1991. *Flow: The Psychology of Optimal Experience*. Harper Perennial.
- GUASTELLO, S.J., JOHNSON, E.A. AND RIEKE, M.L. 1999. Nonlinear Dynamics of Motivational Flow. *Nonlinear Dynamics, Psychology, and Life Sciences* 3, 259-273.
- PAVLIK JR., P.I. AND ANDERSON, J.R. 2008. Using a model to compute the optimal schedule of practice. *Journal of Experimental Psychology: Applied* 14, 101-117.
- PAVLIK JR., P.I., PRESSON, N. AND HORA, D. 2008. Using the FaCT System (Fact and Concept Training System) for Classroom and Laboratory Experiments. In *Inter-Science Of Learning Center Conference*, Pittsburgh, PA.
- VALLACHER, R.R. AND NOWAK, A. 2007. Dynamical social psychology: Finding order in the flow of human experience. In *Social Psychology: Handbook of Basic Principles* Conference Name, A.W. KRUGLANSKI AND E.T. HIGGINS Eds. Guildford Publications, New York.
- WARD, L.M. 2002. *Dynamical cognitive science*. MIT Press, Cambridge, Mass.
- WITHERSPOON, A., AZEVEDO, R., GREENE, J., MOOS, D. AND BAKER, S. 2007. The Dynamic Nature of Self-Regulatory Behavior in Self-Regulated Learning and Externally-Regulated Learning Episodes. In *Proceedings of the 13th International Conference on Artificial Intelligence in Education: Building Technology Rich Learning Contexts That Work* Conference Name, R. LUCKIN AND K.R. KOEDINGER Eds. IOS Press, 179-186.
- WU, S.-M., YU, Y. AND ZHANG, Y. 2006. *Chinese Link: Zhongwen Tiandi, Intermediate Chinese*. Pearson Education/ Prentice Hall.

Desperately Seeking Subscripts: Towards Automated Model Parameterization

J. MOSTOW

Y. XU

M. MUNNA

Carnegie Mellon University, USA

This paper addresses the laborious task of specifying parameters within a given model of student learning. For example, should the model treat the probability of forgetting a skill as a theory-determined constant? As a single empirical parameter to fit to data? As a separate parameter for each student, or for each skill? We propose a generic framework to represent and mechanize this decision process as a heuristic search through a space of alternative parameterizations. Even partial automation of this search could ease researchers' burden of developing models by hand. To test the framework's generality, we apply it to two modeling formalisms – a dynamic Bayes net and learning decomposition – and compare how well they model the growth of children's oral reading fluency.

Key Words and Phrases: model parameters, heuristic search, knowledge tracing, learning decomposition, children's reading fluency growth

1. INTRODUCTION

This paper addresses the problem of defining parameters, more precisely how specific to make them. For example, the parameters in a knowledge tracing model are the probabilities of already knowing a skill, learning it from a practice opportunity, guessing an answer without knowing the skill, or answering incorrectly despite knowing the skill. But how specifically should these parameters be defined? Should we use a different parameter for every skill? For every student? For every \langle student, skill \rangle pair? The last option would generate too many parameters to fit from the available data. Corbett et al. [Corbett and Anderson, 1995] decided to make the knowledge parameters (probabilities of knowing already or learning) skill-specific, and the performance parameters (probabilities of guessing or slipping) student-specific. They judged that the knowledge probabilities vary more by skill than by student, whereas the performance probabilities vary more by student than by skill.

Such decisions – how specific to make a given parameter in order to predict unseen data – are the focus of this paper. This subtle but crucial modeling decision is typically made by hand, often by trial and error. The researcher explores various alternatives, trading off theoretical plausibility, computational tractability, model fit, statistical reliability, interpretability, and informativeness with respect to the research questions of interest. This problem falls in the domain of model selection but differs from prior work on selecting structure [e.g., Madigan and Raftery, 1994] or variables [e.g., Negrin et al., 2010] in that we focus on selecting a specific parameterization of the given variables.

We propose a generic framework to represent and mechanize this process. To test its generality, we apply it to two types of student learning models (dynamic Bayes nets and learning decomposition), which we train and test on children's oral reading fluency data.

Authors' addresses: J. Mostow, E-mail: mostow@cs.cmu.edu; Y. Xu, E-mail: yanbox@cs.cmu.edu; M. Munna, E-mail: mmunna@cs.cmu.edu. Project LISTEN, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, U.S. 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.

2. A HEURISTIC SEARCH SPACE OF MODEL PARAMETERIZATIONS

The title of this paper refers to model development as a search for subscripts because subscripts indicate the specificity of the parameters they index. To formalize this search space, we represent each state in the space as a vector with an element for each parameter in the model. For example, consider a dynamic Bayes net model of Knowledge Tracing (KT), with probabilities for *guess*, *slip*, *forget*, *learn*, and *already know*. We represent a parameterization of this model as a vector of 5 elements, each of which specifies how the corresponding parameter is subscripted. For readability, we write the value of each element as a phrase describing how the parameter is indexed, e.g. ‘*by student*’, ‘*by skill*’, ‘*by student level*’.

Formally, we define a **parameterization** of a model with m parameters p_1, p_2, \dots, p_m as a vector of m split functions (F_1, F_2, \dots, F_m), each of which specifies how to index the corresponding parameter over a set of size N , which we call the *size* of the split. For example, to fit the *guess*, *slip*, and *learn* parameters of a KT model separately for each student, we use the ‘*by student*’ function to split them into separate parameters $guess_j$, $slip_j$, and $learn_j$ for each student j , so its size is the number of students. Likewise, to fit the *already know* parameter separately to the data for each skill, we use the ‘*by skill*’ function to split it into separate $already\ know_i$ parameters for each skill i , so its size is the number of skills.

To set a parameter to a single value for all of the data, we use a function named “*by NULL*” to leave the parameter as is, with no subscripts or splits. We may estimate its empirical value by fitting the data, or supply a theoretical constant. For example, for a KT model, we apply the “*by NULL*” function to the *forget* parameter, and set its value to zero based on the theoretical assumption of no forgetting.

We define the size of a parameterization as the summed sizes of its split functions. Intuitively, this quantity is simply the total number of subscripted parameters. The example parameterization above indexes three parameters by student and one by skill, so its size is $3 * \# \text{ students} + 1 * \# \text{ skills}$.

Given m parameters p_1, p_2, \dots, p_m and a set F of split functions, the cross product F^m generates a search space of $|F|^m$ possible model parameterizations to consider. One simple but inefficient search strategy is brute force, searching for the best model over all expressible splits. Alternatively, one heuristic strategy is to search the space of parameterizations in order of increasing size, fitting the resulting parameterized model to the data, computing some measure of its (complexity penalized) model fit, and halting when we reach a local maximum. Note that the size of the parameterization is a crude measure of model complexity.

3. TWO DIFFERENT MODEL FORMALISMS

Dynamic Bayes nets (DBNs) provide a powerful way to infer a student’s changing knowledge over time from observed student behavior. We extended a previous DBN model of children’s fluency growth [Beck et al., 2008] by adding an observable “Distributed Practice” node whose value is 1 for the student’s first encounter of the day for a given word and 0 otherwise. The resulting model (shown in Fig. 1) has 17 parameters, too many to list here. For example, the parameter “*learn | distributed practice, help*” models the probability $P(K_n = \text{true} | K_{n-1} = \text{false}, D_{n-1} = \text{true}, H_{n-1} = \text{true})$. We used BNT-SM [Chang et al., 2006] to express different parameterizations of the model and fit them to data.

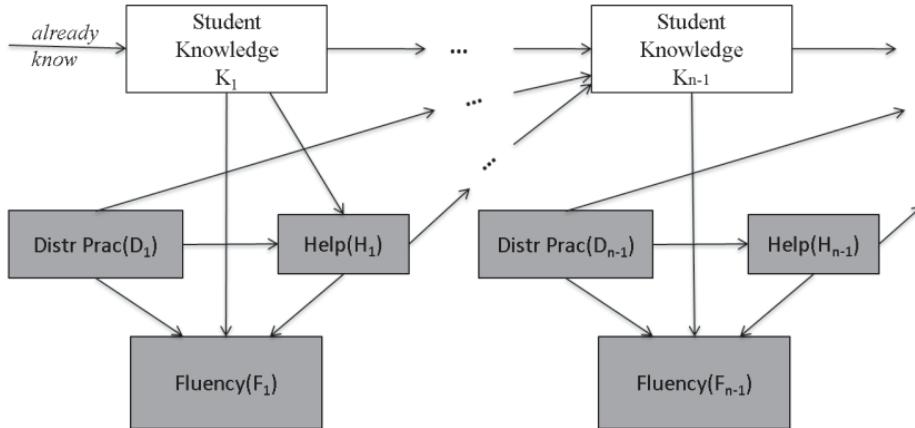


Fig. 1. Architecture of a Bayes Net Model of Children’s Growth in Oral Reading Fluency

Learning decomposition (LD) estimates the relative impact on performance of different types of practice, such as wide vs. repeated reading and distributed vs. massed practice [Beck, 2006]. Using this approach, we developed the following model to predict a child’s latency prior to reading a word aloud in text:

$$\text{latency} = E + L * \text{word_length} + A * e^{-b * (h * m * HM + h * HD + m * NHM + NHD)}$$

Here E represents minimum latency, L scales latency as a linear function of word length, A reflects the latency at the first encounter of a word, and b represents the learning rate. The coefficient h represents the impact of a tutor-assisted encounter relative to an unassisted encounter. The coefficient m represents the impact of a massed encounter (i.e. of a word seen earlier that day) relative to a distributed encounter (i.e. of a word seen for the first time that day). The variable HM counts the number of assisted, massed encounters; HD counts assisted, distributed encounters; NHM counts unassisted, massed encounters; and NHD counts unassisted, distributed encounters. To fit different parameterizations of this model to data, we used MATLAB’s (Ver. 7.6.0.324) non-linear regression function.

4. EVALUATION

4.1 Data

The oral reading fluency data for this paper comes from a random sample of 40 children, stratified by gender and reading level, from the students who used Project LISTEN’s Reading Tutor [Mostow and Aist, 2001] during the 2005-2006 school year, with a median usage of 5.7 hours. In total they attempted to read 5,078 distinct word types ranging in difficulty level from grades 1 to 11. The data includes each student’s unique user id, gender, reading level (from grade K to 6), and performance on each word encounter, which we define as fluent if accepted by the Reading Tutor as read correctly without help or hesitation.

To partition the data into training and test sets, we ordered the distinct word types encountered by each student by the number of encounters. We assigned all the student’s encounters of odd-numbered word types to the training set, and all encounters of even-numbered word types to the test set, so as to be able to train and test models on all of a student’s encounters of a given word.

Given the information in the data set, one set of possible splits is {‘by student’, ‘by student level’, ‘by gender’, ‘by word’, ‘by word level’, ‘by student and word level’, ‘by

student level and word', '*by student level and word level*', '*by gender and word*', '*by gender and word level*'}. We omitted the split '*by student and word*' because we had no overlap in <student, word> pairs between training and test sets.

4.2 Results

Table I compares different parameterizations of DBN and LD models, ordered by size. The DBN models treat fluency as a binary variable, so we show the percentage accuracy of their predictions, both overall and within-class; the test data is unbalanced, with 72% of it in the positive (fluent) class. The LD models predict real-valued latencies, so we use Root Mean Squared Error (RMSE) to measure their accuracy. Since the models make different types of predictions, their accuracies are not comparable. Given the maximized value L of the likelihood function for the estimated model, the number k of parameters and the number n of data points in the training set, we compute AIC (Akaike Information Criterion) as $AIC = 2k - 2 \ln L$. We estimate BIC (Bayesian Information Criterion) as $BIC \approx -2 \cdot \ln L + k \ln(n)$.

DBN and LD models use different likelihood functions. The likelihood function for DBN models is a probability, so their AIC and BIC scores are positive. In contrast, the likelihood function for a linear regression is a product of Gaussian probability density functions, so AIC and BIC scores for LD models can be positive or negative.

Table 1. Accuracy and complexity of DBN and LD models on unseen test data for children's oral reading fluency. The best value(s) in each column are underlined.

Model: split by...	Size	DBN				BIC	Size	RMSE (sec)	LD	
		Acc (%)	Acc on +	Acc on -	AIC				AIC	BIC
<i>NULL</i>	17	72.5	<u>99.7</u>	2.4	504084	504243	6	0.22	-9056	-9000
<i>gender</i>	34	72.5	<u>99.7</u>	2.4	504060	<u>504377</u>	12	0.22	-9269	-9157
<i>student level</i>	136	72.5	<u>99.7</u>	2.4	897462	898733	42	0.23	46	427
<i>word level</i>	170	72.5	<u>99.7</u>	2.5	474230	<u>475818</u>	48	0.35	-10201	-9755
<i>gender, word level</i>	323	72.5	<u>99.7</u>	2.5	474182	477200	96	0.32	-36957	-36048
<i>student</i>	680	<u>72.6</u>	98.1	7.1	497074	503427	210	0.21	-22483	-20546
<i>student level, word level</i>	1054	72.5	98.1	6.5	<u>470057</u>	479905	318	0.28	25977	28692
<i>student, word level</i>	4573	71.8	94.1	14.2	473541	516270	1512	<u>0.18</u>	<u>-144840</u>	<u>-130160</u>
<i>word</i>	5848	72.4	96.0	1.2	495727	550370	1518	0.20	-9136	4715
<i>gender, word</i>	11271	72.5	93.7	<u>15.2</u>	550977	847543	2856	<u>0.18</u>	-32541	-5762
<i>student level, word</i>	31739	71.8	98.1	6.5	512257	617572	3588	0.21	15354	45053

Which models are best? None of the DBN models substantially beats the majority class accuracy of 72%. The five simplest models have almost perfect recall (accuracy on positive examples), but very low accuracy on negative examples. Note that AIC and BIC do not vary smoothly with the size of the parameterization. For example, splitting by student level has size 136 and gives the worst AIC and BIC scores, while word level has size 170 but yields the best BIC score and a near-best AIC score.

The '*by student and word level*' LD model has the lowest AIC and BIC scores. This fact suggests that students at the same estimated student level differ enough to model

individually, possibly due to inaccurate estimates. In contrast, word level apparently captures adequate information about word difficulty.

This model also achieves the best accuracy on unseen test data (RMSE = 0.18 sec). However, the second best accuracy is achieved by '*by word*' model, which has some of the worst AIC and BIC scores even though its size is not enormously larger (1518 vs. 1512). This disparity implies that AIC and BIC can be poor predictors of performance on unseen data. One problem we faced that due to splitting when the dataset size was very small (e.g. less than 4 data points) we failed to fit the LD model. We excluded these datasets and the size of parameterization became smaller than it should be in some of the models.

Although the DBN and LD models have different formalisms and outputs, they are not directly comparable, we can still compare their performance profiles over the same space of parameterizations. In particular, is the same parameterization best for both models? No. For the LD models, the '*by student, word level*' parameterization achieves by far the best AIC and BIC scores. For the DBN models, this parameterization achieves close to the best AIC score, which is for the '*by student level and word level*' model, but so do the '*by word level*' and '*by gender and word level*' models. Moreover, its BIC score is mediocre.

5. CONCLUSION

This paper defines the problem of parameterization selection and formalizes it in terms of a space of parameterizations induced by split functions. It proposes a simple strategy to search this space in order of size, hill-climbing on complexity-penalized model fit. We implemented a prototype of this strategy restricted by using the same split function for every parameter to accommodate a limitation of BNT-SM. We demonstrated its generality by applying it to both DBN and LD models and evaluating the resulting parameterizations on the same data set.

Future work includes expanding the search space to relax the restriction in the implementation, and devising search heuristics to go beyond size and complexity-penalized model fit and address additional criteria discussed in the Introduction. This work will succeed if it helps clarify, accelerate, or automate the discovery of good models.

REFERENCES

- BECK, J.E. 2006. Using learning decomposition to analyze student fluency development. In *ITS2006 Educational Data Mining Workshop*, Jhongli, Taiwan, June 26, 2006, C. HEINER, R. BAKER and K. YACEF, Eds., 21-28.
- BECK, J.E., CHANG, K.-M., MOSTOW, J. and CORBETT, A. 2008. Does help help? Introducing the Bayesian Evaluation and Assessment methodology. In *9th International Conference on Intelligent Tutoring Systems*, Montreal, June 23-27, 2008, 383-394. ITS2008 Best Paper Award.
- CHANG, K.-M., BECK, J., MOSTOW, J. and CORBETT, A. 2006. A Bayes Net Toolkit for Student Modeling in Intelligent Tutoring Systems. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, Jhongli, Taiwan, June 26-30, 2006, K. ASHLEY and M. IKEDA, Eds., 104-113.
- CORBETT, A. and ANDERSON, J. 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction* 4, 253-278.
- MADIGAN, D.M. and RAFTERY, A.E. 1994. Model selection and accounting for model uncertainty in graphical models using Occam's Window. *Journal of the American Statistical Association* 89, 1335-1346.
- MOSTOW, J. and AIST, G. 2001. Evaluating tutors that listen: An overview of Project LISTEN. In *Smart Machines in Education*, K. FORBUS and P. FELTOVICH, Eds. MIT/AAAI Press, Menlo Park, CA, 169-234.
- NEGRIN, M.A., VAZQUEZ-POLO, F.J., MARTEL, M., MORENO, E. and GIRON, F.J. 2010. Bayesian Variable Selection in Cost-Effectiveness Analysis. *International Journal of Environmental Research and Public Health* 7, 1577-1596.

Automatic Generation of Proof Problems in Deductive Logic

B. MOSTAFAVI, T. BARNES, AND M. CROY

University of North Carolina at Charlotte, United States

Automatic problem generation for learning tools can provide the required quantity and variation of problems necessary for an intelligent tutoring system. However, this requires an understanding of problem difficulty and corresponding features of student performance. Our goal is to automatically generate new proof problems in Deep Thought – an online propositional logic learning tool – for individual students based on their performance and a set of instructor parameters.. In an initial exploratory study, we evaluate the generated problems compared to the original set of stored problems. This evaluation uses collected student data and instructor feedback.

Key Words and Phrases: Logic proof, problem generation, problem difficulty, intelligent tutoring system

1. INTRODUCTION AND RELATED WORK

Intelligent tutoring systems (ITS) provide adaptive instruction to students, and have a significant effect on learning [Murray, 1999]. Much of the work in ITS development has been in generating feedback and hints for existing problems, such as CTAT, an example-based authoring tool [Koedinger et al, 2004]. Logic-based tutors such as Logic-ITA support the learning and teaching of logic proofs, verifying proof statements, providing feedback, and logging data for exploration [Lesta and Yacef, 2004] [Yacef, 2005].

Our focus, however, is on automatic problem generation. We present our work in the context of a logic proof tutor, called Deep Thought [Croy, Barnes and Stamper, 2008]. In Deep Thought, students construct proofs by applying logical rules to a set of given premises in order to generate a specific conclusion. Our long-term goal is to provide instantly generated proof problems that are appropriate based on the student's skill level, course progression, and previous performance on Deep Thought problems.

McGough et al [2001] created a web-based dynamic problem generation system in engineering; however, their problems are assembled from a pool of existing problem subsets, while we seek to generate logic proofs from scratch. Beck et al [1997] accomplished this for a tutor in arithmetic operations, which is similar to what we wish to accomplish using logic proofs. However, that tutor took into account several assumptions about a student's basis of knowledge in basic arithmetic, which are not as well defined with logic proof construction.

2. THE AUTOMATIC PROBLEM GENERATOR

Deep Thought is an existing web-based tool with a graphical user interface that provides a set of problems that display logical premises, buttons for logical rules (axioms), and a conclusion that a student must reach by applying logical rules to the premises (Figure 1). Student progress is logged at each step of proof construction, recording attributes including rule use, errors, deletions, time, and successful completion of the problem.

Authors' addresses: B. Mostafavi, Department of Computer Science, University of North Carolina at Charlotte, United States. E-mail: bzmostaf@uncc.edu; T. Barnes, Department of Computer Science, University of North Carolina at Charlotte, University of North Carolina at Charlotte; E-mail: Tiffany.Barnes@uncc.edu; M. Croy, Department of Philosophy, University of North Carolina at Charlotte, United States. E-mail: mjcroy@uncc.edu

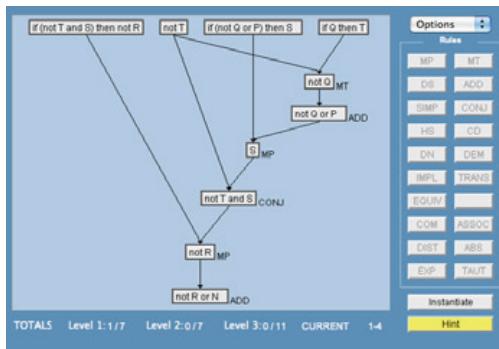


Figure 1. Deep Thought user interface, showing a successfully completed problem (English mode).

We have developed a java-based background process to Deep Thought called LQGen (Logic Question Generator) that automatically generates proof problems that satisfy the conceptual requirements of the course instructor. LQGen takes as input the parameters of the desired problem, and generates a new random problem. The input parameters are extracted from the expert-derived solution of the original problems, and include the number of premises, re-used premises, logical rule use, number of steps, and complexity of statements and conclusion (see Table 1).

LQGen generates problems by working backwards [Croy, 2000], starting with the problem conclusion to generate subsequent steps until a full problem is developed. After creating a random conclusion, it builds a tree of logical statements to a depth equal to the number of steps in the expert solution, based on the parameter requirements, then traverses the tree and deletes branches until the specified number of premises is reached.

In its current state, LQGen generates problems that match the parameters of the expert-solved original problems. However, the parameters do not adapt to an individual student's skill level and performance. Before we can accomplish this, we must first understand how students might behave in constructing logic proofs by examining the features that might determine a problem's difficulty, and compare this behavior between original Deep Thought problems and problems generated by LQGen.

3. MEASURING PROBLEM DIFFICULTY

If we accept the general idea that certain logical concepts are more difficult than others, we can assume that the parameters we use for construction of proofs may be sufficient for generation of problems that have the same conceptual difficulty. However, we need to look at student data to determine how student performance indicates their level of knowledge, and therefore problem difficulty. Beck and colleagues [1997] look at historical student data to determine the rate at which an arithmetic concept is learned, and how often a student continues to use the concept. However, in their tutor, the knowledge gained by the students and the method of proficiency demonstration is linear, which is not the case with logic proofs, as students can solve problems in various ways.

There are several factors we can consider in determining problem difficulty. We can consider the failure rate in the attempts made by the students per problem, and the usage of rules as expected from the expert solution. If a student has a decreasing rate of failure and higher usage of rules for problems of similar type, we can assume the student is performing at or near the level of difficulty set by the course instructors. We can also look at the number of performance errors, any reworking of the problem, the number of rule applications, and the elapsed time. We would expect an exponential learning curve in the factors above when students are working on problems with similar concepts.

Once we determine how these factors affect student performance from the student data, we can determine how to alter the parameters of the problems created by LQGen to give students practice and mastery of the concepts taught before generating problems of higher difficulty.

4. METHODS AND RESULTS

Data are from students solving Deep Thought Level 1 problems as homework in a Deductive Logic course. Eighty Fall 2010 students solved instructor-authored problems, and eighty Spring 2011 students solved LQGen problems designed to match the Fall problems. Table 1 shows the parameters of the six problems as solved by an expert, using inference rules including modus ponens (MP), disjunctive syllogism (DS), simplification (SIMP), hypothetical syllogism (HS), modus tollens (MT), addition (ADD), conjunction (CONJ), and constructive dilemma (CD). The problem set was designed so that students would use all of the inference rules over the set. We expected problem 4 to be more difficult since it is the first to use CONJ, and it requires the re-use of a given premise. Problem 5 was difficult for the same reasons plus it also introduced CD.

Table 1. Deep Thought Level 1 Parameters as determined by expert solutions

Problem	Premises	# Rules Used	Rules used by expert								Premise Reuse
			MP	DS	SIMP	HS	MT	ADD	CONJ	CD	
1.1	3	4	1	1	2	0	0	0	0	0	No
1.2	3	4	1	1	1	0	0	1	0	0	No
1.3	3	4	1	0	1	0	1	1	0	0	No
1.4	4	6	2	0	0	0	1	2	1*	0	Yes*
1.5	4	6	1	1	2	0	0	0	1	1*	Yes*
1.6	4	4	1	1	1	1	0	0	0	0	No

Figure 2 shows the percent of incomplete (failed) attempts, with a learning curve decreasing from problems 1.1 to 1.3 and then again from 1.4 to 1.6, consistent for both original and generated problems.

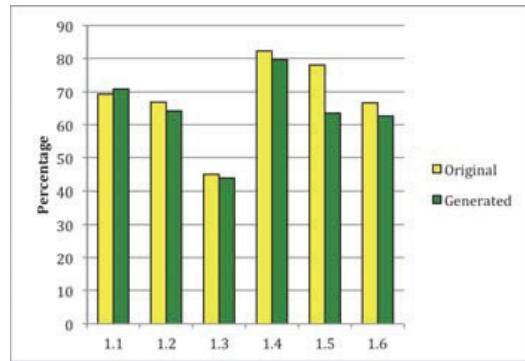


Figure 2. Average percent of incomplete attempts for Deep Thought Level 1 problems

Figure 3 shows that the number of attempts for each problem has a reasonable learning curve from 1.1-1.3 and 1.4-1.6 (with low attempts for 1.2 and 1.3 since they were not required). Figure 4 shows the average time taken per problem attempt. The generated problem 1.4 has more attempts and more time per attempt, showing that it is likely more difficult than the original problem, while generated 1.5 has fewer attempts and much shorter times. One explanation for fewer and shorter tries on 1.5 is that after solving the harder generated 1.4 problem, students had less trouble with 1.5. However, upon investigation, we also found that while both 1.5 problems needed constructive dilemma (CD), the original 1.5 problem needed it much later in the problem so students had to determine how to set up the problem to apply CD, while in the generated problem,

it was apparent to the students that CD could be applied directly to the premises. The number of attempts is greater for generated problems in 4 of the 6 problems, and on the remaining two the number of attempts is very similar. This suggests that the Spring students attempted the generated problems more times, but spent less time on each attempt, than the Fall students tried the original problems.

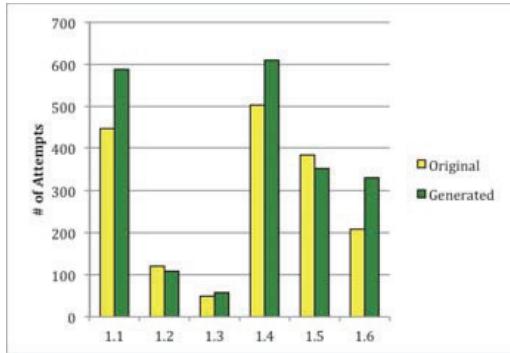


Figure 3: Total attempts (1.2 & 1.3 optional)

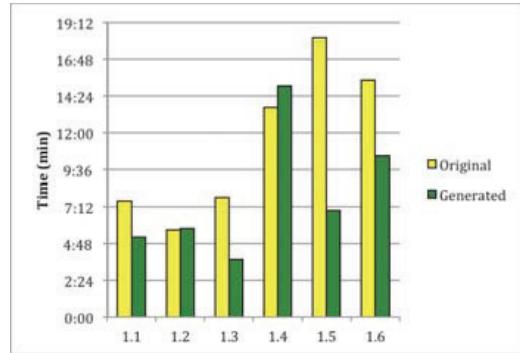


Figure 4. Average time taken per attempt

At first we expected that spring students, who had more short attempts, realized when an attempt was not productive and simply started over, but our analysis of problem length suggests otherwise. Figure 5 shows the total number of errors, deletions, and steps in student attempts. All the generated problems have more steps, therefore students reworked the generated problems more often, with longer proofs, but in shorter average times, than students did with the original problems. We believe this suggests that the generated problems needed longer time, but were not quite as difficult proofs.

Although students spent more time on each attempt for generated problems, they performed slightly more deletions on 1.1, 1.2, and 1.4 and made more errors for 1.2, 1.3, 1.4, and 1.6. This may indicate that students in Spring 2011 are finding rule applications easier and can apply them in Deep Thought more quickly per step, and may be spending more time in constructing proofs and trying more strategies in the tutor.

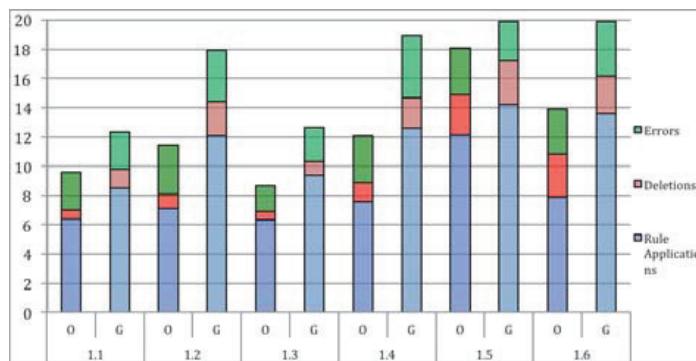


Figure 5. Average number of errors, deletions, and rule applications per attempt for original (O) and generated (G) Deep Thought Level 1 problems

Our expert took 4 - 6 steps on every original and generated problem, but as expected, students applied more rules on average per attempt for both problem sets, as is particularly apparent in Table 2. For problem 1.1 students replaced a use of the SIMP rule with MP in LQGen. LQGen's problem 1.2 seemed to require one more application of DS and ADD for students, but problem 1.4 didn't need MP and MT as much as the original

problem. Students used SIMP a bit more in the LQGen 1.4 problem, and solved LQGen 1.6 with one less MP and HS rule each. Overall, both original and generated problems encouraged students to apply the rules experts used, and the numbers of times students applied the rules in correct solutions across the whole problem set shows a similar pattern to their usage by experts (though each rule is used more by students).

Table 2. Average rule use per successfully completed problem for Deep Thought Level 1

		MP	DS	SIMP	HS	MT	ADD	CONJ	CD		MP	DS	SIMP	HS	MT	ADD	CONJ	CD	
1.1	Expert	1	1	2						1.4	Expert	2				1	1	1	
	Orig.	1.73	1.78	2.5	.30	.06	.34	.33	.08		Orig.	2.02	.45	.63	.84	2.03	2.32	2.04	.31
	Gen.	2.11	1.02	1.25	.26	.06	.44	.21	.16		Gen.	1.15	.75	.99	.72	1.44	2.90	2.18	.48
1.2	Expert	1	1	1			1			1.5	Expert	1	1	2				1	1
	Orig.	1.83	1.78	2.18	.72		1.6	.32	.20		Orig.	2.17	2.38	2.45	.29	.29	.94	1.35	.95
	Gen.	1.76	2.61	2.5	.17	.12	2.46	.56	.10		Gen.	1.22	2.15	2.85	.32	.63	.85	1.57	.85
1.3	Expert	2		1		1	1			1.6	Expert	1	1	1	1				
	Orig.	1.37	.22	1.52	.07	1.22	1.29	.14	.06		Orig.	2.07	2.02	2.1	2.32	.81	.12	.42	.01
	Gen.	1.34	.09	1.37		1.25	1.53	.09	.12		Gen.	1.34	2.03	2.36	1.34	.85	.48	.48	.21
Total				MP	DS	SIMP	HS	MT	ADD	CONJ	CD								
		Expert		7	4		7	1	2	3		2	1						
		Orig.		11.2	8.6		11.1	4.5	4.4	6.6		4.6	1.6						
		Gen.		8.9	8.6		11.3	2.8	4.4	8.7		5.1	1.9						

5. CONCLUSION AND FUTURE WORK

We expected that, using parameters from expert solutions to logic proof problems, we could develop LQGen to generate similar problems for students to practice in deductive logic. Through the features compared in the above section, we believe that we have shown that LQGen can be used to provide practice problems that target a given rule set.

We plan to continue developing LQGen to support all the logical rules available in Deep Thought. We also plan to combine LQGen with a student model to assign problems that will encourage mastery of each rule needed to solve proofs. Once LQGen has been updated using knowledge gained from the study and integration of adaptive difficulty settings, it will be fully integrated into Deep Thought for its use in course instruction.

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. IIS 0845997.

REFERENCES

- BECK, J., STERN, M., AND WOOLF, B.P. 1997. Using the Student Model to Control Problem Difficulty. In Jameson, A., Paris, C., and Tasso, C. (Eds.) *User Modeling: UM97 Proceedings*, Springer, New York.
- CROY, M., BARNES, T., AND STAMPER, J. 2008. Towards an Intelligent Tutoring System for Propositional Proof Construction. In Briggle, A., Waelbers, K., and Brey, E. (Eds.) *Current Issues in Computing and Philosophy*, 145-155, IOS Press, Amsterdam, Netherlands.
- CROY, M. 2000. Problem Solving, Working Backwards, and Graphic Proof Representation. *Teaching Philosophy*, 23, 169-187.
- KOEDINGER, K., ALEVEN, V., HEFFERNAN, T., McLAREN, B., AND HOCKENBERRY, M. 2004. Opening the Door to Non-Programmers: Authoring Intelligent Tutor Behavior by Demonstration. In *ITS 2004 Proceedings*, 162-173.
- LESTA, L. AND YACEF, K. 2004. An Intelligent Teaching Assistant System for Logic. In *ITS 2004 Proceedings*, 421-431.
- MCGOUGH, J., MORTENSEN, J., JOHNSON, J., AND FADALI, S. 2001. A Web-Based Testing System with Dynamic Question Generation. In *FIE 2001 Proceedings*, 3, 23-28.
- MURRAY, T. 1999. Authoring Intelligent Tutoring Systems. In *IJAIED*, 10, 98-129.
- YACEF, K. 2005. The Logic-ITA in the Classroom: A Medium Scale Experiment. In *IJAIED*, 15, 41-62.

Comparison of Traditional Assessment with Dynamic Testing in a Tutoring System

MINGYU FENG

SRI International, USA

AND

NEIL T. HEFFERNAN

ZACHARY A. PARDOS

CRISTINA HEFFERNAN

Worcester Polytechnic Institute, USA

It would be great if ITS can also be used to do the benchmark assessment, so that no time from instruction is “stolen” to do extra assessments. It is presumed that given a limited amount of time for assessing, people should give a test but not wasting time giving students feedback. However, Feng and Heffernan (2010) compared two simulated conditions and found that the condition that lets students get feedback during a test was actually superior (not statistically reliable) at predicting student performance than the “test” condition, in which students did about double the number of problems. In this study, we address the weakness in Feng & Heffernan (2010) (i.e. simulated conditions) and run a new randomized control trial in a tutoring system with participants from two different grades, 7th and 8th to see if the main effect would replicate. Our results suggest that unlike our previous results 1) there is no reliable main effect across all students; 2) the dynamic testing condition works better with 7th graders than with 8th graders. We also find that 7th graders and 8th graders behaved differently while working within the system, which appeared to be related to “gaming”.

Key Words and Phrases: assessment, tutoring system, dynamic metrics

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. The accountability pressure from No Child Left Behind Act of 2001 in the U.S. has led to increased focus on benchmark assessments and practice tests on top of the usual end-of-chapter testing. Such practice assessments can give a crude estimate, but they are also accompanied with the loss of precious, limited instruction time that typically occurs during assessment. It would be great if intelligent tutoring systems could be used to do the benchmark assessment, so that no time from instruction is “stolen” to do extra assessments. However, since students learn from tutoring systems (e.g. Koedinger et al. 1997), 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. Thus, assessing students automatically, continuously and accurately without interfering with student learning is an appealing but also a challenging task.

In Feng, Heffernan & Koedinger (2009), we reported the counter-intuitive results that metrics from an intelligent tutoring system can better predict student’s state test scores

Authors’ addresses: M. Feng, SRI International, Menlo Park, California, USA. E-mail: mingyu.feng@sri.com; N.T. Heffernan, Department of Computer Science, Worcester Polytechnic Institute, Worcester, Massachusetts, USA. E-mail : nth@wpi.edu; Z.A. Pardos, Department of Computer Science, Worcester Polytechnic Institute, Worcester, Massachusetts, USA. E-mail: zparodos@wpi.edu; C. Heffernan, Department of Computer Science, Worcester Polytechnic Institute, Worcester, Massachusetts, USA. E-mail: ch@wpi.edu;

than traditional test context does. The metrics used include the number of hints that students needed to solve a problem correctly, the time it took them to solve it, the number of attempts that students made before correctly answering a question (called assistance metrics). This finding 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. However, there is a caveat that it takes more time for students to complete a test when they are allowed to request for assistance, which seems unfair for the contrast case. Feng & Heffernan (2010) addressed the caveat by controlling for time. We found that students did half the number of problems in a dynamic test setting (where help was administered by the tutor) as opposed to the static condition (where students received no help) and reported better predictions on the state test by the dynamic condition, but not a statistically reliable difference. Trivedi, Pardos, and Heffernan (2011) reanalyzed the same data set by introducing a more sophisticated method to ensemble together multiple models based upon clustering students. Although the findings from Feng & Heffernan (2010) and Trivedi et al. (2011) are encouraging, the predictions were made based upon 40 minutes of historical log data and the traditional test condition was simulated by only including students’ first attempt on the main problems and discarding all information while they were being tutored, which may be different from a real computer-based testing condition in several ways because of factors such as time constraints, test anxiety, etc. (Onwuegbuzie & Seaman, 1995). In order to address these concerns, in this paper, we run a randomized controlled study in a middle school in central Massachusetts.

2. LITERATURE REVIEW

Dynamic assessment (DA, Grigorenko & Sternberg, 1998; Sternberg & Grigorenko, 2001, 2002) 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 as a way to judge the extent of student knowledge limitations. 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 which learners are offered increasingly more explicit prewritten hints in response to incorrect responses. 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. Recently, Fuchs and colleagues (Fuchs et al., 2008) employed DA in predicting third graders’ development of mathematical problem solving.

3. METHODS

3.1 ASSISTments, the test bed

The ASSISTments platform (Razzaq et al., 2005) is an attempt to blend the positive features of both computer-based tutoring and benchmark testing. In ASSISTments, 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 students interact with the system, time-stamped student answers and student actions are logged into the background database.

3.2. Experimental Design

The experiment included two conditions, Test condition and Tutor condition. The Test condition mimicked the traditional computer-based test situation while the Tutor Condition follows the ASSISTments approach as described above. 392 students from 7th or 8th grade classes were randomly assigned to conditions. The experiment was run in one class period, about 45 minutes. Due to class schedules and climate conditions, students from different classes completed the study on different days across two weeks. The materials used for the experiment were selected from released Massachusetts Comprehensive Assessment System (MCAS) test items and the scaffolding questions and hint messages built in by subject matter experts from ASSISTments. The problem set contained 212 randomly organized problems.

4. ANALYSIS AND RESULTS

4.1 Measures and preliminary analysis

MCAS test scores from May, 2010 were used as the outcome measure for prediction¹. After linking student data collected from ASSISTments with their state test scores, we ended up with a total of 320 students, 160 in each condition. We examined the two experiment conditions on several measures (e.g. student average/standard deviation on MCAS score, the total number of problems finished in the experiments, and the total time they actually spent on solving problems) to ensure the two conditions were balanced. No significant difference was noticed except that students in the Test Condition finished more problems as expected. We reused the online metrics for dynamic testing that measures student accuracy, speed, attempts, and help-seeking behaviors (Feng, Heffernan & Koedinger, 2009; Feng & Heffernan, 2010), including measures on students' percent correct on main problems, which we often referred to as the "static metric", the number of main problems students completed, students' percent correct on scaffolding questions, the average number of hint requests per question, the average number of attempts students made for each question, how long it takes for a student to answer a question, whether main or scaffolding, measured in seconds, how often a student reached the "bottom-out" hints that revealed the correct answer, etc. Among these metrics, the Test condition used only measures on students' percent correct on main problems and the number of main problems students completed during the experiment, while the Tutor condition used all of the metrics. Additionally, during our modeling process, the metrics were normalized so that they were all on the same scale.²

4.2. Modeling and results

We followed Feng & Heffernan (2010) to fit a stepwise linear regression model using the dynamic assessment features as independent variables to make a prediction on the MCAS scores. In order to capture non-linear relationships and to fit different models for different groups of students, we also introduced random forests algorithm and clustering technique, following Trivedi et al. (2011). Using random forests (Breiman, 2001) algorithm N decision trees will be trained with each tree selecting a portion of the features at random and resamples the original dataset with replacement. Each decision tree is then used to make a prediction of unseen data. The predictions of all the trees are combined by uniform averaging to give the final prediction of the Random Forests algorithm. Using clustering technique, the training data is first used to define N clusters and then a specific

¹ For those who are confused, yes, we are "predicting" history data. We don't want to wait till September 2011 to do the analysis. Another reason is that our analysis of MCAS data of students from Worcester, Massachusetts in the past 6 years shows there is a high correlation (0.8~0.9) between students' state test scores across years.

² The complete data will be available at http://teacherwiki.assistment.org/wiki/Feng2009#Follow_up_paper

classifier will be trained for the data of each cluster. During the predicting stage, an unseen data was first assigned to one of the three clusters and then the classifier for that cluster is used to make a prediction. We combined approaches mentioned above and fit five different models for each of the two data sets, one for the Test condition and one for the Tutor condition. Namely, the models we applied were a) linear regression, b) stepwise regression, c) random forests with 50 trees, d) clustering method with random forests as classifier, e) clustering method with linear regression as classifier. As mentioned before, for each model, the MCAS state test score was used as dependent variable and the computed online metrics as predictors. 5-fold cross-validation was run to evaluate the results of the analysis for every model.

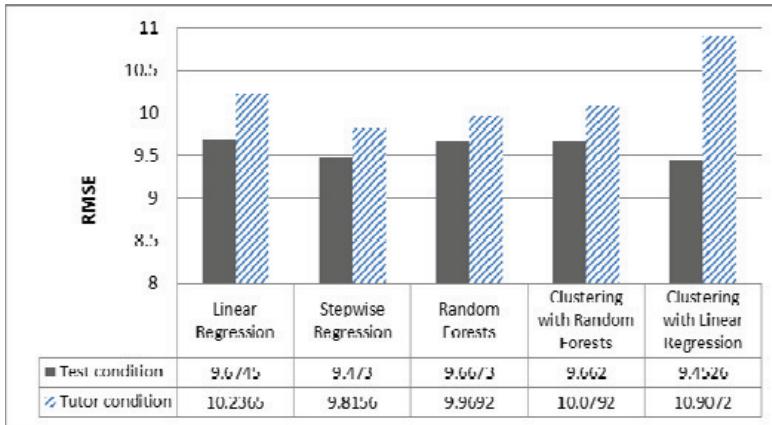


Fig. 1. RMSE from five fitted models

the Tutor condition with lowest correlation coefficient and the highest RMSE, which was contradictive to Trivedi et al. (2011). Additionally, we noticed that there was a trend in favor of the Test condition over the Tutor condition, which was also contradictive to what we have found before. We then conducted a series of t-tests between residuals generated from the five models for both conditions, but failed to find any significant difference.

The results from this experiment varied from our previous findings, which made us ponder what made the change. One thing we observed was that during the experiment, the 7th graders in the Tutor condition finished only a half of the problems as completed by those in the Test condition (9 vs. 18). Yet, for the 8th graders, the gap was not as big (12 vs. 16). Our subject matter expert confirmed that the content was appropriate for both grades. Yet, we found that 8th graders overall requested for significantly more hints than 7th graders did (effect size = 0.4, $p < 0.001$) after examining students' response data to all assignments before the experiment, which appeared to be related to students "gaming" the system (Baker et al., 2004). We speculated that gaming behaviors, such as requesting for hints for every question, always reaching out to the bottom-out hint, would be a big detriment to the dynamic assessment approach as the approach depends heavily on the interaction between students and the system, esp. their help-seeking behavior and response speed. We chose to model 7th and 8th graders separately and repeated the modeling process as described in section 4.3 for 7th graders and 8th graders respectively. We found out that in 7th grade the trend was in favor of the Tutor condition while in 8th grade was in favor of the Test condition, which was aligned with our speculation, but again neither difference was significant.

We report for both conditions the RMSEs from the five fitted models in Fig. 3. We observed that clustering method with linear regression worked best for the Test condition and has produced the highest correlation and lowest RMSE. However, it appeared to be the least effective model for

5. CONCLUSION

The results from Feng, Heffernan & Koedinger (2009) that started this line of work off were so exciting that they were cited in the National Educational Technology Plan (U.S. Department of Education, 2010) and were followed in Feng & Heffernan (2010) in a simulated study. However, we got a null result in a real randomized trial. Although reasoning from a null result is difficult, we think it is important to share with the community this null result as it does bring a caution to the excitement of the prior work. The three most salient differences between this study and our prior work are, firstly, in our prior work, we could run paired t-test using simulated data (see Feng & Heffernan, 2010 for details). Yet this experiment was not a between-subject design that can be more powerful if the variation between students is very large compared to the variation caused by the conditions. Secondly, in this experiment we had many fewer students (320 vs. 1392 in the prior study), which again reduced the statistical power of analysis. Thirdly, in the prior study, since existing log data were reused to simulate conditions, we made sure that the data sets contained exactly 40 minutes of work of every student. However, in this study, there was no guarantee that students all committed to working on ASSISTments problems for 40 minutes. As a matter of fact, on average, students have done only 30 minutes of problem solving work, which provided fewer amounts of data for our analysis. However, more complex models, such as the clustering method often require larger amounts of data to build a competent model. With all factors considered, we conclude that the experiment should be repeated with more students but also have the students swap conditions so that we can make a within-subject comparison.

REFERENCES

- Baker, R. S., Corbett, A.T., Koedinger, K. R., & Wagner, A. Z. (2004). Off-task behavior in the Cognitive Tutor Classroom: When Students “game the system”. *Proceedings of the ACM CHI 2004: Computer - Human Interaction*. (pp. 383-390). New York: ACM.
- Breiman, L. (2001) Random forests. *Machine Learning*, 45(1):5-32, 2001.
- 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. ED 269735.
- Campione. J. C., Brown. A. L., & Bryant. N. R. (1985). Individual Differences in Learning and Memory. In R.J. Steinberg (Ed.). *Human Abilities: An Information Processing Approach*, 103-126,. New York, W. H. Freeman.
- 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.
- Feng. M., Heffernan. N. T., (2010). Can We Get Better Assessment From A Tutoring System Compared to Traditional Paper Testing? Can We Have Our Cake (better assessment) and Eat it too (student learning during the test)? In *Proceedings of the 3rd International Conference on Educational Data Mining*, 41-50.
- Grigerenko, E. L. & Steinberg, R. J. (1998). Dynamic Testing. *Psychological Bulletin*, 124, 75-111
- 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' of Mathematical Problem Solving. *Journal of Educational Psychology*, 100(4), 829-850.
- U.S. Department of Education (2011). National Educational Technology Plan 2010. <http://www.ed.gov/technology/netp-2010>
- Onwuegbuzie, A. J., & Seaman, M. A. (1995). The effect of time constraints and statistics test anxiety on test performance in a statistics course. *Journal of Experimental Education*, 63, 115-124.
- Razzaq. L., Feng M., Nuzzo-Jones. G., Heffernan. N. T., Koedinger. K. R., Junker. B., Ritter. S., Knight A., Aniszczuky. C., Choksey. S., Livak. T., Mercado. E., Turner. T. E., Upalekar R., Walonoski. J.A., Macasek. M. A., & Rasmussen. K. P. (2005). The Assistment Project: Blending Assessment and Assisting. In C. K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds). *Proceedings of the 12th International Conference on Artificial Intelligence in Education, Amsterdam*. ISO Press, pp 555-562.
- Trivedi, S., Pardos, Z.A., Heffernan, N.T. (2011). Clustering Students to Generate an Ensemble to Improve Standard Test Score Prediction. Accepted by the 15th International Conference on Artificial Intelligence in Education. Christchurch, New Zealand.

Evaluating a Bayesian Student Model of Decimal Misconceptions

G. GOGUADZE

Saarland University, Germany,

S. SOSNOVSKY

DFKI GmbH, Germany,

S. ISOTANI

Carnegie Mellon University, U.S.,

AND

B. M. MCLAREN,

Carnegie Mellon University, U.S.

Among other applications of educational data mining, evaluation of student models is essential for an adaptive educational system. This paper describes the evaluation of a Bayesian model of student misconceptions in the domain of decimals. The Bayesian model supports a remote adaptation service for an Intelligent Tutoring System within a project focused on adaptively presenting erroneous examples to students. We have evaluated the accuracy of the student model by comparing its predictions to the outcomes of students' logged interactions from a study with 255 school children. Students' logs were used for retrospective training of the Bayesian network parameters. The accuracy of the student model was evaluated from three different perspectives: its ability to predict the outcome of an individual student's answer, the correctness of the answer, and the presence of a particular misconception. The results show that the model's predictions reach a high level of precision, especially in predicting the presence of student misconceptions.

Key Words and Phrases: Student model evaluation, Bayesian networks, Bayesian student modeling

1. INTRODUCTION

The quality of an adaptive educational system (AES) critically depends on the quality of its student modeling. The system might implement a sound adaptation strategy and provide students with well-designed learning content, but if its estimations of students' knowledge are incorrect, the adaptive interventions it produces are unlikely to be effective. In recent years, significant efforts have been expended to develop a methodology for layered evaluation of AES that allows examining student modeling components in isolation (BRUSILOVSKY, ET AL., 2004). Various approaches have been used for measuring the goodness of a particular student modeling mechanism (SOSNOVSKY, & BRUSILOVSKY, 2005), guiding the improvement of a student model (SM) (MARTIN ET AL., 2011) or selecting the best SM configuration among several alternatives (YUDELSON, ET AL., 2008). All of these evaluations have been based on rigorous analyses of students' logs generated by the systems.

In this paper, we describe the development of a Bayesian network (BN) SM and a data mining study aimed at validating its quality. The model represents students' misconceptions in the domain of decimals. It was designed within the framework of the

Authors' addresses: G. Goguadze, Saarland University, Saarbrücken, Germany, E-mail:george@activemath.org; S. Sosnovsky, DFKI, Saarbrücken, Germany, E-mail: sosnovsky@gmail.com; S. Isotani, B. M. McLaren, Human-Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, PA, U.S., E-mail: sisotani@gmail.com, bmclaren@cs.cmu.edu

AdaptErrEx project¹, which focuses on presenting and adapting erroneous examples (step-by-step solutions to decimal math problems in which at least one step is incorrect) to remediate students' misconceptions.

The evaluation of the model was done based on the data logs of 255 middle school students working with test problems in the domain of decimals. Data from 70% of the students was used for training model parameters. The remaining 30% of the data was used as the test set to compute three metrics, estimating how well the model predicts:

1. the exact answer to the next problem tackled by a student;
2. the correctness of the next answer provided by a student; and
3. the presence of a misconception the student has.

In order to compute these metrics, we compared the predictions of the individual SMs with the students' results on a delayed posttest. Although the values achieved for all three metrics could potentially be improved, they by far exceed the baseline of a random prediction. These results support our belief that the model is capable of accurate adaptation and encourage us to continue investigating ways to improve it.

2. MODELING STUDENTS' MISCONCEPTIONS IN ADAPTERREX

BNs are well-established tools for representing and reasoning about uncertainty in student models (CONATI, ET AL., 2002; MILLÁN, ET AL., 2010). Perhaps the closest example to the BN-based SM developed for AdaptErrEx is the SM of the DCT tutor that helped students learn decimal comparisons (STACEY ET AL., 2003). In the DCT's model, the misconceptions were represented as two probabilistic nodes identifying basic judgments used by a student for comparing decimals (e.g. "longer decimals are larger") and possible misconceived reasons for such judgments (e.g. "because longer integers are larger"). The causal relation between the two nodes was modeled with conditional probabilities defining the chance a student would come up with a basic judgment if she had a particular finer-grained misconception. The evidence nodes representing learning tasks were conditionally dependent on both misconception nodes.

A different approach to BN-based domain and student modeling, focused on skills rather than misconceptions, is described in (COLLINS, ET AL., 1996). The domain model here is a hierarchy of skills, where the probability of mastering a super-skill is conditionally dependent on mastery of the sub-skills. The bottom-level skills are probabilistically connected with the evidence nodes representing the test questions.

In AdaptErrEx we have followed an approach that is a combination of the two prior approaches. Based on the results of an extensive literature review of students' learning of decimals, we identified the most frequently occurring decimal misconceptions and organized them into a taxonomy based on their differences and similarities (ISOTANI, MCLAREN, & ALTMAN, 2010). The resultant taxonomy connects commonly observed misconceptions to the possible higher-level reasons for the misconceptions (e.g., the misconception "longer decimals are larger" is connected to the reason "student treats decimals like integers") providing means for diagnosing students' learning difficulties.

To account for dependencies between misconceptions, a BN was built, where each misconception is represented by a probabilistic node with two possible alternatives (present/absent). The taxonomic relations between the nodes are accompanied by tables of conditional probabilities representing the influence of the probability of presence of a parent misconception on the probabilities of presence of its child misconceptions.

The evidence nodes in the network represent problems. They can be connected to one or more misconceptions. The evidence nodes contain several alternatives, where each alternative corresponds to a possible answer the student might give to the problem. Every

¹ See <http://www.cs.cmu.edu/~bmclaren/projects/AdaptErrEx>

evidence node alternative is probabilistically connected to the corresponding misconception node alternatives. This means that presence/absence of a misconception influences the likelihood of a student giving a certain answer to the problem.

Overall, the developed network contains twelve misconception nodes, where seven nodes represent the most typical decimal misconceptions and five nodes serve as higher-level reasons for their occurrence. The misconception nodes are connected to 126 evidence nodes representing possible answers to decimal problems. The problems are divided into three isomorphic problem sets (set A, set B and set C), each set containing 42 problems. In order to ensure that the results are not driven by differences in the problems, the roles of problem sets A, B and C were counterbalanced across student groups. Each set was used either for a pretest, an immediate posttest, or a delayed posttest. In total, there are six possible combinations of the sets (ABC, ACB, BAC, BCA, CAB and CBA) depending on the role each set plays. Consequently, students were randomly assigned to one of the six groups, facing one of the six sequences of tests.

3. EVALUATING THE ACCURACY OF MODEL'S PREDICTIONS

This section summarizes our approach to evaluating the AdaptErrEx BN's capability to predict the effective state of student's learning. The approach consists of three steps:

- training the domain model based on the pretest data from 70% of the students;
- learning student models using the logs of the remaining 30% of the students;
- evaluating the accuracy of the model's predictions using 3 different metrics.

3.1. Training the initial domain model

Parameter estimation is a well-known challenge in the field of BNs. In our case, these parameters include prior probabilities for misconception nodes and conditional probability tables for links between the nodes. To complete this task, we supplied initial estimations of network parameters and then refined them with the training algorithm.

For the training set we randomly selected 70% of the students participating in the study. Based on the pretest logs of these students, from taking one of the three tests (A, B, or C), the prior probabilities for misconception nodes and the conditional probabilities for evidence nodes of all three problem sets A, B and C are computed. In this way, the resulting BN represents the initial state of knowledge of decimals (more specifically, the predicted state of misconceptions) for a typical student from the target population. The prior probabilities of misconception nodes quantify how likely such an average student is to have a particular misconception. The conditional probabilities encode the strength of a causal relation among misconceptions and between the misconceptions and the problems.

3.2. Learning specific student models

After the initial training/calibration, the BN was ready to learn the specific models of individual students. In order to do this, we fed the activity logs of the remaining 30% of the students to the network. Only answers to the pretest and immediate posttest were used on this step. This evidence back-propagated to the relevant misconception nodes and updated all posterior probabilities, thus individualizing the networks. The resulting collection of BNs contained individual misconception models for every student in the test set. Each resulting individual SM took into account both the collective traits of the target population and the history of idiosyncratic behavior of the corresponding student.

3.3. Estimating accuracy of the student model's predictions

The BNs obtained in Step 2 can be used to make individual predictions about students. Based on such predictions, an AES could control the individual learning experiences of its students. We identified three types of predictions and verified their average accuracy by comparing the models of the students from the test set with their results on the delayed

posttest. The three prediction types were: predicting the actual student answer, predicting the correctness of the student answer, and predicting the presence of a student misconception. The notion of accuracy in these three cases was defined as follows:

I. A prediction of the actual student answer is accurate if the alternative chosen by a student for a posttest problem had the highest probability in this student's BN trained in Step 2. The corresponding metric is computed as a percentage of accurate predictions.

II. A prediction of the correctness of the student's answer is accurate if:

- the student answers correctly to a delayed posttest problem and the probability of the correct alternative for this problem's node is maximum in the BN trained for this student in Step 2;
- or the student answers incorrectly to a delayed posttest problem and the probability of the correct alternative for this problem's node is less than the sum of probabilities of incorrect alternatives in the BN trained in Step 2.

The corresponding metric is computed as a percentage of accurate predictions.

III. A prediction of the presence of a misconception is defined as follows. Based on the state of a misconception node, the student is believed to have a corresponding misconception if its probability is greater than 0.5. This prediction is considered accurate if during the delayed posttest the student has shown more evidence of having the misconception than not having it (and vice-versa). The evidence is quantified as an average rate of misconception occurrence in the students' answers in the delayed posttest. The average rate of misconception occurrence is computed in the following way:

Let $P(M)$ be the probability of the presence of a misconception M in a Bayesian model, $N_{pos}(M)$ – the number of student's answers that provide evidence for the misconception M , $N_{neg}(M)$ – the number of correct answers to the problems that can diagnose M , and $N(M)$ – the total number of problems that address this misconception. Then, the model prediction is said to be accurate if and only if:

$$\left[\left(P(M) \geq 0.5 \right) \& \left(\frac{N_{pos}(M)}{N(M)} \geq 0.5 \right) \right] \vee \left[\left(P(M) < 0.5 \right) \& \left(\frac{N_{neg}(M)}{N(M)} \geq 0.5 \right) \right]$$

4. EXPERIMENT SETTINGS AND EVALUATION RESULTS

The data for the evaluation came from an empirical study conducted in a middle school classroom in Pittsburgh, PA (U.S.A.) during the fall of 2010. Overall, 255 students from 6th-8th grades participated in the study. The study had several subsections conducted over multiple sessions, including a pretest, treatment problems, an immediate posttest, and (one week later) a delayed posttest. The 126 test problems were split into 3 isomorphic problem sets (A, B, and C) and the roles of these problem sets being pretest, posttest or delayed posttest were counterbalanced across student groups. The learning materials came from the domain of decimals. The MathTutor web-based system (ALEVEN, ET AL., 2009) was used to deliver the materials to the participants. Students took 4 to 5 sessions to complete all of the materials.

MathTutor logs all students' interactions, as well as diagnostic information in the PSLC DataShop (KOEDINGER ET AL., 2010). In total, we analyzed 31,049 student interaction events, which resulted from each of the 255 students solving up to 126 problems. The accuracy values were calculated for the test set (i.e., the 77 students; data from these students was not used in the training phase). Using the metrics defined in Section 2.3, we evaluated the accuracy of the predictions of our SM. As a result of the calculation, the average accuracy of predicting the actual answer of the students in the delayed posttest was 60%, whereas the average accuracy of predicting the answer correctness was 69%. The average accuracy of predicting misconceptions was 87%

($\sigma=0.148$). Similar studies on evaluating the accuracy of predictions of a BN student model of the DCT tutor (NICOLSON ET AL., 2001), achieve comparable accuracy for predicting misconceptions (80-90%). However, our model estimates students' misconceptions with higher granularity (a node per misconception compared to one node for all misconceptions).

5. CONCLUSIONS AND FUTURE WORK

We have presented the design and evaluation of a Bayesian approach to modeling student misconceptions. Three different metrics have been compared for estimating how well the model predicts student's misconceptions. The evaluation results demonstrate high accuracy of models' predictions, yet leaving room for improvement.

Future work is planned in two main directions: improving the structure of the Bayesian model and enhancing the methods of evaluation of the model validity. We plan to experiment with different configurations of BNs, such as dynamic BNs, and the networks with soft evidence nodes. When adjusting the evaluation method we could experiment with additional parameters of the students such as gender, grade, or general math skills. Difficulty of the problems could be used here as well as an additional parameter in the computation of the accuracy metrics. For example, if the problem is very easy, the student is likely to solve it correctly even if the probability of having a misconception is high, and the other way round, difficult problems can be solved incorrectly even if the probabilities of misconceptions are low.

ACKNOWLEDGEMENTS

The U.S. Department of Education Institute of Education Sciences, grant# R305A090460, funded this work.

REFERENCES

- ALEVEN, V., MCLAREN, B., & SEWALL, J., 2009. Scaling up programming by demonstration for intelligent tutoring systems development: An open-access website for middle school mathematics learning. *IEEE Transactions on Learning Technologies*, 2(2), 64-78.
- BRUSILOVSKY, P., KARAGIANNIDIS, C., AND SAMPSON, D., 2004. Layered evaluation of adaptive learning systems. *Int'l Journal of Continuing Engineering Education and Lifelong Learning* 14 (4/5), 402 – 421.
- CONATI, C., GERTNER, A. AND VANLEHN K., 2002. Using Bayesian networks to manage uncertainty in student modeling. *Journal of User Modeling and User-Adapted Interaction*, vol. 12(4), p. 371-417
- COLLINS, J., GREER, & J., HUANG, S., 1996. Adaptive assessment using granularity hierarchies and Bayesian nets. In: *Lecture Notes in Computer Science*. Vol. 1086. pp. 569–577.
- ISOTANI, S., MCLAREN, B., & ALTMAN, M., 2010. Towards intelligent tutoring with erroneous examples: A taxonomy of decimal misconceptions. Submitted as a poster paper to the *Tenth International Conference on Intelligent Tutoring Systems (ITS-2010)*. Pittsburgh, PA.
- KOEDINGER, K., BAKER, R., CUNNINGHAM, K., SKOGSHOLM, A., LEBER, B., STAMPER, J., 2010. 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, pp 43-55, FL: CRC Press.
- MARTIN, B., MITROVIC, A., KOEDINGER, K., & MATHAN, S., 2011. Evaluating and Improving Adaptive Educational Systems with Learning Curves, *J. of User Modeling and User Adapted Int.*, 21(3), Springer.
- MILLÁN, E., LOBODA, T., & PÉREZ-DE-LA-CRUZ, J., 2010. Bayesian networks for student model engineering, *Computers & Education*, Vol. 55, Issue 4, 1663-1683.
- NICOLSON, A., BONEH, T., WILKIN, T., STACEY, K., SONENBERG, L., & STEINLE, V., 2001. A case study in knowledge discovery and elicitation in an intelligent tutoring application. In *Proceedings of the 17th Conference on Uncertainty in AI*, pp. 386-394, Seattle.
- SOSNOVSKY, S., & BRUSILOVSKY, P., 2005. Layered Evaluation of Topic-Based Adaptation to Student Knowledge. In *Proceedings of 4th Workshop on the Evaluation of Adaptive Systems*, 47-56.
- STACEY, K., SONENBERG, E., NICHOLSON, A., BONEH, T., & STEINLE, V., 2003. A teacher model exploiting cognitive conflict driven by a Bayesian network. In Peter Brusilovsky, Albert T. Corbett, Fiorella De Rosis (Eds), *User Modeling 2003: Proceedings of the Ninth International Conference*. (pp. 352-362) New York: Springer-Verlag (ISBN 3540403817).
- YUDELSON, M., MEDVEDEVA, O., AND CROWLEY, R., 2008. A multifactor approach to student model evaluation. *User Modeling and User-Adapted Interaction*, 18(4), 349-382.

Exploring user data from a game-like math tutor: a case study in causal modeling

D. RAI AND J. E. BECK

Worcester Polytechnic Institute, USA

We have used causal modeling to understand data from a game-like math tutor, *Monkey's Revenge*. We collected student data of various types such as their attitude and enjoyment via surveys, performance within tutor via logging, and learning as measured by a pre/post test. Although the data are observational, we want to understand the causal relationships between the variables we have collected. We contrast the causal modeling approach to the results we achieve with traditional approaches such as correlation matrix and multiple regression. Relative to traditional approaches, we found that causal modeling did a better job at detecting and representing spurious association, and direct and indirect effects. We found that the causal model, particularly one augmented with domain knowledge about likely causal relationships, resulted in much more plausible and interpretable model. We present a case study for blending exploratory results from causal modeling with randomized controlled studies to validate hypotheses.

Key Words and Phrases: Causal modeling, confounders, structural equation modeling, case study

1. INTRODUCTION

Making causal inferences based on non experimental statistical data has been a controversial topic [Freedman, 1987, Rogosa, 1987, Denis, 2006]. While randomized controlled trials are the standard approach to take care of intervening third variables, causal modeling is an alternative method of making causal inferences [Pearl, 2009, Sprites et al., 2001] based on observational data making certain causal assumptions. We are using the causal modeling approach to analyze and explore the data from a game-like math tutor, *Monkey's Revenge*.

We created four different versions of Monkey's Revenge with varying degree of game-like properties. A total of 297 middle school (12-14 year olds) students from four Northeastern schools in the United States participated in the study. We logged their tutor activities and asked 16 survey questions using 5 point Likert scale from "strongly disagree" to "strongly agree." We then used factor analysis to reduce the variables into six categories: ***likeMath***, ***mathSelfConcept***, ***pedagogical preference*** (prefer Computers to books; find real world examples helpful for learning Math.), ***tutorHelpful***, ***tutorConfusing***, ***likeTutor***. From students' log data, we calculated variables ***%correct*** (ratio of correct problems to total problems); ***avgAttemptTime*** (average time spent on each attempt) and ***avgHints*** (average number of hints asked on each question). We also collected ***preTestScore*** and ***prePostGain*** and used a variable ***game-like*** as an ordinal parameter (taking on values 1 through 4). Along with using causal modeling to explore and analyze our data, we analyze the causal modeling approach itself. We compare it against the standard statistical approaches of correlation and multiple regression.

2. CAUSAL MODELING, CORRELATION MATRIX

Based on the data we collected, we used TETRAD, a free causal modeling software package [Glymour et al., 2004] with the PC search algorithm to generate a causal graph (fig 1). We also generated a graph based on the correlation matrix, computed by finding the correlations between the variables (fig 2).

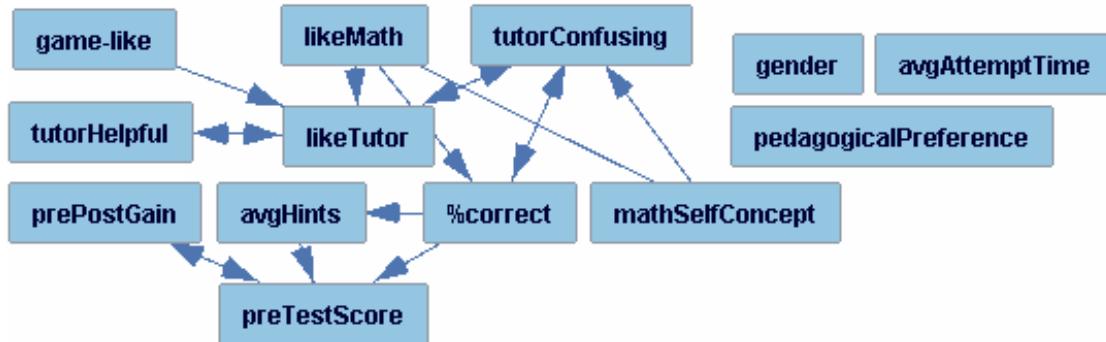
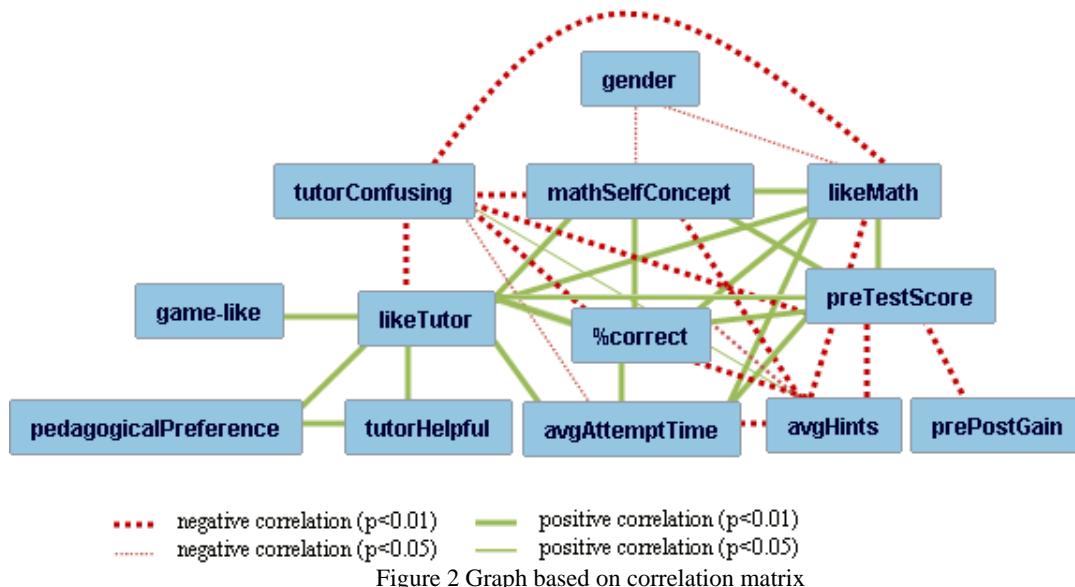


Figure 1 Causal model from PC algorithm without domain knowledge

True negatives (indirect and spurious associations): Correlation is not causation as there might be possible confounders causing the spurious association, and causal modeling controls for all third variables regarding them as possible confounders. From the correlation matrix, we see that *likeTutor* and *%correct* are correlated which would suggest that students who like the tutor performed better. This result could be interpreted as evidence for student engagement, since students who liked the tutor are presumably more engaged while using it. But the causal model (Fig 1) infers that this is a spurious association confounded by *likeMath*. Students who like math tend to like tutor more and to have better performance. Once we control for *likeMath*, there is no relation between *likeTutor* and *%correct*. Thus, the scientific interpretation of the data changes.



Still, the causal model is limited to assertions about the observed variables as there might be other confounders which we have not observed. Causal modeling makes also distinction between direct and indirect association. For example, *likeMath* and *avgHints* are negatively correlated, which suggests that the students who like math ask for fewer hints. But once we control for *%correct*, that correlation is gone (see Fig 1), suggesting that the students who like math ask for fewer hints only because they already know the correct responses and so do not need as much help—there is no direct effect.

False negatives (reduced statistical power and multicollinearity): Controlling on third variables reduces statistical power and we might get false negatives if we have few data. Multicollinearity is an extreme case when the independent variables are correlated among themselves. For example: *avgAttemptTime* is correlated with both *%correct* (0.3**) and *preTestScore*(0.3**). But since, *%correct* and *preTestScore* are highly correlated among themselves (0.6**), *avgAttemptTime* is conditionally independent to both of them. We can see that *avgAttemptTime* is an isolated node in figure 1; in contrast, the correlation graph (Figure 2) indicates *avgAttemptTime* is related to both *preTestScore* and *%correct*.

3. CAUSAL STRUCTURE, PATH ORIENTATION AND DOMAIN KNOWLEDGE

In the causal model, some edges have plausible orientations (e.g: *likeMath* → *likeTutor* ← *game-like*). Using the information that *likeTutor* is correlated with both *likeMath* and *game-like*, but *likeMath* and *game-like* are independent between themselves, search algorithm correctly identifies that it is not *likeTutor* influencing *likeMath* and *game-like* but the other way round. However, we see that there are other edges which are incorrectly oriented such as *%correct* → *preTestScore*; student performance on the tutor cannot have influenced a pretest that occurred before students began using the tutor.

Correlation underdetermines causality as covariance in statistical data is rarely sufficient to disambiguate causality. Therefore, even after we use search algorithms to find some structure, there are a number of “Markov equivalent” structures. In TETRAD, we can add domain knowledge in the form of knowledge tiers which represent the causal hierarchy. Causal links are only permitted to later tiers, and cannot go back to previous tiers. We used the following knowledge tier based on our knowledge of assumed causal hierarchy and temporal precedence.

- i. Gender,
- ii. Game-like, mathSelfConcept
- iii. likeMath, Pedagogical preference
- iv. preTestScore
- v. %correct, avgAttemptTime, avgHints, tutorConfusing, tutorHelpful
- vi. likeTutor
- vii. prePostGain

We see from Figure 1 and Figure 3 that adding domain knowledge not only fixes the path orientations (*preTestScore* → *%correct*), but have changed the whole causal structure adding some new causal links (*gender* → *mathSelfConcept*, *pedagogicalPreference* → *tutorHelpful*, *correct* → *avgAttemptTime*).

At first, it may appear that knowledge of causal hierarchy only helps to orient the edges specifying which one is cause and which one is effect. However, besides distinguishing variables as potential causes and effects, the domain knowledge also restricts the set of variables to be considered as confounders and mediators. Note that many data analyses have such causal hierarchies implicit in the analysis and conclusions.

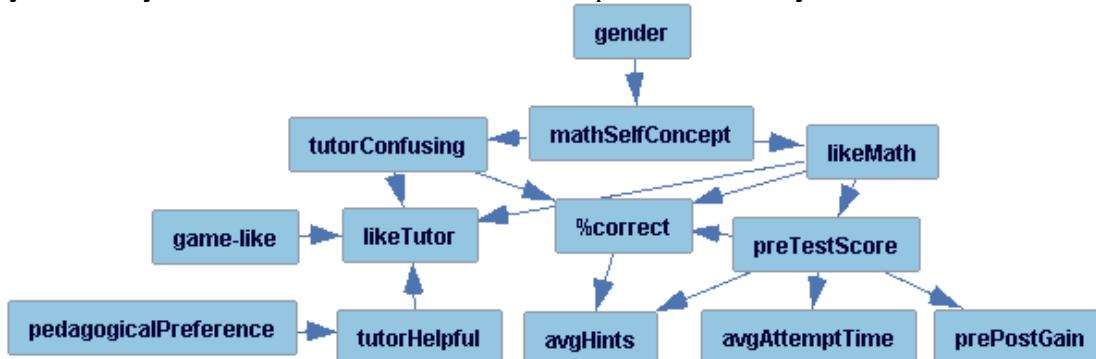


Figure 3 Causal model with domain knowledge

Sometimes, we do not know about the causal hierarchy of the variables we are trying to analyze and may not know which is the cause and which is the effect, but having information of the causal hierarchy of third variables, such as whether they are a potential confounder or a potential mediator, can help infer if there is any causal path between the variables of interest. We can illustrate this with a concrete example in education. Suppose we have observed that engagement and learning are correlated, but want to understand the causal relation between them. Imagine there are two other variables, prior knowledge, a potential confounder, and performance, a potential mediator. Consider two scenarios: if partialling out prior knowledge removes the correlation, then we know there is no causal relationship between engagement and learning, and the causal structure is $\text{engagement} \leftarrow \text{prior knowledge} \rightarrow \text{learning}$. On the other hand, if partialing out performance removes the correlation between engagement and learning, then there is still an *indirect* causal effect between the two, either $\text{engagement} \rightarrow \text{performance} \rightarrow \text{learning}$, or $\text{learning} \rightarrow \text{performance} \rightarrow \text{engagement}$.

Interestingly, adding domain knowledge can also address the problem of multicollinearity. Since we have set *preTestScore* on higher causal tier than *%correct*, *%correct* cannot be a possible confounder or mediator and therefore, the partial correlation (*preTestScore*, *avgAttemptTime* / *%correct*) is not calculated and the correlation between *preTestScore* to *avgAttemptTime* is maintained.

4. CAUSAL MODELING AND MULTIPLE REGRESSION

Causal modeling is a sophisticated extension to multiple regression which employs a series of multiple regression. Multiple regression only looks at direct effect but fails at identifying **indirect effects**. While multiple regression can be equally robust when it comes to predictive accuracy, causal modeling provides a better representation and framework to *understand* interrelationships of variables. Since causal modeling allows multiple layers of associations of variables, it adds affordance to insert **domain knowledge** in the form of a causal hierarchy. On top of the statistical assumptions used by statistical methods such as regression, causal modeling adds **causal assumptions** such as faithfulness and causal sufficiency [Sprites et al., 2001]. Stronger assumptions add more analytical power but also higher chances of inaccuracy. It is up to researcher to select these assumptions based on their data and domain. We have accepted the causal assumptions made by TETRAD since they seem reasonable for our data and purpose.

5. CONFIRMATORY, EXPLORATORY AND GRAPHICAL TOOL

Causal modeling can be used to test goodness of fit of a model constructed *a priori* from theory. We did not start with such a model, but the causal model has supported some of our prior hypotheses ($\text{pedagogicalPreference} \rightarrow \text{tutorHelpful} \rightarrow \text{likeTutor}$) and ($\text{mathSelfConcept} \rightarrow \text{tutorConfusing} \rightarrow \text{likeTutor}$).

Using causal modeling as an exploratory tool to generate new theories is controversial as the possibility of unobserved confounders and under determination of causality from correlation pose serious limitation to generate new valid conclusions. But, conditional independencies in data and domain knowledge can offer some new inferences which can be helpful in guiding us towards further analyses and examination. For example, in our causal model, we found that *likeMath* has both direct ($\text{likeMath} \rightarrow \text{percentCorrect}$) and indirect ($\text{likeMath} \rightarrow \text{preTestScore} \rightarrow \text{percentCorrect}$) effect on *percentCorrect*. Based on this, we are considering two possible causal models as shown in figure 4. Model I suggests that *pretestScore* does not capture all of the variance in prior knowledge of the student, as represented by the latent node “Prior knowledge.” So,

students who like math and have high prior knowledge may have a low pre test score but they have high performance nonetheless. Model II on the other hand suggests that students who like math both have higher prior knowledge and are more engaged, and have therefore higher performance. In other words, likeMath affects both prior knowledge and engagement.

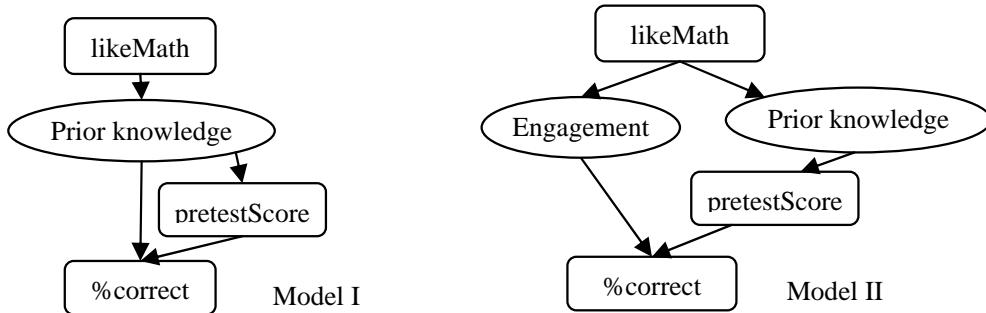


Figure 4 Two possible causal models linking *likeMath* and *%correct*

We were not able to make any conclusive findings with the causal model, but it has at least made interesting inferences and raised questions that are very important for us. It has directed towards the possibilities that we would like to make further examination and possibly run some controlled randomized trials.

Even if researchers are skeptical of the domain knowledge we have brought to bear and are dubious of the causal modeling assumptions, it is still possible to consider Figure 1 without the assumption that the edges represent causality. This graph would be a compact representation of the partial correlation relationships among the variables

6. CONCLUSIONS

We have used a causal modeling approach to explore the data from a game-like math tutor. Based on statistical independence within data and our domain knowledge, TETRAD's causal inference algorithm generated a causal model which not only confirmed some of our prior hypotheses about data but also made some interesting new findings. Causal modeling does a good job of identifying not only cause and effect but also confounders so that we can find spurious associations and mediators so that we know both direct and indirect effects. But those inferences cannot be claimed as accurate since causal modeling cannot identify spurious association caused by unobserved confounders and there are multiple Markov equivalent causal models that can be generated from the same data. Still, causal modeling is the best approach we have found, particularly when compared with common statistical techniques such as correlation and multiple regression to generate most plausible inferences from observational educational data sets.

REFERENCES

- DENIS, D. J. LEGERSKI, J. 2006. Causal Modeling and the Origins of Path Analysis. *Theory & Science*, Vol. 7, No. 2
- FREEDMAN, D. A. 1987. As Others See Us: A Case Study in Path Analysis. *Journal of Educational Statistics*, (12:2), pp. 101-128.
- GLYMOUR, C., MADIGAN, D., PREGIBON, D., SMYTH, P. Statistical Themes and Lessons for Data Mining. *Data Mining and Knowledge Discovery*, 2004. p. 11-24
- GLYMOUR,C., SCHEINES.R. 2004. Causal modeling with the TETRAD program. *Synthese*.37-64
- PEARL J. 2009. Causality. 2nd Edition. Cambridge University Press.
- ROGOSA, D. 1987. Causal models donot support scientific conclusions: A comment in support of Freedman. *Journal of educational statistics*. Vol. 12, No. 2, pp.185-195
- SPRITES, P. GLYMOUR, C. SCHEINES, R. 2001. Causation, Prediction, and Search. 2nd Edition. MIT press

POSTERS

Goal Orientation and Changes of Carelessness over Consecutive Trials in Science Inquiry

A. HERSHKOVITZ

R.S.J.D. BAKER

J. GOBERT

AND

M. WIXON

Worcester Polytechnic Institute, USA

In this paper, we studied the relationship between goal orientation within a science inquiry learning environment for middle school students, and the manifestation of carelessness over consecutive trials. Carelessness is defined as not demonstrating a skill despite knowing it; we measured carelessness using a machine learned model. Findings suggest that students with performance goals demonstrate an increase in carelessness sooner in the set of trials than do students with learning goals, and that students with lack of goals are consistent with their carelessness over trials.

Key Words and Phrases: carelessness, goal orientation, science inquiry, cluster analysis, automated detectors

1. INTRODUCTION

In recent years, there is increasing evidence that the goals students have during learning play a key role in their learning outcomes. These goals might impact learning by creating different forms of disengagement. One disengaged behavior is carelessness, i.e., when a student fails to answer an item correctly despite possessing the required skills (Clements, 1982). Previous research has indicated that students possessing mastery or performance goal orientation have (on average) double the probability of carelessness as students characterized by low scores for these goal orientations (Hershkovitz et al., 2011). In this research, carelessness was automatically detected using a log-based machine-learned model. This measurement can be quickly applied to data from new students and situations. Within this study, we examine how carelessness is manifested over consecutive trials by students with different goal orientations.

2. METHODOLOGY

The learning environment. We used the Science Assistsments learning environment (www.scienceassistsments.org) and a microworld for phase change to study carelessness in demonstrating three science inquiry skills – namely, control for variable strategy (CVS), testing articulated hypotheses, and planning using the table tool. The microworld engages students in authentic science while supporting inquiry via widgets. The students completed three learning activities involving these three skills. The learning environment detects whether students demonstrate inquiry skills using validated machine-learned models of these behaviors.

Participants, Data. 130 public middle school 8th graders (Central Massachusetts), 12-14 years old. Students' fine-grained actions were logged, then analyzed at the "clip" level (a consecutive set of a student's actions during experimentation). Goal orientation was assessed with the PALS (Patterns of Adaptive Learning Scales) survey (Midgley et al.,

1997).

Carelessness Detector. We developed the carelessness detector in RapidMiner 5.0 using REPTree, a regression tree classifier. Carelessness, first predicted at the clip-level, was averaged at the student-level. The resulting regression tree (a 6-fold cross-validation correlation of $r=0.63$) includes 13 variables, has a size of 35, and a total depth of 13.

Cluster Analysis. Exploratory cluster analysis was conducted to group students by their PALS scores in order to examine whether certain sub-groups of students with specific characteristic patterns on the PALS survey also differ on carelessness manifestation over trials. We used Two-step Cluster Analysis (in SPSS 17.0) with the PALS scores (Z-standardized) and a log-likelihood distance measure. We chose $k=3$ as it led to more interesting separations between aspects of the PALS. The clusters corresponded to: 1) *mastery goal orientation* ($N=35$), 2) *performance goal orientation* ($N=66$), and 3) *lack of goal orientation* ($N=20$).

3. RESULTS

The mean value of carelessness for the population was 0.08 ($SD=0.12$; $N=130$). Overall, carelessness increased over all trials (as students have more practice opportunities for the same skill), and its means for activities 1-3 were 0.04 ($SD=0.08$; $N=126$), 0.07 ($SD=0.13$; $N=127$), and 0.11 ($SD=0.19$; $N=112$), respectively. According to a paired-sample t-test, the increase in carelessness from activity 1 to 2 is significant, $t(122)=2.7$, $p<0.01$, as was the increase from activity 2 to 3, $t(109)=2.2$, $p<0.05$. Next, we analyzed how carelessness manifests over trials in the different clusters. These results are summarized in Figure 1. For the students in the **Learning Goals cluster**, mean carelessness does not significantly increase from activity 1 to 2, $t(32)=1.3$, $p=0.2$, nor from activity 2 to 3, $t(27)=1.5$, $p=0.15$; however, it is significantly higher in activity 3 compared to activity 1, $t(28)=3.1$, $p<0.01$. In the **Performance Goals cluster**, the mean carelessness significantly increases between activities 1 to 2, $t(61)=2.6$, $p<0.05$, but does not significantly increase between activities 2 to 3, $t(56)=1.5$, $p=0.14$.

These results suggest that the increase in carelessness for students with learning goals is slower but more dramatic than for students with performance goals. As for **Lack of Goals cluster**, no significant differences were found in mean carelessness for either pair of activities (all t-tests had $p>0.63$).

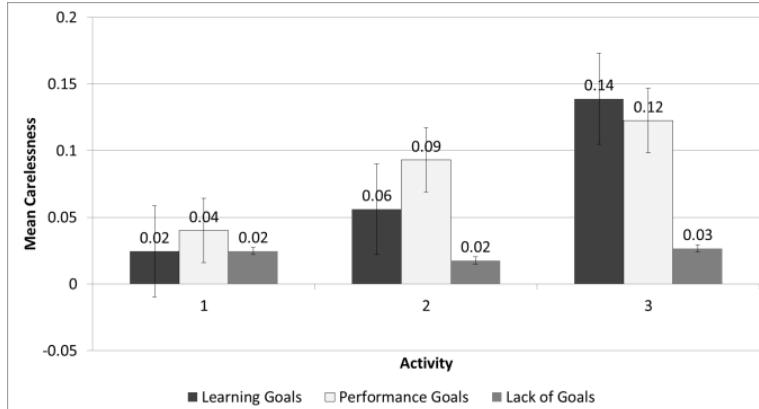


Fig. 1. Mean Carelessness over consecutive trials by PALS-based clusters

REFERENCES

- CLEMENTS, M.A. 1982. Careless errors made by sixth-grade children on written mathematical tasks. *Journal for Research in Mathematics Education*, 13, 136-144.
- HERSHKOVITZ, A., BAKER, R.S.J.d., GOBERT, J., WIXON, M. AND SAO PEDRO, M. 2011. Carelessness and goal orientation in science microworld. Poster presented at *AIED'2011*, Auckland, New Zealand.
- MIDGLEY, C., MAEHR, M., HICKS, L., ROESER, R., URDAN, T., ANDERMAN, E., & KAPLAN, A., ARUN-KUMAR, R., MIDDLETON, M. (1997). *Patterns of adaptive learning survey (PALS)*. Ann Arbor, MI: University of Michigan.

Towards Improvements On Domain-Independent Measurements For Collaborative Assessment

ANTONIO R. ANAYA

UNED, Spain

JESÚS G. BOTICARIO

UNED, Spain

Abstract. Assessment on collaborative student behavior is a longstanding issue in user modeling. Nowadays thanks to the proliferation of online learning and the vast amount of data on students' interactions this modeling issue features some alternatives. The purpose is not to depend on teachers or students assessments, which either requires management effort difficult to assume (due to some students-per-teacher ratios) or depends on individual motivations (i.e. student willingness on providing explicit feedback related to collaboration). In our research we have shown that based on frequent and regular analysis of those interactions it is feasible to obtain collaborative assessments that concurs with expert valorizations. This approach relies on data mining and machine learning techniques, which are applied to infer collaborative significant student's characteristics such as regularity, in terms of activity and initiative, and student acknowledgment of fellow-students. The advantages of the approach are to obtain domain-independent assessments, applicable in different learning management systems and exploitable over different courses and learning settings without the teacher involvement in the analysis process. The method has been developed from a collaborative experience involving hundreds of students over 3 consecutive year-courses. Here we focus on discussing the improvements on measurements provided during a new collaboration learning experience of this academic year.

Keywords. Collaborative Learning, Quantitative and Timely Data Mining Approach.

INTRODUCTION

Student collaboration assessments can improve learning and motivate students (Swan et al., 2006), albeit they must come from a frequent and regular student collaboration analysis (Johnson & Johnson, 2004). Thus, students' interactions, mainly communication interactions, should be frequently analyzed to provide a timely collaboration assessment, which can be used by students and teacher to improve the collaboration learning process.

To offer frequent and timely assessment the expert-based analysis approach is almost unaffordable (Bratitsis et al., 2008), only researchers who used quantitative student interaction and automatic methods were able to cope with the requirements (Gaudioso et al., 2009). In current LMS-based e-learning scenarios communication is mostly done through forums, and that is why they have been extensively used to reveal relevant students' collaborative characteristics (Dringus & Ellis, 2005). Once the collaboration assessments were inferred, most related research has advocated for displaying assessment to students and teacher (Bratitsis et al., 2008, Gaudioso et al., 2009).

Our approach is based on frequent and regular analysis of learners' interactions to obtain collaborative assessments that concurs with expert valorizations. To this end data mining and machine learning techniques have been applied. The advantages of the approach are to obtain domain-independent assessments, applicable in different learning management systems and exploitable over different courses and learning settings without the teacher involvement in the analysis process (Anaya & Boticario, 2011).

To support the assessment we have previously compared clustering and quantitative metric approaches and finally proposed a quantitative Metric Approach, which draws on a range of decision tree algorithms to inferring quantitative indicators such as regularity, in terms of activity and initiative, and student acknowledgment of fellow-students (Anaya & Boticario, 2010; Anaya & Boticario, 2011). From the lessons learnt after three years of experimentation with hundreds of students in a real collaborative learning experience we have organized this year-course experience in which there are several improvements on the Metric Approach and display strategies.

Following the collaborative learning experience is introduced. Next, the Metric Approach steps are summed up. Then, collaborative assessments results and displaying strategy are commented, and to conclude the future analysis.

COLLABORATIVE LEARNING EXPERIENCE

We have offered to students of Artificial Intelligence and Knowledge based Engineering (AI-KE) at UNED (2010-11) a collaborative e-learning experience using dotLRN platform, which provides documentation support and forums to collaborate. The experience is divided into two phases. In the 1st phase the students perform an individual task, which allows them to participate in the 2nd phase, where they are grouped into three-member teams and every team has to carry out five collaborative tasks. Team members' communications are managed through group forums.

METRIC APPROACH

From mining techniques applied on collected data from forum interactions the Metric Approach, which is based on machine learning techniques, proposes a mathematical formula that uses quantitative indicators to measure students'

Authors' addresses: Artificial Intelligence Department at the School of Computer Science (CSS) at UNED, Spain. E-mail: [arodriguez,jgb]@dia.uned.es

collaboration (Anaya & Boticario, 2011). That formula is refreshed on a regular basis to cope with the course pace. Here we sum up the main issues involved:

- Twelve quantitative statistical indicators are proposed (see Table I). These indicators are related to relevant student's characteristics: initiative, activity, regularity and acknowledgement.
- A set of decision tree algorithms (BFtree, DecisionStump, Functional trees, J48, Logistic trees, NBtree, Random tree, REPTree, Simple Cart) are applied to research the relationship between those indicators and students' collaboration labels supported by expert-based analysis (required for the configuration phase not any more on different courses). The research shows that the most collaborative-related indicators are (Anaya & Boticario, 2010): the regularity of the student initiative (L_{thrd}) and activity (L_{msg}), and the students' acknowledgment (N_r_{msg}).
- A metric (mathematical formula) is built from the above quantitative statistical indicators ($\text{Metric} = L_{msg} + N_{reply_msg} + L_{thrd}$). This metric is selected because it outperforms (i.e. less error and better discrimination of collaborative levels) other metrics, which consider alternative indicators, data set filters and normalized additions.

Table I: Quantitative statistical indicators of the student interactions in forums.

Forum conversations started	Forum messages sent	Replies to student interactions
$N_{thrd} = \sum_i^n(x_i)$; x number of threads started on day i and n a set of days in the experience	$N_{msg} = \sum_i^n(x_i)$; x number of messages sent on day i and n a set of days in the experience	N_r_{thrd} = number of messages in the thread started by user
M_{thrd} = average (N_{thrd})	M_{msg} = average (N_{msg})	M_r_{thrd} = N_{reply_thrd} / N_{thrd}
V_{thrd} = variance (N_{thrd})	V_{msg} = variance (N_{msg})	N_r_{msg} = number of replies
L_{thrd} = N_{thrd} / $\sqrt{V_{thrd}}$	L_{msg} = N_{msg} / $\sqrt{V_{msg}}$	M_r_{msg} = N_{reply_msg} / N_{msg}

COLLABORATION STUDENTS ASSESSMENTS

The collaborative learning experience started on February the 21st 2011. 100 students signed up for the collaborative learning experience. 43 of them finished the 1st phase and 15 teams were created. The 2nd phase started on March the 10th 2011 and finished on April the 17th 2011. All along the 2nd phase the quantitative statistical indicators were measured and the students collaboration metric were calculated on a daily basis. Collaboration assessments were displayed to 11 teams and statistical indicators were displayed to 6 teams out of them. 4 teams made up the control group. From the lessons learnt in previous experiences we opted for simplifying displayed results (see Fig. 1).

Collaboration Metric				Statistical Indicators			
Name	Metric	Ranking	Date	Name	N_{thrd}	N_{msg}	N_r_{thrd}
Almudena Cárdenas Marqués	2.21064	3rd of 44th	2011-04-17	Almudena Cárdenas Marqués	7	60	93
Alberto Andrés Vizán	1.71229	5th of 44th	2011-04-17	Alberto Andrés Vizán	5	61	51
Jose García Rodríguez	0.892001	36th of 44th	2011-04-17	Jose García Rodríguez	0	35	0
Maximum	2.7971			Maximum	17	66	93

The metrics of the collaboration is a quantitative indicator that indicates in an approximate way the individual collaboration. The aim is to help you notice of the collaborative work to improve it.

In this table it reports of four indicators of the interactions realized in the team forum. The aim is to help you notice of the individual communications inside the team.

Fig. 1. Assessments displayed to team-members.

FUTURE ANALYSIS

Students are currently facing the final exam and they are answering an evaluation questionnaire. From this data we will be able to compare the usefulness of the metric and displaying strategies, and the expected improvements with respect to previous collaborative learning experiences, as it was reported in (Anaya & Boticario, 2011).

REFERENCES

- A. R. Anaya, And J. G. Boticario, 'Ranking Learner Collaboration According To Their Interactions,' The 1st Annual Engineering Education Conference (Educon 2010), Madrid, Spain, Ieee Computer Society Press, 2010.
- A. R. Anaya And J. G. Boticario, 'Application Of Machine Learning Techniques To Analyze Student Interactions And Improve The Collaboration Process,' Special Issue Of Expert Systems With Applications On Computer Supported Cooperative Work In Design. Volume 38, Issue 2, February 2011, Issn 0957-41742.
- T. Bratitsis, A. Dimitracopoulou, A. Martínez-Monés, J. A. Marcos-García And Y. Dimitriadis, 'Supporting Members Of A Learning Community Using Interaction Analysis Tools: The Example Of The Kaleidoscope Noe Scientific Network,' Proceedings Of The Ieee International Conference On Advanced Learning Technologies, Icalt 2008, 809-813, Santander, Spain, July 2008.
- L. P. Dringus And E. Ellis, 'Using Data Mining As A Strategy For Assessing Asynchronous Discussion Forums,' Computers & Education, 45(2005), 140-160.E.
- Gaudioso, M. Montero, L. Talavera And F. Hernandez-Del-Olmo, 'Supporting Teachers In Collaborative Student Modeling: A Framework And An Implementation,' Expert Systems With Applications, 36 (2009) 2260-2265.
- D. W. Johnson And R. Johnson, 'Cooperation And The Use Of Technology,' In D. Jonassen (Ed.). Handbook Of Research On Educational Communications And Technology. 785-812, 2004.
- Swan, K., Shen, J., Hiltz, S.R.: 2006. Assessment And Collaboration In Online Learning. Journal Of Asynchronous Learning Networks, Vol. 10, No. 1., Pp. 45

A Java desktop tool for mining Moodle data

R. PEDRAZA-PEREZ, C. ROMERO AND S. VENTURA
University of Cordoba, Spain

This paper shows a Java desktop Moodle mining tool oriented not only for experts in data mining but also for newcomers like instructors and courseware authors. This tool has a wizard with a simple interface based on big friendly buttons and default options to facilitate the execution of all the steps in the data mining process.

Key Words and Phrases: Educational Data Mining, Moodle, Data Mining Tool, Wizard.

1. INTRODUCTION AND DESCRIPTION OF THE MINING TOOL

General data mining tools today range from the commercial, such as DBMiner, SPSS Clementine, SAS Enterprise Miner, IBM Intelligent Miner, to open sources, like Weka and RapidMiner. However, all these tools are not specifically designed for pedagogical/educational purposes so they are cumbersome for an educator to use because they are normally designed more for power and flexibility than for simplicity (Romero and Ventura, 2010). In order to resolve this problem, we have developed a Java desktop data-mining tool specifically for mining Moodle data. We have chosen Moodle because it is one of the most frequently used Learning Management Systems (LMS). This tool has been implemented in Java using the KEEL framework (Alcalá et al., 2009) which is an open source framework for building data mining models. Our Moodle mining tool provides a wizard that has a simple interface based on big friendly buttons (see Figure 1) to facilitate its execution.



Fig. 1. Main window of the Moodle mining tool interface.

As Figure 1 shows, our wizard guides users through the most common steps involved in any knowledge discovery process, which are described below.

- *To create data files.* First of all, users must create data files starting from a Moodle database or log files provided by Moodle. Then, they can choose from among different options: creating a data file from one particular Moodle table, creating a summary data file from different Moodle tables, or creating a data file

from one Moodle log file. The summarization option allows for the integration of all the most important student information during a full course, since student and interaction data are spread over several Moodle tables. At the end of this process, a text data file is saved/created using the KEEL data format (Alcalá et al., 2009).

- *To pre-process data files.* This module allows users to modify KEEL data files. The application provides several pre-processing tasks such as: discretization (to divide numerical values into categorical values), edition (to add or delete columns or rows, for example to add students' final marks) and division (to split data file into training and test data files).
- *To visualize data files.* This module will show (it is now under construction) different types of graphical representations of the aforementioned data.
- *To mine data files.* Finally, users can apply data mining techniques. Our mining tool provides a large number of algorithms grouped into three methods/tasks: classification, regression, and association. Firstly, users have to select the desired data mining method and then one of the available algorithms. Our wizard always recommends one algorithm and its parameters by default for each method in order to facilitate its usage/execution for beginners. Users can change the current selection to any other algorithm and change its parameters. Figure 2 shows the execution of the C4.5 algorithm over a summary data file and 10-fold cross-validation. Any resulting files (.tra and .test files that contain partial classification results, and the .txt file containing the model obtained) can be selected and visualized easily (see Figure 2 on the right hand side). In this example, the model obtained is a decision tree that uses IF-THEN rules and also shows a summary with the number of nodes and leaves on the tree, the number and percentage of correctly and incorrectly classified instances, etc.



Fig. 2. Some windows in the data mining wizard.

ACKNOWLEDGMENTS

This research is subsidized by projects of the Regional Government of Andalucía and the Ministry of Science and Technology, P08-TIC-3720 and TIN2008-06681-C06-03 respectively, and FEDER funds.

REFERENCES

- ALCALÁ, J., SÁNCHEZ, L., GARCÍA, S., DEL JESÚS, M. ET AL. 2009. KEEL a software tool to assess evolutionary algorithms to data mining problems. *Soft Computing*, 13(3), 307-318.
 ROMERO C, VENTURA, S. Educational Data Mining: A Review of the State-of-the-Art. 2010. *IEEE Transactions on Systems, Man and Cybernetics, part C: Applications and Reviews*, 40(6), 601-618.

Using data mining in a recommender system to search for learning objects in repositories

A. ZAPATA-GONZALEZ, Autonomous University of Yucatán, México

V.H. MENENDEZ, Autonomous University of Yucatán, México

M.E. PRIETO-MÉNDEZ, University of Castilla la Mancha, Spain AND

C. ROMERO, University of Cordoba, Spain

This paper describes a hybrid recommender system for personalizing the search for Learning Objects (LOs) by using data mining. In fact, a nearest-neighbors based approach has been used to discover the most similar LOs and users, and rule mining used to discover LOs that have been downloaded together by similar users.

Key Words and Phrases: Learning Objects, Hybrid Recommender Systems, Association rule mining.

1. INTRODUCTION AND DESCRIPTION OF THE RECOMMENDER SYSTEM

This paper proposes a hybrid recommendation method to assist users (normally instructors) in personalized searches for Learning Objects (LOs) in repositories. The method proposed (see Figure 1) provides a ranking of LOs, starting from a query by applying the following three phases: preselecting LOs, applying filters and ranking LOs.

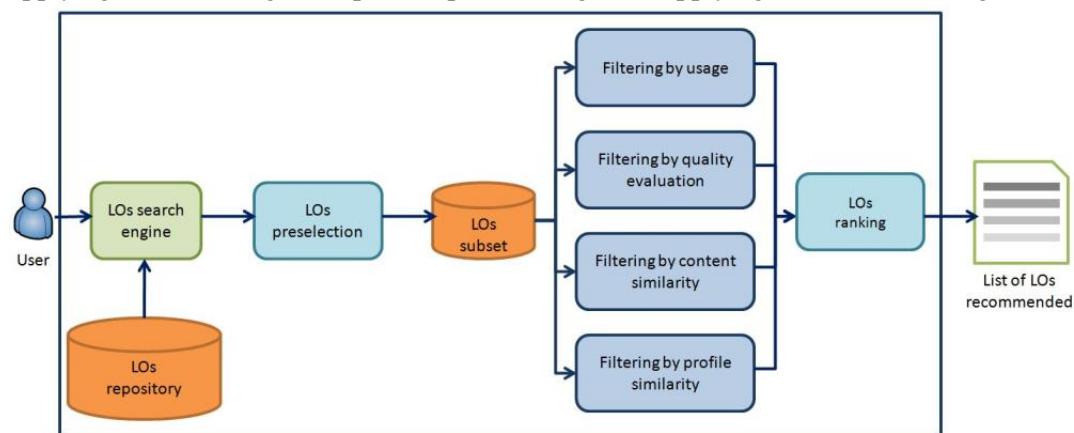


Fig. 1. Proposed architecture for the LO hybrid recommender method.

Firstly, a user does a query using the search engine based on keywords and/or values of some relevant metadata associated with LOs in order to preselect only a subset of LOs that contain the desired contents. Then four filtering or recommendation techniques are applied in parallel: filter by usage (with a ranking of the most commonly downloaded LOs), filter by evaluation (to rank according to the best LOs evaluated), filter by content similarity (ranked according to the most similar LOs) and filter by profile similarity (according to the most similar users/authors). The last two filters use the Nearest Neighbors approach (Ricci et al., 2011) that is like a lazy learner classification algorithm. For a given LO or user, it compares the LO's or user's attributes to the rest of the LO

Authors' addresses: A. Zapata-Gonzalez, V.H. Menendez, Autonomous University of Yucatán, Mexico, E-mail: zgonzal@uady.mx, mdoming@uady.mx; M.E. Prieto-Méndez, University of Castilla la Mancha, Department of Computer Science, Spain, E-mail: Manuel.Prieto@uclm.es; C. Romero, Department of Computer Science, University of Cordoba, Spain, E-mail: cromero@uco.es.

subsets or their authors by measuring their similarity. This metric lets us determine the degree of similarity between two LOs (by comparing metadata values) or between two users (by comparing values in profile attributes). Then, the outputs (partially ordered LO sets) of the four filters are combined in the final LO ranking. This ranking is obtained by a weighted composition in order to be able to attach more or less importance to one filter or another. This hybrid recommender method has been implemented into the AGORA repository (Prieto et al., 2008) and is called DELPHOS. Figure 2 shows an example of a ranking of LOs obtained by DELPHOS.

Learning Object	Rating	Statistics	Why?	Related	Rate
The Greek alphabet. Learning Greek		[15 [10 [8]	4 2		
Writing Greek. Some examples and common errors		[5 [7 [4]	6 4		
Greek Art		[2 [2 [1]	2 1		

Fig. 2. An example of search results found by DELPHOS.

For each LO recommended, DELPHOS shows the user the following information (see Figure 2): type, full name and description of the LO, rating (by using a five start scale as graphical representations of the numerical value), some statistics (total number of downloads, visualizations and evaluations), a brief explanation about why this particular object has been recommended (by using the values obtained in each filter), a list of related users (who have rated/evaluated the LO highly) and a list of related LOs (that have been downloaded together). Association rule mining has been applied to reveal these relationships (or item-item correlations) among current recommended LO and other LOs which have often been downloaded together by users like the one who is currently carrying out the search. In fact, Apriori Predictive (Scheffer, 2005) is used because it is a parameter-free association algorithm that does not require the user to specify either the minimum support threshold or confidence values and is therefore easier for a non-expert in data mining to manipulate. Finally, the user can visualize or download one or several of the lists of recommended LOs (and provide implicit information) and then he/she can rate them using a questionnaire (and provide explicit information). All the information provided can be used to evaluate the accuracy of the recommendations.

ACKNOWLEDGEMENTS

This research has been partially supported by the Regional Government of Andalucía P08-TIC-3720 project; TIN2010-20395 FIDELIO project, MEC-FEDER, Spain, PEIC09-0196-3018 SCAIWEB-2 excellence project, JCCM, Spain and POII10-0133-3516 PLINIO project, JCCM, Spain; The National Council of Science and Technology (CONACYT, México) and the Council of Science and Technology of Yucatán State (CONCyTEY, México).

REFERENCES

- PRIETO, M. E., MENENDEZ, V., SEGURA, A. VIDAL, C. 2008. A Recommender System Architecture for Instructional Engineering. *Emerging technologies and Information systems for knowledge society*, 314-321.
- RICCI, F., ROKACH, L., SHAPIRA, B., KANTOR, P.B. 2011. Recommender Systems Handbook. Springer.
- SCHEFFER, T. 2005. Finding association rules that trade support optimally against confidence. *Intelligent Data Analysis*, 9(4), 381–395.

E-learning Web Miner: A data mining application to help instructors involved in virtual courses

D. GARCÍA-SAIZ

University of Cantabria, Spain

AND

M.E. ZORRILLA PANTALEÓN

University of Cantabria, Spain

In this demo we present a data mining application, called E-learning Web Miner (EIWM), which aims to help instructors involved in distance education to discover their students' behavior profiles and models about how they navigate and work in their virtual courses which are offered in Learning Content Management Systems. The main characteristic is that the users do not require data mining knowledge to use EIWM, they only have to send a data file according to one of the templates provided by the application and request the results. The application carries out the Knowledge Discovery in Database (KDD) process itself. Furthermore, the application provides an interface based on web services which allow its integration and use by any external software.

Key Words and Phrases: Educational Data Mining, Learning Content Management Systems, Towards Parameter-free Data Mining, Data Mining Web Services

1. INTRODUCTION

Our goal is to describe a data mining application, called E-learning Web Miner (EIWM), implemented in the University of Cantabria which assists the instructors involved in virtual education in their teaching activity in the sense that this application helps instructors to discover on the one hand, the distance students' behavior based on their navigation and demographic data and on the other hand, how they surf and work in a distance course offered in an e-Learning platform such as Moodle or Blackboard. The patterns and models which it generates help instructors to better understand the learning process and to analyze the course organization effectiveness (design, tasks, resources used, and so on).

EIWM has been developed following a Service Oriented Architecture (SOA) and implemented by means of web services (a more extended description is found in [Zorrilla and García-Saiz 2011]) with the aim of it being easily extended and choreographed with other web services offered by third parties. The two main characteristics of this Data Mining Application are that: it offers a set of templates which resolve some of the common questions of instructors involved in virtual courses and it is configured to be used by non-data mining experts. As far as we know there is not a similar tool available and a clear necessity for a tool which addresses this issue exists according to the extensive research activity which is being carried out in this field [Romero and Ventura 2010].

2. WORKING WITH E-LEARNING WEB MINER

EIWM currently offers instructors, through an easy web-based interface, the possibility of

Authors' addresses: D. García-Saiz; M. E. Zorrilla Pantaleón. Department of Mathematics, Statistics and Computation, University of Cantabria. Avda. De los Castros s/n, 39005, Santander, Spain E-mails: garciasad@unican.es and zorrillm@unican.es

answering three different questions: What kinds of resources are frequently used together (forum, mail, etc.) in each learning session; What is the student session profile; and What is the profile of the students who enroll in the course.

The end-user chooses the question he wants the application to answer and indicates where the data file to process is stored or selects the course if this is hosted in Blackboard, the official Learning Content Management System of our University. Next, EIWM carries out the KDD process and returns the result to the instructor in textual and graphic format.

The KDD process to answer each question is defined by means of templates. The templates contain the input attributes which the data file must have as well as the pre-processing tasks, the mining algorithms and the parameter-setting which is adequate for obtaining the patterns. These templates have been defined and validated by means of a previous experimentation carried out in several virtual courses taught at the University of Cantabria.

As an example, we describe briefly the student profile template. The objective of this template is to group students according to their activity in the e-learning platform and their demographic data. The template establishes as input parameters: gender, age, number of sessions in the course, time spent in the course, average sessions per week, average time spent per week; and as data mining algorithms for obtaining the patterns: EM and SimpleKMeans from Weka. EM algorithm is used to determine the number of clusters with which the SimpleKMeans algorithm will be executed. We generate the patterns with SimpleKMeans because it is one of the most used in practical problems, its execution is quick and furthermore, the results which it offers are easy to understand statistically and graphically. Before the mining process starts, a pre-processing task is carried out in order to evaluate the quality of the data for the process, for example, to eliminate correlated or highly unbalanced data or outliers.

3. CURRENT AND FUTURE WORK

Currently, we are extending EIWM with a new template. The goal is to predict the final mark which the student will obtain in the course according to the global activity the student has carried out during the course. Likewise, our group is improving *yacaree*, a parameter-free rule miner [Balcazar 2011], to respond adequately to educational data needs [Zorrilla et al. 2011]. Once this task is completed, the algorithm will be used by the application. Next, we will continue adding new templates and, in the near future, our objective is to deploy our application as a service in Cloud Computing, offering this as a Software-as-a-Service (SaaS). According to [Software & Information Industry Association 2006], SaaS is a model for software delivery that allows lower total cost of ownership and better return on investment for subscribers.

4. REFERENCES

- BALCÁZAR, J. L. 2011. Parameter-free association rule mining with *yacaree*. In EGC, A. Khenchaf and P. Poncet (Eds) Revue des Nouvelles Technologies de l'Information Series, vol. RNTI-E-20. Hermann-Editions, 251–254.
- ROMERO, C. AND VENTURA, S. 2010. Educational Data Mining: A Review of the State-of-the-Art. IEEE Transaction on Systems, Man, and Cybernetics, Part C: Applications and Reviews. Vol 40. Issue 6. Pages 601 – 618.
- SOFTWARE & INFORMATION INDUSTRY ASSOCIATION. 2006. Software-as-a-Service: A Comprehensive Look at the Total Cost of Ownership of Software Applications. Retrieved May, 2011 from <http://www.winnou.com/saas.pdf>
- ZORRILLA, M.E., AND GARCÍA-SAIZ, D. 2011. A Service Oriented Architecture to provide data mining services for non-expert data miners. Accepted. To appear in Decision Support Systems.
- ZORRILLA, M.E., GARCÍA-SAIZ, D., AND BALCÁZAR, J.L. 2011 Towards Parameter-Free Data Mining: Mining Educational Data with *yacaree*. Submitted to EDM 2011

Computerized Coding System for Life Narratives to Assess Students' Personality Adaption

Q. HE, B.P. VELDKAMP, G.J. WESTERHOF

University of Twente, the Netherlands

The present study is a trial in developing an automatic computerized coding framework with text mining techniques to identify the characteristics of redemption and contamination in life narratives written by undergraduate students. In the initial stage of text classification, the keyword-based classifier algorithm – product score model – performed more sensitive in finding the “change” elements in life stories comparing to the Decision Tree and Naïve Bayes models. The verbal features of life narratives were also discussed to enhance the classification accuracy further in assessing students’ personality adaption.

Key Words and Phrases: Text mining, text classification, personality adaption, life narratives

1. INTRODUCTION

In the first decade of the 21st century, narrative approaches to personality have moved to the center of the discipline of personality psychology (McAdams, 2008). Besides the traditional qualitative methods and case studies, an increasing number of quantitative studies have been used in narrative research. For instance, McAdams and his colleagues developed objective coding systems for assessing narrative tone, themes of agency and communion in life-narrative accounts, redemption and contamination sequences, and goal articulation (see www.sesp.northwestern.edu/foley). However, such coding systems are specified for manual rating rather than computer-based assessment, which results in heavy workloads for the human raters.

The present study is to develop an automatic computerized coding framework corresponding to McAdams’s manual coding system in identifying the characteristics of redemption and contamination based on a sample of life narratives written by the undergraduate students. Redemption and contamination are the two most important sequences to reveal the “change” tendency in people’s emotional well-being through writing. In a redemption sequence, a demonstrably “bad” or emotionally negative event or circumstance leads to a happy outcome, whereas in a contamination scene, a good or positive event or state becomes bad or negative. There are two specific objectives in the current research: (1) to develop a text mining model to accurately categorize the students’ life narratives; and (2) to extract verbal features in life narratives to enhance the prediction further.

2. METHOD

The collected life stories were written by 271 undergraduate students in the United States, containing 656 life stories in total. The target classification was labeled into four categories: redemption (RED), contamination (CON), both of redemption and contamination (BOTH), neither redemption nor contamination (NEITHER). Three experienced human raters ($\kappa=0.67$) labeled each story corresponding to the manual coding system. These results were set as the “standards” in the performance evaluation.

Based on a common feature – the “change” tendency in both redemption and contamination life stories, we constructed a hierarchical classification framework shown in the Figure 1. On the first stage, all the input were divided into two groups, “change” and “nochange”, by identifying the presence and absence of “change” features in the stories. Based on these preliminary results, a more detailed classification was conducted

Authors’ addresses: Q. He, Department of Methodology, Measurement and Data Analysis, University of Twente, the Netherlands. E-mail: q.he@gw.utwente.nl; B.P. Veldkamp, Department of Methodology, Measurement and Data Analysis, University of Twente, the Netherlands. E-mail: b.p.veldkamp@gw.utwente.nl; G.J. Westerhof, Department of Health and Technology, University of Twente, the Netherlands; E-mail: g.j.westerhof@utwente.nl

on the second stage to label each story based on the patterns extracted for redemption and contamination.

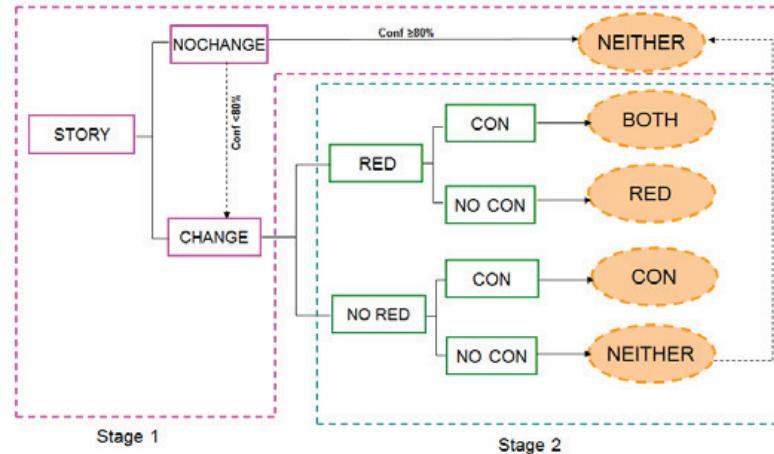


Fig 1: The hierarchical classification framework for life story computerized coding system

The dataset was split into two sets by a random selection of 70% as training set and 30% as test set. During training, the most discriminating keywords to determine the story with or without the tendency of “change” were extracted using the Chi-Square selection algorithm (Oakes et al., 2001). Afterwards, pairs of feature sets and labels were fed into a machine learning algorithm to “learn” their relationship and generated a classifier model. During prediction, the same feature extractor was used to convert new inputs to feature sets. These features were then fed into the “trained” classifier model, which predicted the most likely label for the new input. We utilized two standard machine learning algorithms, Decision Tree and Naïve Bayes in the textual classification process and also constructed an alternative model named product score, which is analogous to the log-likelihood ratio test.

3. RESULTS

The relationship between computerized coding method at various thresholds and the gold standard was assessed through 2-by-2 tables with four indicators, precision, recall, overall correct classification, and F1 score. In the initial stage of text classification, the product score model with the cutoff threshold at (-4) yielded the highest sensitivity in finding the “change” elements in life stories compared with the Decision Tree and Naïve Bayes models, with a precision rate around 85% but sacrificing the recall rate around 50%. Further analysis for the second stage is still in process.

4. CONCLUSIONS

The present study represents an initial development of text classifier model for computerized coding system in life narratives. This trial demonstrated that the text mining technique was very promising in assessing the people’s personality adaption with verbal features. Some latent semantic analysis and sequence measurement model are also expected to add in the future research.

REFERENCES

- McAdams. D. P. 2008. Personal narratives and the life story. In John, Robins, & Pervin (Eds.), *Handbook of Personality: Theory and research (3rd Ed.)* NY: Vilfert Press.
 Oakes, M., Gaizauskas, R., Fowkes, H., Jonsson, A., Wan, V. & Beaulieu, M. 2001. A method based on Chi-Square test for document classification. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 21)*, 440-441.

Partially Observable Sequential Decision Making for Problem Selection in an Intelligent Tutoring System

EMMA BRUNSKILL AND STUART RUSSELL, University of California, Berkeley

A key part of effective teaching is adaptively selecting pedagogical activities to maximize long term student learning. In this poster we report on ongoing work to both develop a tutoring strategy that leverages insights from the partially observable Markov decision process (POMDP) framework to improve problem selection relative to state-of-the-art intelligent tutoring systems, and evaluate the computed strategy in the classroom. We also highlight some of the challenges in data mining related to automatically constructing pedagogical strategies.

1. INTRODUCTION

We are interested in creating algorithms to automatically construct adaptive pedagogical strategies for intelligent tutoring systems. Our approach assumes as input a student model, which can be learned from educational data. In this paper we assume that such a model is provided, and focus on the task of computing adaptive policies given the student model; however, in the future we plan to learn the student model from data and at the end of this document we briefly sketch some of the data mining challenges involved.

Different pedagogical activities vary in their instructional effectiveness, and in the information they provide about the student's underlying understanding. Our approach explicitly reason about pedagogical activity features when computing a conditional sequence of activities to provide to a student. We pose tutoring policy creation as an instance of Partially observable Markov decision process (POMDP) planning [Sondik 1971]. The student's knowledge state is represented as a set of hidden variables. The value of these variables can change in response to a pedagogical activity, and these hidden variable values may be probed, or partially observed, such as by posing a test question to the student). A probability distribution over possible knowledge states of the student is maintained and updated using Bayesian filtering: examples of prior work using this approach to student modeling include Corbett and Anderson [1995] and Conati et al. [2002].

There has been some very recent interest in using POMDPs to incorporate Bayesian models of student learning into algorithms for constructing sequential pedagogical policies: see Brunskill and Russell [2010], Rafferty et al. [2011], Tehocharous et al. [2009], and Folsom-Kovarik et al. [2010]. While this initial work is encouraging, none of these prior approaches have performed experiments with standard school curriculum, nor evaluated their approaches relative to existing, high-performing problem selection strategies used in intelligent tutoring systems. Here we report on an ongoing effort to examine whether POMDP planning can further improve the effectiveness of an existing intelligent tutoring system for high school math.

2. APPROACH

We follow Corbett and Anderson [1995] and model the student knowledge state as whether the student has mastered or not mastered a set of N skills. Though the algorithm we developed can be applied more broadly, as our goal was to compare our approach to existing state of the art methods, and to conduct classroom studies, we focus on developing an approach for the Algebra I tutor, produced by the software company Carnegie Learning. In this tutor students do a number of different interactive exercises, and our goal was to adaptively select the best exercise to give to the student, given our Bayesian estimate of the student's

underlying knowledge state, in order to help the student learn the most in a limited amount of time. The possible set of exercises is large, and, if an individual exercise consisted of K skills, there were 2^K possible observations that could be received after the student completed the exercise, corresponding to whether or not the student got each skill correct. Standard approaches to solving POMDPs struggle to scale to domains with large numbers of (pedagogical) actions to select among, and a large set of possible observations. We first explored in simulation using a prior approach by Brunskill and Russell [2010], but the high number of observations possible after a single exercise resulted in challenges for that particular method.

Due to space limitations we only briefly sketch out the algorithm we developed. We computed a depth one POMDP forward search tree from a set of initial knowledge states, and used a heuristic estimate of the value at the leaves of the tree: roughly, this heuristic estimate corresponds to how well we expect the student to perform if we used a heuristic method to select pedagogical activities from this point onwards for a fixed number of activities. These heuristic values were calculated assuming that the student would complete a fixed number of pedagogical activities before receiving a post test.¹ The forward search tree values were cached and combined to construct a look up table that was used to select pedagogical activities during tutoring.

3. ONGOING WORK

We have done some preliminary evaluation of our approach in a classroom study with ninth grade students. However, there were several limitations of this initial study. We are currently focusing on using simulation studies to better evaluate and modify our proposed algorithm.

So far the only pedagogical activities considered as part our algorithm were problem exercises. The POMDP framework should be particularly helpful in trading off among activities with variable components of instruction and evaluation. We hope to soon consider a more diverse set of pedagogical activities types.

Finally, the approach we have described depends critically on an input set of parameters which describe the student model. There has recently been some promising work by [Chi et al. 2010] which learned Markov transition dynamics from a set of data collected using random tutoring strategies, and used that to infer a successful tutoring strategy. However, most existing data on student-tutoring system interactions will involve a non-random strategy. This can lead to sparsity in the collected data as many pedagogical activities will never have been tried from particular student states. Constructing good models in these scenarios, particularly when the student state is modeled as hidden, remains an interesting unfinished challenge.

REFERENCES

- BRUNSKILL, E. AND RUSSELL, S. 2010. RAPID: A reachable anytime planner for imprecisely-sensed domains. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*.
- CHI, M., VANLEHN, K., AND LITMAN, D. 2010. Do micro-level tutorial decisions matter: Applying reinforcement learning to induce pedagogical tutorial tactics. In *Proceedings of the International Conference on Intelligent Tutoring Systems*.
- CONATI, C., GERTNER, A., AND VANLEHN, K. 2002. Using Bayesian networks to manage uncertainty in student modeling. *Journal of User Modeling and User-Adapted Interaction* 12, 371–417.
- CORBETT, A. AND ANDERSON, J. 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction* 4, 253–278.
- FOLSOM-KOVARIK, J., SUKTHANKAR, G., SCHATZ, S., AND NICHOLSON, D. 2010. Scalable POMDPs for diagnosis and planning in intelligent tutoring systems. In *AAAI Fall Symposium on Proactive Assistant Agents*.
- RAFFERTY, A., BRUNSKILL, E., SHAFTO, P., AND GRIFFITHS, T. 2011. Faster teaching by POMDP planning. In *to appear in Proceedings of the 15th international conference on Artificial Intelligence in Education (AIED)*.
- SONDIK, E. 1971. The Optimal Control of Partially Observable Markov Processes. *DTIC Research Report AD0730503*.
- TEHOCHAROUS, G., BECKWITH, R., BUTKO, N., AND PHILIPPOSE, M. 2009. Tractable pomdp planning algorithms for optimal teaching in "SPAIS". In *IJCAI PAIR Workshop*.

¹Ultimately we are interested in scenarios in which there is a fixed amount of time to provide teaching. Here we approximated a fixed amount of time by modeling a fixed number of problems. In reality different students take varying amounts of time, as do different types of problems.

Mining Teaching Behaviors from Pedagogical Surveys

J. BARRACOSA

Instituto Superior Técnico / Technical University of Lisbon, Portugal

AND

C. ANTUNES

Instituto Superior Técnico / Technical University of Lisbon, Portugal

Domain knowledge incorporated in the mining process can improve it, in particular by constraining the patterns discovered. In this paper we show how some domain knowledge (embodied as constraints) combined with sequential pattern mining can be applied to pedagogical surveys, in order to identify frequent teachers behaviors.

1. INTRODUCTION

In the educational context, domain knowledge has been gained for years in the studies performed on each particular aspect of the educational process, and educational data mining is a privileged field to apply domain driven data mining (D3M) (see for example (Antunes2008), where a method for mining students' behaviours is proposed). Mining teaching behaviours however, are much harder, since teachers are at most only evaluated at the end of the curricular unit, usually by pedagogical surveys filled by their students.

In this paper we propose a methodology to mine teaching behaviors, from surveys filled by students, making use of some domain knowledge. In particular we propose a set of constraints to apply to sequential pattern mining, in order to identify how teachers respond to their evaluation, collected from students' surveys.

2. MINING PEDAGOGICAL SURVEYS

Unlike students, teachers are not evaluated in a quantified way. Indeed, their performance is usually appreciated through pedagogical surveys, that in general comprise a long set of questions with a reasonable number of choices, usually divided in topics. We are just interested in surveys with closed-ended questions with multiple choices that follow some order (as in the Likert scale (Likert1932)).

Since our goal is to identify frequent behaviors among teachers, and pedagogical surveys are filled by several students individually, the first step is to get the global evaluation collected for each teacher on each time period. From the application of basic statistics (mean or mode, for example) to the set of individual questionnaires evaluating a single teacher for a single time period, we determine the set of items that characterize teacher's behavior in that period.



Fig. 1. Mining methodology

Since teachers teach along time, a teacher may be represented as a sequence of his performance in each semester, with his performance being represented as an itemset with the results achieved for each question in the pedagogical surveys. With teachers represented as temporal sequences is then possible to apply sequential pattern mining to identify frequent behaviors (Antunes2008) and try to anticipate their evolution Fig. 1.

The next challenge faced is to decide what domain knowledge may be useful on mining teaching behaviors. Indeed, what we need to understand is how teachers evolve along time. In this manner, the most interesting knowledge is the one that can contribute to distinguish among teachers that improve their performance against the ones that do not.

Pushdown automata (PDA) in Fig. 2 establish a set of meta-patterns that can guide the discovery process, reducing the combinatorial explosion on discovered patterns, allowing for the discovery of relevant patterns in the scope of teaching evolution. In each meta-pattern, relations among the values of specific questions are established, both for a single semester and along time.

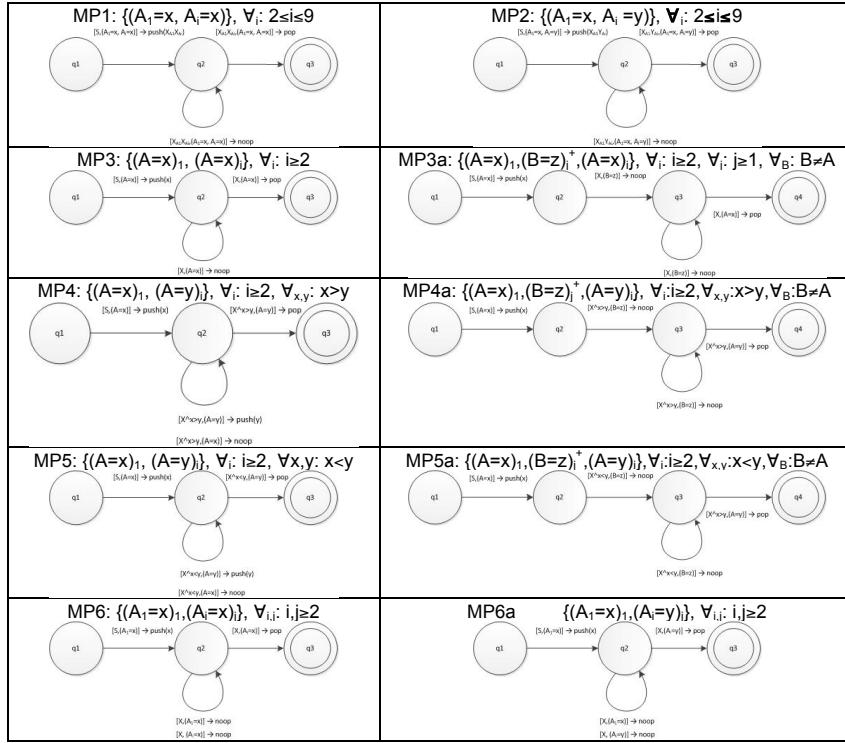


Fig. 2. Meta-patterns to constrain sequential pattern mining process

Each PDA is composed of a stack, a set of states and transitions that are constrained by the current state and entry, but also by the top of the stack – the context (Antunes2008). For example, MP1 catch patterns that have between two and nine variables with the same value in the same semester, for example $(assiduity=good, security=good)_{2011}$. On the other hand, MP3 catch patterns where some variable has the same value for more than two semesters, for example $(assiduity=good)_{2010}(assiduity=good)_{2011}$.

Experimental results on mining real pedagogical surveys are interesting, with the proposed meta-patterns covering all the discovered patterns. The next step is just to enrich each teacher record with his verification of each meta-pattern, creating a more significant training data set, and then train a classifier for anticipating teachers' performance.

REFERENCES

- ANTUNES, C. 2008. Acquiring Background Knowledge for Intelligent Tutoring Systems. In *Proceedings. of International Conference on Educational Data Mining (EDM 2008)*, 18-27.
 LIKERT, R. 1932. A Technique for the Measurement of Attitudes. *Archives of Psychology*, 22(140), 1-55.

Variable Construction and Causal Modeling of Online Education Messaging Data: Initial Results

S. FANCSALI

Carnegie Mellon University, USA
Apollo Group, Inc., USA

Preliminary results are described of search for variable constructions from “raw” student messaging data in an online forum. Variable constructions are judged based upon their contribution to a novel measure of “causal predictability” for a target learning outcome using graphical search procedures for causal modeling.

Key Words and Phrases: variable construction; dimensionality reduction; Bayesian networks; causality; causal discovery; online education

1. INTRODUCTION

Research on causal discovery from observational data using graphical causal models (e.g., Spirtes, *et al.* 2000; Pearl 2000) focuses almost entirely on situations in which data are available at an appropriate unit/level of analysis (i.e., well-defined random variables that capture salient features of interest). However, in many social science settings, “raw” data manifests in log files and other sources whose variables are not suited to straightforward causal interpretations. In online education settings, for example, we might have access to logs of student behaviors, online forum messages, and other aspects of student interactions with courseware. In such cases, the problem of causal discovery is two-fold: in addition to discovering relevant causal relations, we must also discover, from complex, high-dimensional, “raw” data, the very variables that figure in those causal relations.

While there is a great deal of work in statistics and machine learning on dimensionality reduction (feature extraction and selection), little work is aimed at the construction of variables, let alone for causal modeling. Arnold, *et al.* (2006) developed a method for feature discovery with education data, but did not use it for causal modeling. We extend some of those ideas to the problem of causal discovery by simultaneously searching for both suitable constructed variables and resulting graphical causal models. Specifically, we search over different deterministic functions of the underlying “raw” variables and evaluate those constructed variables by informativeness with regards to the underlying causal structure (including suitability for manipulation) in online course environments.

2. DATA AND METHOD

We consider data from 646 learners that used a message forum for an online, graduate-level economics course and consider only raw data from forum logs. With this data, we seek to develop models of the causes of student learning outcomes as measured by instructor-assigned course grades and independently assessed final exams scores. While a plethora of other data sources and (behavioral) causes of these learning outcomes might be considered, the focus on an “impoverished” data set provides an interesting and difficult test of our method.

We iteratively construct and evaluate sets of constructed variables—deterministic functions of base characteristics of student messaging behavior (e.g., length of messages, message counts, etc.). At the first stage, we consider simple functions (max, average, variance, etc.) of these base characteristics and continue at later stages to consider

Author’s addresses: S. Fancsali, Department of Philosophy, Carnegie Mellon University, Pittsburgh, PA, USA. Product Strategy and Development, Apollo Group, Inc., San Francisco, CA, USA. E-mail: sfancsal@andrew.cmu.edu

discretizations as well as interactions among them. At each stage, a variety of strategies can be used to prune the large set of possible constructions to a smaller number.

From any particular set of constructed variables, we search for causal structure using the FCI algorithm (Spirtes, *et al.* 2000). FCI can infer the presence of unmeasured (or “latent”) confounders of observed variables, which are almost certainly present given our (deliberately) “impoverished” data. FCI outputs a partial ancestral graph (PAG), which encodes all of the causal models consistent with the observed data. We deploy a novel method to assess the value of this PAG, and thus the set of constructed variables. We first enumerate all of the directed acyclic graphs (DAGs) that are compatible with the causal explanation provided by the PAG. From each DAG, we build a regression model for a target variable (i.e., learning outcome) based upon the target’s parents in that DAG. We then assess the set of constructed variables by the average R^2 value over all of these regression models (i.e., assuming every DAG represented by the PAG is equiprobable). We seek the set of constructed variables that maximizes “causal predictability” as assessed by this “average causal R^2 ” value.

3. RESULTS

Table I provides a comparison of average causal R^2 values for constructed variables found via the above search procedure compared to those achieved using ad hoc variables previously specified in a pilot study for this data (Fancsali, forthcoming). With this novel analysis, we find substantial improvement in the acquisition of causal knowledge for both of the measured learning outcomes. FCI search over ad hoc constructed variables and target learning outcomes indicates that any relationships between the two are confounded by unmeasured common causes, thus providing no “causally predictive” information (i.e., causal R^2 values of 0.0) with respect to these learning outcomes. Importantly, this new method unambiguously learns that appropriate constructed variables are direct causes of the course grade, recovering aspects of known “ground truth” for the instructed-assigned grade. Search for constructed variables is a ripe area for future research and will likely have many fruitful applications for both predictive and causal modeling. Future work will deploy similar search on data with richer semantic content (e.g., data from intelligent tutoring systems) to ideally lead to more causally interpretable variable constructions.

Table I. Average Causal R^2 Values for Ad Hoc Constructed Variables vs. Constructed Variables from Search

	course grade	final exam
ad hoc construction	0.0	0.0
variable construction search	0.37	0.13

REFERENCES

- ARNOLD, A., BECK, J., AND SCHEINES, R. 2006. Feature Discovery in the Context of Educational Data Mining: An Inductive Approach. *Proceedings of the AAAI2006 Workshop on Educational Data Mining*, Boston, MA, 7-13.
- FANCSALI, S. forthcoming. Variable Construction for Predictive and Causal Modeling of Online Education Data. *Proceedings of the First International Conference on Learning Analytics and Knowledge*. Banff, Alberta, Canada.
- PEARL, J. 2000. *Causality: Models, Reasoning, and Inference*. Cambridge UP.
- SPIRTES, P., GLYMOUR, C., AND SCHEINES, R. 2000. *Causation, Prediction, and Search*. Second Edition. Cambridge, MA: MIT.

The Hospital Classrooms Environments Challenge

CARINA GONZÁLEZ, PEDRO A. TOLEDO

Universidad de La Laguna, Spain.

This poster describes a challenging application field for EDM, in the context of collaborative learning open learner model frameworks. An open and challenging field arises from the situation lived by children suffering a serious illness. The responsibility of the government continuing their schooling process is harder to achieve, since different communities (family, school teachers, hospital teachers, medical doctors, psychologists...) have to be coordinated. An appropriate educational solution, designed for this situation should solve the specific issues that arise in the described situation: Motivation, Isolation, Heterogeneity, Spatial Dispersion, Dynamicity, etc. This poster tries to explain the reasons that makes this context specially interesting for EDM; It proposes a framework to support learning in the cited environment based on the use of Multi-user virtual environments (MUVEs); And it points out some Data Mining fields specially appropriate for the problem of analysis of educational data and interactions among students for the construction of an Open Learner Model (OLM), suitable to for decision support.

Key Words and Phrases: Open Learner Model, CSCL, MUVEs

1. INTRODUCTION

There are still some application fields where the use of EDM is specially promising and challenging, but still few efforts have been reported. This is the case of educational support of children suffering serious illnesses. These students are frequently provided with computers and with access to the Internet. However, there is still a lack of intelligent systems to support their education, adapted to their special circumstances.

The particular variables that arise in the mentioned context are mainly: **Motivation**: It is frequently shown that these children are hardly interested on any topic related with their educational process. This comprehensible attitude may be changed with the appropriated tools, designed with special attention to the user appealing. **Isolation**: Isolation may be mitigated integrating the educational process in a framework of collaborative learning, making available tools and techniques that make possible and necessary the interaction with other students. **Heterogeneity**: The heterogeneity makes it more difficult to teachers to give the proper educational response to the particular circumstances of each student. **Spatial Dispersion**: It cannot be assumed that students are located at the same room at any time of the schooling process. There is not either direct contact with other students and teachers. **Dynamicity**: Finally, all the circumstances around the student may change in short periods of time. Their mood, needs, and skills may be affected severely, and consequently, the systems they interact with should detect changes and adapt themselves immediately to the new circumstances.

All the described issues makes this context at the same time, challenging, since most of the developed approaches does not seem to fit appropriately to this environment, difficult because accurate self-adapting systems should be created, and especially suited for application of EDM, since there is a large amount and diversity of data to be analyzed. This research has been done in the framework of the SAVEH Project (Gonzalez2011), dealing with the development and integration of ICT tools and electronic contents for young students who suffer serious illnesses.

2. EDUCATIONAL SUPPORT FRAMEWORK AND EDM CHALLENGES

Based on the analysis previously made of the context of hospital classrooms, it has been designed an educational support framework. The design has been done with special attention to very diverse issues: The importance of the social component of learning, to create the new tools, new learning materials and applications (Gaudioso2009); The necessity of adapting systems to each student particular needs, using Open Learner Models (OLM) from multiple students (Ahmad2009) and allowing its inspection and the review of the learning process by the students themselves; The convenience of providing effective mechanisms of communication to increase the knowledge about student behavior and incorporating this new knowledge to the OLM for further analysis of students and influence of other roles as hospital teacher, school teacher, classmates, family, nurses, etc.; The power of educational games to improve the results of ITS (Eagle2010), engaging and motivating students in learning (González2008).

Consequently, we have proposed in SAVEH the development of a MUVE videogame with collaborative activities, intended also as an interface for an ITS using OLM. The videogame is connected with a Social Network, and an Educational Platform.

Based on the proposed educational support framework, mentioned above, there have been noticed among others some challenging problems: The detection of the influence of educational social games on student emotional state, may receive special attention. There should be designed techniques to extract and predict student emotions (motivation, isolation, etc.) in relation with the proposed collaborative activities in the framework; The detection of changes in the interests or skills of students that may be originated by their disease evolution; The extraction of models such as: Model of the reasoning process for the solutions of problems, based on social interactions among users in the collaborative activities, Model of influence of the social interactions in the learning process, Model of the emotional state of student in the MUVEs, Model of the social role of each person within a group. To achieve these goals, a wide range of DM techniques are to be applied, including classification, MDP modelling and solving, concept drift analysis, frequent pattern extraction, graph analysis, structured prediction, SNA, etc.

4. CONCLUSIONS

It has been pointed out the main issues affecting the educational process in the context of Hospital Classrooms. It has been stated the major features for the design of educational tools for it and the reasons why it is especially challenging from an EDM perspective.

REFERENCES

- Ahmad, N. & Bull, S. (2009). Learner Trust in Learner Model Externalisations. In V. Dimitrova, R. Mizoguchi, B. du Boulay & A. Graesser (eds.) Artificial Intelligence in Education, IOS Press, Amsterdam, 617-619
- Gaudioso, E., Montero, M., Talavera, L. & Hernandez-del-Olmo, F. (2009). Supporting Teachers in Collaborative Student modeling: A Framework and Implementation, Expert Systems with Applications 36(2), 2260-2265.
- González, C., Toledo, P., Alayón, S., Muñoz, V., & Meneses, D. (2011). Using Information and Communication Technologies in Hospital Classrooms: SAVEH Project. Knowledge Management & E-Learning: An International Journal, Vol.3, Nº1.
- González C.S. & Blanco F. (2008). Integrating an educational 3D game in Moodle. Simulation & Gaming. Vol. 39 No. 3, September 2008 399-413. DOI: 10.1177/1046878108319585.
- Eagle M. & Barnes T. (2010). Intelligent Tutoring Systems, Educational Data Mining, and the Design and Evaluation of Video Games. Intelligent Tutoring Systems Lecture Notes in Computer Science, 2010, Volume 6095/2010, 215-217, DOI: 10.1007/978-3-642-13437-1_23.

Combining study of complex network and text mining analysis to understand growth mechanism of communities on SNS

OSAMU YAMAKAWA

Fukui Prefectural University, JAPAN

TAKAHIRO TAGAWA, HITOSHI INOUE

Kyushu University, JAPAN

KOICHI YASTAKE, TAKAHIRO SUMIYA

Hiroshima University, JAPAN

1. INTRODUCTION

Japanese 7 higher educational institutions in Fukui prefecture started to cooperate with each other in the middle of 2008. The aim of this project called “F-LECCS” is in order to build “One virtual university environment” on a computer network by using open source software such as SNS (OpenSNP), LMS (Moodle), and e-Portfolio (Mahara). This open platform allows all the users to access learning resources across the universities, and enables them to form intercollegiate learning communities as the “Communities of Practice (CoP). The center system of the CoP is the social networking service (F-LECCS SNS) and has started to work on the beginning of April, 2009. On the F-LECCS SNS, over 300 communities and 360,000 logins (370 logins/day) have been made by the end of March, 2011. The CoPs are growing well.

Our research interest is to understand the growth mechanism of communities on a SNS. Therefore, we combine the complex network analysis for understanding how network grows and the text mining analysis for understanding how people communicate.

2. COMPLEX SYSTEM ANALYSIS

Fig 1 shows a sample graph of the friend-network on F-LECCS SNS. On these networks, the network index, i.e. the network density, average degree, clustering coefficient, average path length and assortative mixing by degree, widely used for

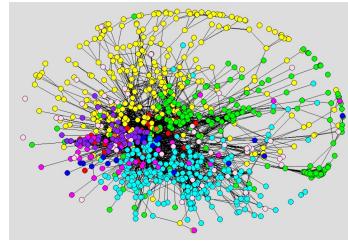


Fig.1 Friend-network on F-LECCS SNS

O.Yamakawa: Center for Arts and Sciences, Fukui Prefectural University, JAPAN, E-mail: yamakawa@fpu.ac.jp
T.Tagawa and H.Inoue: Computing and Communications Center, Kyushu University, JAPAN E-mail: tagawat@cc.kyushu-u.ac.jp, jin@cc.kyushu-u.ac.jp

K.Yasutake: Graduate School of Social Sciences, Hiroshima University, JAPAN, E-mail: ystake@hiroshima-u.ac.jp

T.Sumiya: Information Media Center, Hiroshima University, JAPAN, E-mail: sumi@riise.hiroshima-u.ac.jp

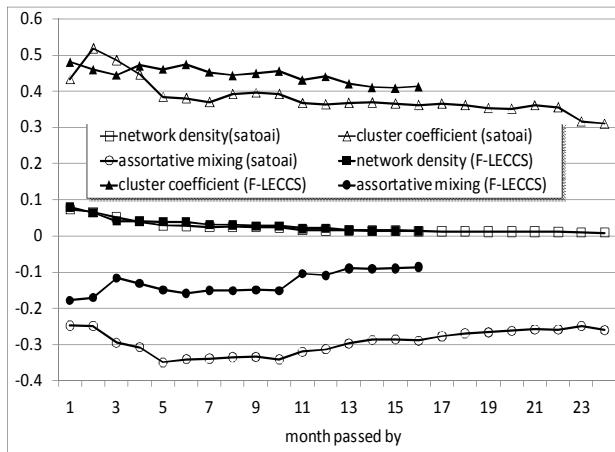


Fig.2 Variation of the network indexes

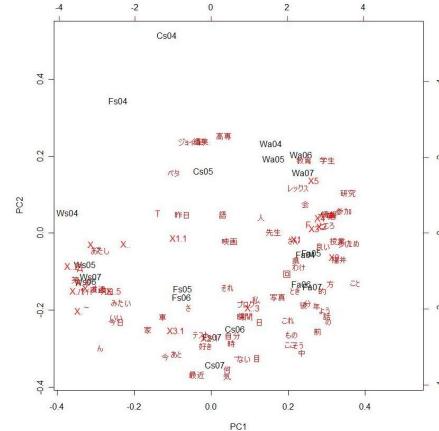


Fig.3 TF-IDF + principal component analysis

complex system analysis(Watts98), have been calculated by network analysis software pajek(Nooy et al. 2005) and igraph package for statistical software R.

In order to compare other friend-network on SNS, “Satoai-SNS” have been also analyzed. Satoai is the cross-university project in Sikoku area in JAPAN. The log data of satoai project for 24 months and F-LECCS project for 16 months data have been analyzed. The variation of the network indexes are shown in Fig.2.

Most indexes have same tendency except the assortative mixing by degree. This result means Satoai’s network contains more star type sub-networks than F-LECCS’s. In another words, F-LECCS has weaker tendency to connect each other depend on degree than Satoai.

3. TEXT MINING ANALYSIS

In order to understand the interaction of communication, the blogs on F-LECCS SNS have been analyzed by text mining method. Fig.3 shows the result of TF-IDF and principal component analysis. We have chosen the 100 important words in blogs by TF-IDF method and analyzed the words by the principal component analysis. Users have been divided into groups by their attributions i.e. universities, teachers or students. In fig.3, red characters are Japanese words and black ones are the groups at certain month. For example, Fs04 indicate University “F”, Students “s”, and April “04”. Fig.3 shows us that the groups Wa, Fa and Ws stay nearly same position. However, the groups Cs and Fs moved from upper part to lower part. This result means that the contents of blogs in some groups did not change and those of the other groups changed for 4 months at the view point of the frequency in the use of important word.

4. CONCLUSION

The complex network analysis is used to understand the formal side of communities and the text mining analysis is for the interactive side of communities. We try to combine the two analyses to understand the growth mechanism of communities on SNS.

REFERENCES

- Watts, D and Strogatz, S. (1998), Collective dynamics of 'small-world' networks., Nature 393, 440-442.
 Nooy, W. de, Mrvar, A., Batagelj, V. (2005). Exploratory Social Network Analysis with Pajek. Structural Analysis in the Social Sciences 27, Cambridge University Press.

Logistic Regression in a Dynamic Bayes Net Models Multiple Subskills Better!

YANBO XU

JACK MOSTOW

Carnegie Mellon University, United States

A single student step in an intelligent tutor may involve multiple subskills. Conventional approaches either sidestep this problem, model the step as using only its least known subskill, or treat the subskills as necessary and probabilistically independent. In contrast, we use logistic regression in a Dynamic Bayes Net (LR-DBN) to trace the multiple subskills. We compare these three types of models on a published data set from a cognitive tutor. LR-DBN fits the data significantly better, with only half as many prediction errors on unseen data.

Key Words and Phrases: Knowledge Tracing, multiple subskills, conjunctive models, Dynamic Bayes Net

1. INTRODUCTION

Knowledge tracing [Corbett and Anderson, 1995] is widely used to estimate from the observable steps that require a skill the probability that the student knows the skill. However, steps that require multiple subskills are problematic.

One approach [Cen et al., 2006] tries to sidestep the problem by modeling each set of subskills as a distinct individual skill, e.g., computing the area of a circle embedded in a figure vs. by itself. However, modeling them as individual skills ignores transfer of learning between them. Another solution [Cen et al., 2008; Gong et al., 2010] models the step as applying each subskill independently, and assigns it full credit or blame for getting the step right or wrong. Then the subskills can be traced independently. This solution assumes that the probability of having the knowledge needed for the step is the product of the probabilities of knowing the subskills. A third solution assigns all the credit or blame to the least known subskill. This solution approximates the probability of having the knowledge required for the step as the minimum of those probabilities instead of their product.

Recently, Xu and Mostow [2011] described a method to trace multiple subskills by using logistic regression in a Dynamic Bayes Net (LR-DBN). It models the transition probabilities from the knowledge state $K^{(n-1)}$ at step $n-1$ to the knowledge state $K^{(n)}$ at step t as logistic regressions over all m subskills:

$$P(K^{(0)} = \text{false}) = \frac{\exp(-\sum_{j=1}^m \beta_j^{(0)} s_j^{(0)})}{1 + \exp(-\sum_{j=1}^m \beta_j^{(0)} s_j^{(0)})}, \quad (1)$$

$$P(K^{(n)} = \text{false} \mid K^{(n-1)} = \text{false}) = \frac{\exp(-\sum_{j=1}^m \beta_{1,j} s_j^{(n)})}{1 + \exp(-\sum_{j=1}^m \beta_{1,j} s_j^{(n)})}, \quad (2)$$

$$P(K^{(n)} = \text{false} \mid K^{(n-1)} = \text{true}) = \frac{\exp(-\sum_{j=1}^m \beta_{2,j} s_j^{(n)})}{1 + \exp(-\sum_{j=1}^m \beta_{2,j} s_j^{(n)})}. \quad (3)$$

Here the indicator variable s_j is 1 if the step requires subskill j , 0 otherwise; $\beta_j^{(0)}$ is the coefficient for subskill j fit at the initial state; and $\beta_{1,j}$ and $\beta_{2,j}$ are other coefficients.

We now compare the prediction accuracy and complexity of the three types of models: conjunctive product, conjunctive minimum, and LR-DBN.

Authors' addresses: Y. Xu, E-mail: yanbox@cs.cmu.edu; J. Mostow, E-mail: mostow@cs.cmu.edu, Project LISTEN, RI-NSH 4103, 5000 Forbes Avenue, Carnegie Mellon University, Pittsburgh, PA 15213-3890, USA. 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. We thank Ken Koedinger for his data and assistance.

2. EXPERIMENTS AND RESULTS

We compare the three models on a published [Koedinger et al., 2010] data set from 123 students working on a geometry area unit of the Bridge to Algebra Cognitive Tutor®. All three models use the same 50 subskills. We fit the models separately for each student on the first half of all of that student's steps, test on the second half, and average the results.

Table I shows the mean fit of each student's data to the three types of model. The values in parentheses show 95% confidence intervals based on standard error calculated from the unbiased weighted sample variance of individual students' accuracies. LR-DBN does best with 92.7% ($\pm 2.0\%$) accuracy. It makes only half as many prediction errors as the other models, neither of which predicts significantly better than the majority class.

Since the data is unbalanced (84.7% of all steps are correct), we also report within-class accuracy. All three models exceed 96% accuracy within the positive class, with no significant differences. In contrast, accuracy within the negative class is highest by far for LR-DBN (72.3%), while the other two models do much worse than (50-50) chance. Confidence intervals are looser within the negative class, both because it is smaller (15.3% of the data vs. 84.7%), and because its accuracy varies more across students.

LR-DBN is less complex than the other two models in that it has fewer parameters for each student. For each student, it has 50 times 3 logistic regression coefficients for the initial state and two transition probabilities, plus two more parameters for *guess* and *slip*, for a total of 152. In contrast, the other two models have $200 = 50$ times 4 parameters for *already know*, *guess*, *slip*, and *learn* since we fit knowledge tracing per subskill per student. Table II compares their total AIC and BIC scores for model complexity.

Future work should compare on other data sets to see if LR-DBN fares best there too, and to alternative models of multiple subskills to see if LR-DBN beats them as well.

Table I. Mean Per-student Fit of Each Type of Model

	Accuracy	Accuracy Within Positive Class	Accuracy Within Negative Class
Conjunctive product	85.0% ($\pm 1.0\%$)	96.5% ($\pm 0.7\%$)	21.2% ($\pm 3.8\%$)
Conjunctive minimum	85.8% ($\pm 1.0\%$)	98.7% ($\pm 1.0\%$)	14.6% ($\pm 2.3\%$)
LR-DBN	92.7% ($\pm 2.0\%$)	96.5% ($\pm 1.3\%$)	72.3% ($\pm 7.8\%$)

Table II. Summed Complexity Measures of Each Type of Model

	AIC	BIC
Conjunctive product	66,402.1	135,581.9
Conjunctive minimum	66,402.1	135,581.9
LR-DBN	60,545.2	114,159.5

REFERENCES

- CEN, H., KOEDINGER, K. and JUNKER, B. 2006. Learning Factors Analysis – A General Method for Cognitive Model Evaluation and Improvement. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, Jhongli, Taiwan, June 26-30, 2006, 164-175.
- CEN, H., KOEDINGER, K.R. and JUNKER, B. 2008. Comparing Two IRT Models for Conjunctive Skills. In *Ninth International Conference on Intelligent Tutoring Systems*, Montreal, 2008, 796-798.
- CORBETT, A. and ANDERSON, J. 1995. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction* 4, 253-278.
- GONG, Y., BECK, J. and HEFFERNAN, N.T. 2010. Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting Procedures. In *Proceedings of the 10th International Conference on Intelligent Tutoring Systems*, Pittsburgh, PA, 2010, V. ALEVEN, J. KAY and J. MOSTOW, Eds. Springer Berlin / Heidelberg, 35-44.
- KOEDINGER, K.R., BAKER, R.S.J.D., CUNNINGHAM, K., SKOGSHOLM, A., LEBER, B. and STAMPER, J. 2010. A Data Repository for the EDM community: The PSLC DataShop. In *Handbook of Educational Data Mining*, C. ROMERO, S. VENTURA, M. PECHENIZKIY and R.S.J.D. BAKER, Eds. CRC Press, Boca Raton, FL, 43-55.
- XU, Y. and MOSTOW, J. 2011. Using Logistic Regression to Trace Multiple Subskills in a Dynamic Bayes Net. In *4th International Conference on Educational Data Mining* Eindhoven, Netherlands, July 6-8, 2011.

Studying problem-solving strategies in the early stages of learning programming

E. CAMBRANES-MARTINEZ

Universidad Autónoma de Yucatán, México

AND

J. GOOD

University of Sussex, United Kingdom

Computer science and engineering undergraduates must develop problem-solving skills, which are particularly important when solving programming problems which use simple algorithms. Ideally these skills are developed in the early stages of learning programming, in particular during the first weeks or months of a student's first course. For complete beginners, problem solving and logical thinking are key to the development of programming skills. This article presents an initial proposal for a study of complete beginner programmers, i.e. students with no prior formal experience of programming. This research is particularly focused on the period in which students learn basic programming concepts to solve problems. The investigation will include the study and analysis of students' interactions with tools and systems during the problem solving process. Data mining techniques will be used to uncover underlying patterns in the interactions. The goal is to exploit these patterns to enhance tutors' teaching processes.

Key Words and Phrases: learning programming, patterns, data mining, programming beginners.

1. INTRODUCTION

Students pursuing a bachelor's degree in computer science or engineering must develop good problem-solving skills if they are to become competent programmers. These skills are best acquired in the early stages of learning programming, in particular during the first weeks or months of a student's first course. Since this competence is essential for the success of future programming courses, the study of the development of this ability is important.

2. RELATED WORK

Most of the research in this field has compared novice and expert programmers, [Bateson et al 1984, du Boulay 1986, Schneiderman 1976, Weiser and Shertz 1983, Webb et al. 1986, Mayer 1987] reporting on a variety of skills and knowledge based on the contrast between these groups. However, in the literature the concept of 'novice programmer' can be ambiguous: in some experiments novice programmers had no experience; while in others they had completed a couple of programming courses. Schneiderman [1976] proposed a classification of programmers in which the lowest category is 'naive', referring to students without any formal instruction in programming. Nowaczyk [1984] argued that knowledge of the approaches or procedures used by skilled programmers could be useful for teaching novice students. Alternatively, tutors could refine and utilise any basic strategies employed by complete beginner programmers.

3. PROPOSAL

In the early stages of learning the basic concepts of structured programming (inputs, outputs, assignments and control flow) a tutor typically uses pseudo code or flow charts. Some tutors prefer flow charts because they involve minimal coding and they also often use interactive tools that can facilitate better comprehension and easier testing. In an experiment conducted with students between 12 and 13 years of age, where interactions were recorded using an interactive tool, Kiesmüller found that it was possible to detect some patterns in the problem-solving process using the chronology of the interactions, [Kiesmüller 2009]. The proposal for this research is based on a study of the problem-solving strategies that first-year undergraduate students use when they solve basic algorithmic problems. It is proposed to use a tool which allows students to create and test flow charts [Rodriguez et al 2009]. This tool will be adapted to record significant interactions during the problem solving process: time taken to complete an exercise, number of blocks used (added or deleted), expressions used in the blocks of the flow chart (conditions or assignments), testing and errors. Every interaction will be recorded with a time stamp to preserve the chronology of the problem-solving process. Based on a sample of 150 students completing 30 algorithmic exercises, a large number of interactions will be recorded. Therefore it will be possible to analyse these interactions using a data mining approach. Initially descriptive data mining techniques will be used to find frequent patterns, associations and classifications. Other patterns can later be extracted using the students' profiles and interactions with other systems such as e-learning platforms. This knowledge will then be used to attempt to improve the didactic practice of tutors when teaching basic programming concepts.

ACKNOWLEDGEMENTS

The authors wish to thank James Jackson for providing valuable comments on this article. This work is supported by Programa de Mejoramiento del Profesorado PROMEP (Mexico).

REFERENCES

- BATESON, A. G., ALEXANDER, R., AND MURPHYA, M. 1987. Cognitive processing differences between novice and expert computer programmers. *International Journal of Man-Machine Studies* 26: 649-660.
- DU BOULAY, B. 1986. Some Difficulties of Learning to Program. *Journal of Educational Computing Research* 2, no. 1: 57 - 73. <http://baywood.metapress.com/link.asp?id=3lfx9rrf67t8uvk9>.
- KIESMÜLLER, U. 2009. Diagnosing Learners' Problem-Solving Strategies Using Learning Environments with Algorithmic Problems in Secondary Education. *ACM Transactions on Computing Education* 9, no. 3 (September): 1-26. doi:10.1145/1594399.1594402.
- NOWACZYC, R. 1984. The relationship of problem-solving ability and course performance among novice programmers. *International Journal of Man-Machine Studies* 21, no. 2 (August): 149-160. doi:10.1016/S0020-7373(84)80064-4. [http://dx.doi.org/10.1016/S0020-7373\(84\)80064-4](http://dx.doi.org/10.1016/S0020-7373(84)80064-4).
- SCHEIDERMAN, B. 1976. Exploratory Experiments in Programmer Behavior. *International Journal of Computer and Information Sciences* 5, no. 2: 123-143. <http://www.springerlink.com/content/m852q12168q76224/fulltext.pdf>
- WEBB, N. M., ENDER, P., AND LEWIS, S. 1986. Problem-Solving Strategies and Group Processes in Small Groups Learning Computer Programming. *American Educational Research Journal* 23, no. 2 (January): 243-261. doi:10.3102/00028312023002243.
- WEISER, M., AND SHERTZ, J. 1983. Programming problem representation in novice and expert programmers. *International Journal of Man-Machine Studies* 19, no. 4 (October): 391-398. doi:10.1016/S0020-7373(83)80061-3.
- MAYER, R. 1997. From Novice to Expert. In *Handbook of Human-Computer Interaction*, 781-795. Elsevier. doi:10.1016/B978-044481862-1.50099-6. <http://dx.doi.org/10.1016/B978-044481862-1.50099-6>.
- RODRÍGUEZ-CÁMARA, V., KU-QUINTANA J., AND CAMBRANES-MARTÍNEZ E.. 2009. Origami: Tool to support novice programmers based on flowcharts. In *Extended Proceedings of the 4th. Latin-American Conference on Human Computer Interaction*, 88-87. Mérida, Yucatán, México: Universidad Autónoma de Baja California.

Brick: Mining Pedagogically Interesting Sequential Patterns

ANJO ANJEWIERDEN and HANNIE GIJLERS, University of Twente

NADIRA SAAB, University of Leiden

ROBERT de HOOG, University of Twente

One of the goals of the SCY project (www.scy-net.eu) is to make (inquiry) learning environments adaptive. The idea is to develop “pedagogical agents” that monitor learner behaviour through the actions they perform and identify patterns that point to systematic behaviour, or lack thereof. To achieve these aims, it is necessary to develop methods to identify relevant patterns, and to associate these patterns with feedback to the learner. Here, we present a tool, called Brick, in which pedagogical researchers can interactively specify global patterns (e.g., “which actions occur between running a simulation and providing a correct answer”). The tool then responds with the matching sub-sequences in the data along with various metrics of interestingness (such as frequency and likelihood). Researchers can then determine which patterns are relevant to include in a pedagogical agent and how the agent should respond when a pattern occurs. We describe the tool, the pattern language and metrics used, and provide examples of applying these to log files of a simulation-based inquiry learning environment.

1. INTRODUCTION

We provide a brief illustrated example of how Brick is used to explore patterns from action sequences derived from a simulation-based inquiry learning environment in which learners collaborated in dyads, using a chat channel for communication [Saab 2005]. The environment provided *assignments* that consisted of a question, a list of possible answers and a simulation learners could manipulate. Action types that are extracted directly from the log files are: ⑤ **start** (an assignment), ⑥ **simulation** (running a simulation), ⑦ **correct** (correct answer to a question), ⑧ **wrong** and ⑨ **end** (of an assignment). The symbol in front of the action type is the “brick” (character) used to build patterns from. Chat communication within dyads has been manually coded capturing the inquiry process [Kuhn et al. 2000]: ⑩ **planning** (domain-related chat about planning a simulation), ⑪ **interpretation** (interpreting results of an experiment), ⑫ **emotional** (non-domain chat about the results), ⑬ **answer** (discussing what answer to give), ⑭ **next** (which assignment is next), ⑮ **off-task**.

2. PATTERN LANGUAGE

Patterns are specified using the following standard notation of regular expressions: ⑯ **wildcard** (matches precisely a single action), ⑰ **any sequence** (any sequence of actions including none), | **alternation**, ? + * **quantification**, and ⑱ **grouping**. Negation (!) is often required to filter unwanted matches. For example, finding all sequences in which the first answer provided by the learner is correct, require that there should not be any intermediate wrong answers: ⑲ **(*) ! (red) correct**. See the figure for the matching sequences for this pattern.

3. METRICS

Running a pattern often results in many different matching sequences. In order to filter “interesting” short sequences several metrics have been proposed in the literature. In statistical natural language processing sequences of two or more words that correspond to some conventional way of expression are called collocations [Manning and Schütze 1999].

This study was conducted in the context of Science Created by You (SCY), which is funded by the European Community under the Information and Communication Technologies (ICT) theme of the 7th Framework Programme for R&D (Grant agreement 212814). This document does not represent the opinion of the European Community, and the European Community is not responsible for any use that might be made of its content. Author’s address: A. Anjewierden, H. Gijlers, R. de Hoog, University of Twente, PO Box 217, 7500 AE Enschede, The Netherlands; email a.a.anjewierden@utwente.nl; N. Saab, Leiden University, PO Box 9555, 2300 RB Leiden, The Netherlands; email nsaab@fsw.leidenuniv.nl.

Recently, D'Mello and colleagues [2010] have proposed a *likelihood metric* for the study of behavioural sequences. The general idea behind both these metrics is that sequences that occur much more (or less) likely than chance are the most interesting.

4. RESULTS AND CONCLUSIONS

The figure illustrates the interface pedagogical researchers can use to explore sequences. The pattern for example, can be used to find the action learners perform after planning an experiment. As the figure shows both the likelihood metric $L(X)$ and collocation Coll give a high value to the most “logical” matching pattern: . The pattern provides a typical example of learner behaviour: after giving a wrong answer, learners don't enter the inquiry learning cycle again, but either give another answer or start discussing what answer to give next. As with other data mining approaches, for example association rules, the output of the pattern search is not always clear cut and requires interpretation by the researcher.

REFERENCES

- D'MELLO, S., OLNEY, A., AND PERSON, N. 2010. Mining collaborative patterns in tutorial dialogues. *Journal of Educational Data Mining* 2, 1–37.
- KUHN, D., BLACK, J., KESELMAN, A., AND KAPLAN, D. 2000. The development of cognitive skills to support inquiry learning. *Cognition and Instruction* 9, 285–327.
- MANNING, C. D. AND SCHÜTZE, H. 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, Massachusetts.
- SAAB, N. 2005. Chat and explore: The role of support and motivation in collaborative scientific discovery learning. Ph.D. thesis, University of Amsterdam.

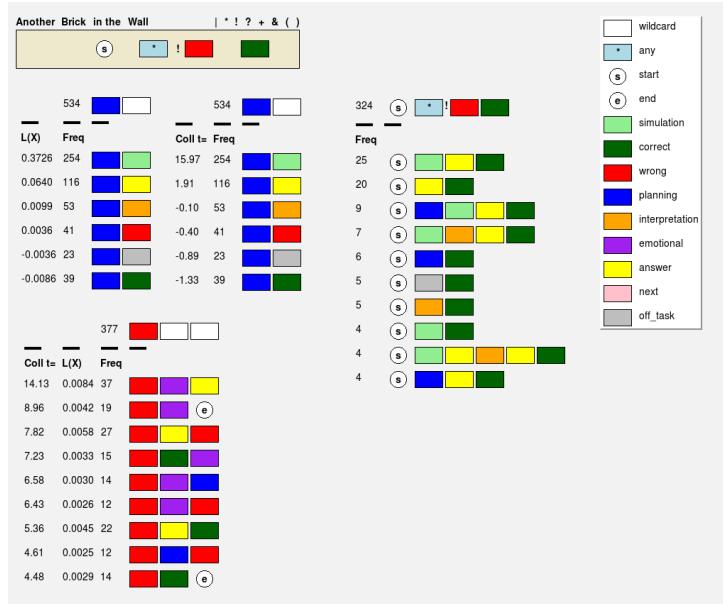


Fig. 1. The Brick user interface consists of three parts. On the right is a palette with bricks representing action types for the given corpus (see Section 1). At the top left is a search box, and above the search box are controls and the meta-characters to create regular expressions from (see Section 2). The largely area displays search results as lists of matching sequences preceded by the values for the metrics. Queries are specified by dragging bricks and meta-characters into the search box.

Intelligent evaluation of social knowledge building using conceptual maps with MLN

L. MORENO, C.S. GONZÁLEZ, R. ESTÉVEZ, B. POPESCU
University of La Laguna, Spain.

Students learning effectively in groups encourage each other to ask questions, explain and justify their opinions, articulate their reasoning, and elaborate and reflect upon their knowledge, and computer tools can help them producing these kind of collaborative learning situations. In this sense, we have developed several tools to support the collaborative learning-process for the Computer Architecture's subject in Computers Science Engineering. This paper present an intelligent tool called SIENA, composed by the open student model, a student conceptual map with multimedia learning nodes (MLN) and Bayesian tests.

Key Words and Phrases: Conceptual Maps, Multimedia, Open Learner Model, CSCL

1. INTRODUCTION

In Computer Support Collaborative Learning (CSCL) there are several interaction computer models that give us the functional descriptions and help us in the understanding, explanation and prediction of the behavior group patrons and to support the learning-process in group (Soller2004). These models can help us to structure the environment where the collaboration is carried out, and to regulate the interactions among students during the learning activities. By the other hand, the graphic visualization systems of student's actions and contributions can increase the awareness of their actions (Ogata2000).

Our problem in the area of Educational Data Mining is the definition of a model of collaboration to incorporate in the system SIENA several processes that allows the intelligent evaluation of knowledge building socially using conceptual maps with MLN. One of the most important aspects in evaluating the process of collaboration is defining the criteria for evaluating such process. An improvement in the collaboration process should bring about the development of end products of higher quality. In order to improve the process of collaboration it is first necessary to evaluate this process with a certain degree of accuracy so that different learning processes taken on by different group of learners can be contrasted. Based on this premise, our proposal includes aspects of the design of the collaborative activities, as well as of the evaluating and monitoring of the collaborative process. In defining the activities, it is necessary to specify the group of people that will make up the group, the required conditions of collaboration, the nature of the activity, the type, and the mechanisms that provide positive interdependence and coordination. Similarly, through the evaluation of the collaboration process, certain weaknesses of the groups can be determined, and thus, supportive mechanisms and feedback can be provided to them. Through a continuous evaluation and monitoring process, the initial conditions can be re-defined, changing certain activities in order to achieve an environment of greater participation and interaction among the members of the group, which can have a positive effect on the collaboration mechanisms.

2. SIENA: INTELLIGENT EVALUATION TOOL THROUGH AUDIOVISUAL CONCEPTUAL MAPS

Researches on collaboration mode shows new trends in the creation of CSCL tools about the social building of knowledge (Fesakis2004; Martínez-Mones2004), SIENA try to

solve the problem related with the information flow into an collaborative environment of knowledge-build and, through the progressive questioning, allows students more aware of the nature of the process of constructing their own knowledge (Jemann2004). More specifically, it is a web-based application used for two purposes, one is to assess the existing abilities and knowledge of a student and the other is to serve as tool which aids self-study and self-evaluation with its main purpose being to support student focused learning (significant learning). In the process of creation of SIENA, both students in collaboration with teachers, were involved in the creation of content and test questions for each of the themes of the Computer Architecture course, providing content to each node of the concept map. The construction and validation of the volume of questions has been a process carried out during three academic years. In the case of the Computer Architecture conceptual map, it consists of 15 nodes and has about 1500 questions. Moreover, we have integrated into the map MLN using the audiovisual format of ULLMedia (González2010). These nodes are created by the teacher. Each MLN is an audiovisual piece of 5-10 minutes about a concept of the map. Moreover, the MLN has a structure of learning object following the IMS LOM standard, thus, can be reutilized in other contexts.

This web-tool use of adaptive tests based on Bayesian networks is developed in Ruby on Rails and employs concept maps in XML format (Moreno2009).

In the knowledge Bayesian estimation we are going to include several collaboration factors to model the social building knowledge (Avouris, 2004). The estimated social knowledge will be the base to build an adequate automatic feedback.

REFERENCES

- NIKOLAOS AVOURIS , MELETIS MARGARITIS , VASSILIS KOMIS (2004). Modelling interaction during small-group synchronous problem-solving activities: the Synergo approach. Proceedings of 2nd International Workshop on Designing Computational Models of Collaborative Learning Interaction, ITS2004, 7th Conference on Intelligent Tutoring Systems, Maceio, Brasil, September 2004.
- FESAKIS, G., PETROU, A., AND DIMITRACOPOULOU, A. 2004. Collaboration Activity Function: An interaction analysis tool for Computer Supported Collaborative Learning activities. Proceedings of the 4th IEEE International Conference on Advanced Learning Technologies (ICALT 2004), 196-200.
- JERMANN, P. 2004. Computer Support for Interaction Regulation in Collaborative Problem-Solving. Doctoral Dissertation, University of Geneva.
- MORENO L., GONZÁLEZ E.J., GONZÁLEZ C.S., AND PIÑERO J.D.. 2009. "Towards a Support for Autonomous Learning Process" . Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living, 10th International Work-Conference on Artificial Neural Networks, IWANN 2009 Workshops, Salamanca, Spain, June 10-12, 2009. Proceedings, Part II. Lecture Notes in Computer Science 5518. 582-585,2009 Editorial: Springer
- MARTÍNEZ-MONÉS, A., GUERRERO, L., AND COLLAZOS, C. 2004. A model and a pattern for the collection of collaborative action in CSCL systems. In J. Mostow, & P. Tedesco (Eds.) Designing Computational Models of Collaborative Learning Interaction, workshop at. ITS 2004 (pp. 31-36). Maceió, Brazil.
- SOLLER, A. 2004. Computational Modeling and Analysis of Knowledge Sharing in Collaborative Distance Learning. User Modeling and User-Adapted Interaction: The Journal of Personalization Research, 14 (4), 351-381.
- GONZÁLEZ, C-S., CABRERA D., BARROSO A. AND LÓPEZ D. 2010. ULLMedia: Sistema PDP de Contenidos Multimedia de la Universidad de La Laguna. In Actas de los Talleres de las Jornadas de Ingeniería del Software y Bases de Datos, Vol. 4, Nº3.
- IMS LOM: Learning Resource Metadata specification v1.1.2. <http://www.imsglobal.org/metadata>
- OGATA, H., MATSUURA, K., AND YANO, Y. 2000. Active Knowledge Awareness Map: Visualizing learners' activities in a Web based CSCL environment. International Workshop on New Technologies in Collaborative Learning, Tokushima, Japan.

Identifying Influence Factors of Students Success by Subgroup Discovery

F. LEMMERICH AND M. IFLAND AND F. PUPPE

University of Würzburg, Germany

The identification of influence factors on students' success rate is of high interest. Especially knowledge about factors, which can be determined before the start of a student's degree program, is important, since most drop-outs occur in the very first semesters. In this paper we show, how the data mining technique of subgroup discovery can be utilized to identify such influence factors on the overall success of students. Additionally, we also discuss several interesting measures for this purpose..

Key Words and Phrases: Subgroup Discovery, Pattern Mining

1. INTRODUCTION AND METHOD

In order to avoid drop-outs in degree programs, early knowledge about influence factors on students' success is a key issue. In this case study, we show how the technique of *subgroup discovery* can be applied to identify important combinations of factors, which are known before students start their degree program, e.g., age, sex, regional origin or previous activities.

Subgroup discovery [Klösgen 1996] aims at finding *descriptions* (conjunctions of attribute-value pairs) of subsets in the population that show an interesting behavior with respect to a certain target concept, e.g., the drop-out rate. The results can be interpreted as a set of association rules, in which the consequent is always the target concept and the antecedent is the description of the respective subgroup. For example, the subgroup with description *age = '20-21' \wedge previous_courses = false* can be reformulated for a search with the target concept drop-out as: *age = '20-21' \wedge previous_courses = false \Rightarrow drop_out = true*.

In a subgroup discovery task the top rules with respect to a predefined measure of interestingness, see [Geng 2006], are returned. The most important quality measures q can be formulated as: $q_a(sg) = n^a(p-p_0)$, where n denotes the size of the subgroup sg , that is, the number of individuals, for which the description of sg applies. p is the share of the target concept in the subgroup and p_0 the share of the target in the total population. The parameter a weights the sizes of subgroups with respect to the deviation in the target share. For example, for $a=1$ the resulting quality function $q_1(sg) = n(p-p_0)$ equals the *Weighted Relative Accuracy*, for $a=0$ we get the *Added Value* $q_0(sg) = (p-p_0)$, which has been applied to evaluate association rules in the educational domain [Merceron 2008]. In addition, we will use adapted *Relative Measures*, cf. [Grosskreutz 2010], which replace the target share in the total population p_0 in the above formula by the maximum target share in any generalized subgroup, i.e., subgroup descriptions, that contain only a subset of describing attribute-value pairs: $r_a(sg) = n^a(p - \max_i(s_i))$. Thus, a specialization is only considered as interesting, if it significantly increases the target share with respect to all its generalizations.

2. CASE STUDY AND DISCUSSION OF RESULTS

For our experiments we first extracted the data of the students, from the multi-relational warehouse into a tabular form, resulting in a total row count of 9400. In additional preprocessing steps attributes were discretized and categorized. The resulting attributes included: drop-out, sex, age at the start of the degree program, school final exam grade,

Authors' addresses:

Universität Würzburg, Fakultät für Mathematik und Informatik
Lehrstuhl für Künstliche Intelligenz und Angewandte Informatik (Informatik VI)
Am Hubland, D-97074 Würzburg

nationality, a flag for previous courses, which were passed before its start, a flag for previous degree programs at this university, and the categorized place of final school exams. The overall drop-out rate in the population was 26.3%. Important single factors, that increase the likelihood of drop-outs are for example $age > 30$, $foreigner=true$ (both 40% drop-outs.) or $age < 20$ (39% drop-outs).

To find interesting combinations of influence factors, we performed subgroup discovery using the interesting measures q_0 , $q_{0.5}$, q_1 and their relative counterparts r_0 , $r_{0.5}$, and r_1 . As an example, the top resulting subgroups for descriptions with two influence factors are presented below with their basic statistics and rankings according to the different interesting measures. Statistics for the respective generalizations are given below each subgroup. For example, the last three rows indicate, that there were 4394 students not originating in the universities state. 28% of these were drop-outs. The drop-out rate for the 4668 male students was equally at 28%. However, the 2066 students, for which both these influence factors apply, had a drop-out rate of 32%. This subgroup was ranked at 14 for the interesting measure q_0 , but was ranked first for the interesting measure r_1 .

SG	Description	size	target share	q_0	$q_{0.5}$	q_1	r_0	$r_{0.5}$	r_1
S_1	grade < 1.5 \wedge age = [26;30]	23	61%	1	-	-	1	1	-
	grade < 1.5	716	35%						
	age = [26;30]	619	21%						
S_2	Prev._course = false \wedge grade = ‘?’	861	44%	4	1	1	14	7	8
	Prev._course = false	9027	27%						
	grade = ‘?’	986	42%						
S_3	Origin_in_state = false \wedge sex = male	2066	32%	-	14	5	13	2	1
	Origin_in_state = false	4394	28%						
	sex = male	4668	28%						

The results show significant differences depending on the used interesting measure. As expected, the measures with a higher parameter a , i.e., q_1 and r_1 prefer larger subgroups, while those with a lower parameter a prefer smaller subgroups with an higher deviation of the target share. The deviation of the target share for combinations of influence factors that result from the absolute measures can sometimes be almost completely be explained by one factor alone. For example, the high deviation of the target share in subgroup S_2 can be almost completely explained by its generalization $grade = ‘?’$. Therefore, we consider this combination of influence factors as less interesting. Such subgroups are ranked significantly lower by utilizing relative measures.

For the result presentation to the head of degree programs we chose subgroups resulting from the interesting measure $r_{0.5}$, as it provided a nice balance between large subgroups and high deviation of the target share. The project received positive feedback and continues to regularly report results each semester.

REFERENCES

- GENG, L., HAMILTON, H.J. 2006. Interestingness measures for data mining: A survey. *ACM Cmp. Surv.* 38, 3, 9.
- GROSSKREUTZ, H., BOLEY, M., AND KRAUSE-TRAUDES, M. 2010. Subgroup Discovery for Election Analysis: A Case Study in Descriptive Data Mining. In: *Discovery Science*, B. Pfahringer, G. Holmes, and A. Hoffmann, Eds. Lecture Notes in Computer Science Series, vol. 6332. Springer, 57–71.
- KLÖSSGEN, W. 1996. A Multipattern and Multistrategy Discovery Assistant. In *Advances in Knowledge Discovery and Data Mining*, U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy, Eds. AAAI Press, 249–271.
- MERCERON, A. AND YACEF, K.. 2008. Interestingness Measures for Association Rules in Educational Data. In: *EDM 2008: 1st International Conference on Educational Data Mining, Proceedings*. 57–66.
- ROMERO, C., GONZALEZ, P., VENTURA, S., DELJESUS, M., AND HERRERA, F. 2009. Evolutionary algorithms for subgroup discovery in e-learning: A practical application using Moodle data. *Expert Systems with Applications* 36, 2, 1632–1644.

Analyzing University Data for Determining Student Profiles and Predicting Performance

D. KABAKCHIEVA

Sofia University “St. Kl. Ohridski”, Bulgaria

K. STEFANOVA, V. KISIMOV

University of National and World Economy, Sofia, Bulgaria

Keywords: Educational Data Mining, Student Profiling, Predicting Student Performance, Classification

1. INTRODUCTION AND RELATED RESEARCH

The open access to education within the enlarged Europe, and even outside it, is introducing hard competition among universities and requires them to introduce advanced methods for data analysis, in order to identify their own uniqueness and to select the most appropriate students to reach better performance.

The implementation of data mining is considered a powerful instrument for acquiring new knowledge from existing data to support decision making. Literature reviews of the Educational Data Mining (EDM) research field are provided by Romero and Ventura (Romero et al. 2007), covering the research efforts between 1995 and 2005, and Baker and Yacef (Baker et al. 2009), for the period after 2005. Recent research papers in the EDM field are focused on understanding student types and targeted marketing (Ma et al 2000, Luan 2002, Luan 2004, Antons 2006), using predictive modeling for maximizing student retention (Noel-Levitz 2008, DeLong 2007, Yu et al 2010), developing enrollment prediction models based on admission data (Nandeshwar 2009), predicting student performance and drop-out (Kotsiantis et al. 2004, Vandamme et al. 2007, Cortez and Silva 2008, Dekker et al. 2009, Kovačić 2010, Ramaswami et al. 2010).

This paper presents a data mining research project that is started at a Bulgarian university, with the main goal to reveal the high potential of data mining applications for university management. The specific objective of the research work is to find out interesting patterns in the available data that could contribute to predicting student performance at the university based on their personal and pre-university characteristics. This is a first attempt of applying data mining in the Bulgarian educational sector.

2. THE RESEARCH METHODOLOGY

The data mining project is based on the CRISP-DM (Cross-Industry Standard Process for Data Mining) research approach. The open source software tool WEKA is used for the project implementation. During the ***Business Understanding Phase*** the specific University management needs are identified. In the ***Data Understanding Phase*** the types of data collected from the university applicants during the enrollment campaigns, and stored in electronic format, are studied. During the ***Data Preprocessing Phase***, student data from the two University databases is extracted and organized in a new data mart.

The research sample includes data about 10330 students, described by 20 parameters (gender, birth year and place, living place and country; type, profile, place and total score from previous education, university admittance year, exam and achieved score, current semester, total university score. The provided data is subjected to many transformations – removing parameters that are considered useless (e.g. fields with one value only), replacing fields containing free text with nominal variable (with a number of distinct values), transforming numeric to nominal variables, etc. The data is also being studied for missing values (very few and not important), and obvious mistakes (corrected).

The data mining task is to predict the student university performance based on the student personal and pre-university characteristics. The target variable is the “student class”. It is constructed as a categorical variable, based on the numeric values of the “student total university score” attribute, and has five distinct values - “excellent” (5.50-6.00), “very good” (4.50-5.49), “good” (3.50-4.49), “average” (3.00-3.49) and “bad” (below 3.00). The dataset contains 10330 instances (539 classified as excellent, 4336 as very good, 4543 as good, 347 as average, and 564 as bad), each described with 14 attributes (1 output and 13 input variables), nominal and numeric.

During the ***Modeling Phase***, several different classification algorithms are selected and applied. Popular WEKA classifiers (with their default settings unless specified otherwise) are used, including a common decision tree algorithm C4.5 (J48), two Bayesian classifiers (NaiveBayes and BayesNet), a Nearest Neighbour (kNN) algorithm (IBk) and two rule learners (OneR and JRip).

3. THE ACHIEVED RESULTS

The WEKA Explorer application is used at this stage. Each classifier is applied for two testing options - cross validation (using 10 folds) and percentage split (2/3 of the dataset used for training and 1/3 – for testing). The results for the overall accuracy of the applied classifiers, including True Positive Rate and Precision (the average values for the 10-fold cross validation and split options), are presented in Table I. The results for the classifiers’ performance on the five classes are presented on Fig.1.

Table I. Results for the accuracy of the applied classifiers

	J48	NaiveBayes	BayesNet	k-NN, k=100	k-NN, k=250	OneR	JRip
TP Rate	0,663	0,586	0,591	0,613	0,593	0,546	0,632
Precision	0,640	0,594	0,597	0,574	0,563	0,480	0,611

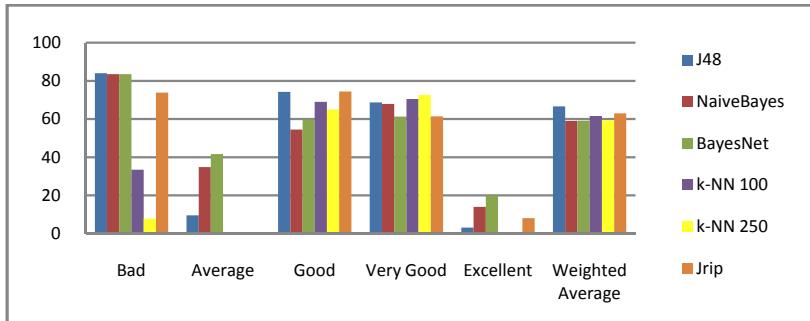


Fig.1. Classification Algorithms Performance Comparison

The achieved results reveal that the decision tree classifier (J48) performs best, followed by the rule learner (JRip) and the k-NN classifier. The Bayes classifiers are less accurate than the others. However, all tested classifiers are performing with an overall accuracy below 70% which means that the error rate is high and the predictions are not very reliable. The predictions are worst for the Excellent class, and bad for the Average class, the k-NN classifier being absolutely unable to predict them. The highest accuracy is achieved for the Bad class (except for the k-NN). The predictions for the Good and Very Good classes are more precise than for the other classes, and all classifiers perform with accuracies around 60-75%.

The selected classifiers perform with similar overall accuracies on the dataset, but they differ with respect to the five classes. All algorithms do not successfully predict the Average and Excellent classes which might be explained with the uneven construction of the target variable. Further research efforts will be directed at achieving higher accuracy of the classifiers’ prediction by additional transformations of the dataset, reconstruction of the target variable, tuning of the classification algorithms’ parameters, etc.

The EDM Vis Tool

Matthew W. Johnson, Michael John Eagle, Leena Joseph and Dr. Tiffany Barnes

University of North Carolina at Charlotte

We introduce EDM Vis, an information visualization software tool for exploring, navigating and understanding student data logs. EDM Vis loads student logs, entire classes, hundreds of students, at a time, and displays the student behavior of those students as they solved problems using a software tutor. The visualization uses a tree structure to provide an overview of class performance, and interface elements to allow easy navigation and exploration of student behavior.

1. INTRODUCTION

“Advancing personalized learning” has been declared a Grand Challenge by the National Academy of Engineers. With increasing use of the web for instructional materials and learning management systems, the amount of data available to help address this challenge is growing rapidly. However, these large datasets on learning can be unwieldy and deciding just how to use them for making learning more effective is also a challenge. The PSLC DataShop, a repository for educational data, has collected logs from over 42,000 students from different tutors with a wide range of topics, from algebra to Chinese [Koedinger et al. 2010]. The DataShop was also used for the 2010 Knowledge Discovery and Data Mining Cup challenge, illustrating that mainstream data mining conferences are realizing the growing need to understand educational data sets.

Anscombe in 1973 identified some key problems in understanding data through statistical measures alone, demonstrating that the same mean and standard deviations can be used to summarize a “quartet” of distinctly different data sets. “A computer should make both calculations and graphs. Both sorts of output should be studied; each will contribute to understanding.”[Anscombe 1973]. Visualization can help avoid misinterpretation of data. In 2002, Ben Shneiderman wrote an influential article, *Inventing Discovery Tools: Combining Information Visualization with Data Mining*, highlighting that both communities, InfoVis and Data Mining, should “integrate data mining and information visualization to invent discovery tools.”[Shneiderman 2001]. Stuart K. Card defines the purpose of visualization to “amplify cognition” about data [Card et al. 1999], in our case, amplify an educator’s cognition about student behavior.

Being able to see and use step-by-step data logs is an important advantage Intelligent tutoring systems can have over traditional homework methods, but this advantage will not be fully leveraged until Visualization systems can be developed to make the data consumable by the masses. In this paper, we report on our attempt to combine InfoVis and data mining to invent a “discovery tool”: EDM Vis. We describe some of the design decisions we made when developing EDM Vis and highlight some of the key considerations we made when developing a visualization tool for log data from students performing procedural problem solving.

2. THE EDM VIS TOOL

EDM Vis is a software visualization tool built to provide educators and researchers a graphical representation of tutor log data, to help improve understanding of student problem-solving and to provide insight about how to improve teaching and learning. EDM Vis is intended to be domain-independent and useful interactive visualization for procedural problem domains.

We designed our visualization tool for visualizing procedural domains, domains requiring a series of steps in order to solve a problem so we use a hierarchical graph representation. In the graph, each node represents a state and each edge an action that takes a user from one state to the next. In this way, we can allow users to visualize an entire set of student attempts at once providing an effective overview of the solution domain as created by the students working in the tutor. An additional inherent requirement is that the tutor's data can be expressed in the form of states, typically the values of the variables in the tutor at discrete times. In general, this is reasonable because most logging systems in tutors log their data in this way.

In addition to the required data: student-ID, start-state, action, end-state, we also support loading in additional data which is stored on the student. Future versions will also support storing extra data on states and actions. This functionality allows users to, for example, load test-scores for the students and compare their actions to other students based on their performance on, for example a pre-test.

3. CONCLUSION AND FUTURE WORK

We presented EDM Vis a software visualization tool designed for exploring and interacting with software-tutor log data. Our tool allows users to get a better understanding, over looking strictly at tutor logs, of how students interacted with a software tutor by visualizing the tutor's interaction logs.

A short survey we provided to our users gave us helpful feedback for areas to improve on in future interactions of the tool. EDM Vis is our first steps to providing a domain independent visualization tool for understanding student behavior in software tutors, and our initial results seem promising for the future development of this tool.

Allowing the user to annotate and augment their data in the visualization seems useful, particularly for highlighting an important aspect of the data a user wishes to share. In addition a method of allowing comparison between problems could prove interesting, for both looking at a single student over multiple problems as well as the same problem from multiple tutors.

It is also clear that for each graph and sub-graph we should provide some general statistics to supply the user with concrete numbers, essentially more details on demand. In addition, we must fix some issues with our use of color, making it easier to know which nodes are errors when selected. Another advanced functionality would be to allow selections to be unions, intersections or excludes of sub-selections, as well as a way to check if student-1 performed action-X did, they also perform action-Y. These types of selection and filtering and their extensions seem to be a good direction to pursue for the next iteration of EDM Vis. Lastly, the advice we gathered from the qualitative portion of the survey will certainly be taken into account and implemented in all possible cases.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. IIS 0845997. Thank you to the EDM conference for supporting LaTex.

REFERENCES

- ANScombe, F. J. 1973. Graphs in statistical analysis. *The American Statistician* 27, 1, pp. 17–21.
- CARD, S. K., MACKINLAY, J. D., AND SHNEIDERMAN, B. 1999. Readings in information visualization. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, Chapter Using vision to think, 579–581.
- KOEDINGER, K., BAKER, R., CUNNINGHAM, K., SKOGSHOLM, A., LEBER, B., AND STAMPER, J. 2010. A data repository for the edm community: The pslc datashop.
- SHNEIDERMAN, B. 2001. Inventing discovery tools: Combining information visualization with data mining. In *Discovery Science*, K. Jantke and A. Shinohara, Eds. Lecture Notes in Computer Science Series, vol. 2226. Springer Berlin / Heidelberg, 17–28.

Towards Modeling Forgetting and Relearning in ITS: Preliminary Analysis of ARRS Data

Y. WANG

Worcester Polytechnic Institute, USA

AND

N.T. HEFFERNAN

Worcester Polytechnic Institute, USA

Researchers in the intelligent tutoring system field have been using the well known Knowledge Tracing model (Corbett and Anderson 1995) to modeling student learning for decades. A lot of variations of the standard Knowledge Tracing model have been developed to improve performance, such as works of Pardos and Heffernan (2010) and works of Baker, Corbett etc. (2010). These models all make the assumption that there is no forgetting during student learning. Yudelson and Medvedeva(2008) used the coupled HMM topology M3 to modeling forgetting aspects in their medical ITS, but didn't give a clear comparison of the model with forgetting and without it. In this paper, we applied the Knowledge Tracing model to analyze data from the Automatic Reassessment and Relearning System. We found that it statistical reliably over predicting student performance after a seven days duration. Corbett and Bhatnagar (1997) reported similar results in their experiments of predicting test performance. We then extended the knowledge tracing framework to model forgetting and relearning but failed to get better results in predicting student performance.

Key Words and Phrases: Intelligent Tutoring Systems, Knowledge Tracing, Bayesian Networks, Data Mining, Prediction

1. INTRODUCTION

The Automatic Reassessment and Relearning System (ARRS) is an extension of the "Mastery Learning" Problem sets in the ASSISTment system. ASSISTments system is a freely available web-based tutoring system for 4th through 10th grade mathematics. Mastery Learning is a strategy that requires students to continually work on a problem set until they have achieved a criterion (typically three consecutive correct answers). The idea of ARRS is if a student masters a problem set, the ARRS system will automatically reassess students a week later, a month later, and then finally two months after that. If students fail the re-assessment, they will be given an opportunity to relearn the topic.

We used the data from two ARRS experiment classes ran in September and November of 2010, with in total 136 students and 53449 data instances. To simplify the analysis, we focused on the first reassessment and relearning phase, that means only data from the original assignment and the one week later reassessment and relearning phase is considered.

2. METHODOLOGY AND RESULTS

First we analyzed the performance of the standard Knowledge Tracing model on the Automatic Reassessment and Relearning System data, which focus on enhance student

Authors' addresses: Y. Wang, Department of Computer Science, Worcester Polytechnic Institute, MA, USA. E-mail: yutaowang@wpi.edu; N.T. Heffernan, Department of Computer Science, Worcester Polytechnic Institute, MA, USA; E-mail: nth@wpi.edu

long term learning instead of short term knowledge boost. The result shows that lacking of consideration of forgetting causes a significantly over predicting on students' first opportunity on a new phase of learning. This proves the necessity of modeling forgetting in student long term learning.

We tried simple extensions of the standard Knowledge Tracing trying to model forgetting and relearning on new reassessment and relearning phase. The design of the Knowledge Tracing with new phase forgetting model is shown in Figure 1.

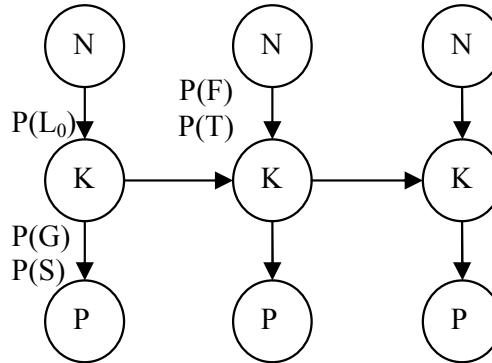


Fig. 1. The Knowledge Tracing with new phase forgetting model design. The node N indicates if it is a new reassessment and relearning phase, the node K indicates student current knowledge level, the node P indicates the student performance of current question. The parameter $P(L_0)$ is the probability of initial knowledge, $P(G)$ is the probability of guess, $P(S)$ is the probability of slip, $P(T)$ is the probability of learning, and $P(F)$ is the probability of forgetting when facing a new learning phase.

We did 10 fold cross validation and used the RMSE (Root Mean Squared Error) as a measurement of the predicting accuracy. The new models gave no better result than the standard Knowledge Tracing. Simulation experiment showed a difficulty of the “forgetting” parameter in these new models to converge into its real value.

ACKNOWLEDGEMENTS

This research was made possible by the U.S. Department of Education, Institute of Education Science (IES) grants #R305K03140 and #R305A070440, the Office of Naval Research grant # N00014-03-1-0221, NSF CAREER award to Neil Heffernan, and the Spencer Foundation. All the opinions, findings, and conclusions expressed in this article are those of the authors, and do not reflect the views of any of the funders.

REFERENCES

- Baker, R.S.J.d., Corbett, A.T., Gowda, S.M., Wagner, A.Z., MacLaren, B.M., Kauffman, L.R., Mitchell, A.P., Giguere, S.: Contextual Slip and Prediction of Student Performance After Use of an Intelligent Tutor. Proceedings of the 18th Annual Conference on User Modeling, Adaptation, and Personalization, 52-63 (2010)
- Corbett, A., Anderson, J.: Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. User Modeling and User-Adapted Interaction 4:253-278 (1995)
- Corbett, A.T. and Bhatnagar, A.: Student modeling in the ACT programming tutor: Adjusting a procedural learning model with declarative knowledge. Proceedings of the COURSES AND LECTURES- INTERNATIONAL CENTRE FOR MECHANICAL SCIENCES, 243-254 (1997)
- Pardos, Z.A., Heffernan, N.T.: Modeling Individualization in a Bayesian Networks Implementation of Knowledge Tracing. In Proceedings of the 18th International Conference on User Modeling, Adaptation and Personalization. pp. 225-266 (2010)
- Pardos, Z.A., Heffernan, N.T.: Navigating the parameter space of Bayesian Knowledge Tracing models: Visualization of the convergence of the Expectation Maximization algorithm. In Proceedings of the 3rd International Conference on EDM (2010)
- Yudelson, M.V. and Medvedeva, O.P. and Crowley, R.S.: A multifactor approach to student model evaluation. Proceedings of the User Modeling and User-Adapted Interaction, 349-382 (2008)

Quality Control and Data Mining Techniques Applied to Monitoring Scaled Scores

A. A. VON DAVIER

Educational Testing Service, Princeton, NJ, USA

For testing programs that provide a large number of administrations each year, the challenge of maintaining comparability of test scores is influenced by the potential rapid accumulation of errors and by the lack of time between administrations to apply the usual techniques for detecting and addressing scale drift. Traditional quality control techniques have been developed for tests with only a small number of administrations per year, and therefore, while very valuable and necessary, they are not sufficient for catching changes in a complex and rapid flow of scores. Model-based techniques that can be updated at each administration could be used to flag any unusual patterns. The basis for the paper is recent research conducted at Educational Testing Service. I will describe an application of traditional quality control charts, such as Shewhart and CUSUM charts on testing data, time series models, change point models, and hidden Markov models to the means of scaled scores to detect abrupt changes. Some preliminary data mining approaches and results also will be discussed. This type of data analysis of scaled scores is relatively new and any application of the aforementioned tools is subject to the typical pitfalls: Are the appropriate variables included? Are the identified patterns meaningful? Can time series models or hidden Markov models be generalized to data from other tests?

Key Words and Phrases: data mining, quality control, scale drift, scaled scores, time series, Shewhart charts, CUSUM charts, change-point models, hidden Markov models

1. INTRODUCTION

In this paper I provide an overview of recent research conducted at Educational Testing Service (ETS) to enhance the data analysis, monitoring, classification, and prediction in evaluating equating results. The perspective I take here is that quality control and data mining tools from manufacturing, biology, and text analysis can be successfully applied to scaled scores and other relevant variables of an assessment. The quality control techniques may help with detecting trends, while the data mining tools may help with identifying (useful) patterns in the data that accompany the scaled scores.

In recent years at ETS, researchers considered monitoring the following variables: means and variances of the scaled and raw scores, means and variances of item parameters after they were placed on a common item response theory (IRT) scale, IRT linking parameters over time (the estimated slope and intercept of the linear relationship between the item/person parameters from the old and new administrations or from the item bank and the new administration), correlations among different sections of the tests, automatic and human scoring data, background variables, and so on. Some of these variables have been investigated by the team responsible for the quality of scores, but in the recent years, this investigation became more focused on patterns over a long chain of administrations. We attempted to address these inquiries by using Shewhart control charts to visually inspect the data over time, time series models to model the relationship of test difficulty and test scores means over time, harmonic regression to remove seasonality, cumulative sum (CUSUM) charts, change-point models and hidden Markov models to detect sudden changes, weighted mixed models and analysis of variance to detect patterns in the data.

2. THE PROCESS OF QUALITY CONTROL IN ASSESSMENT

A brief exposition of the quality control process of the assessment data is as follows: After a testing administration and after the customary data analysis is conducted, the next step is to inspect Shewhart control charts for individual or average of the means of scaled scores over time. One visually inspects the control charts and identifies outliers. CUSUM charts should be inspected next.

The next step is applying time-series techniques to assess the level of autocorrelation and the degree of seasonality in the data as in Lee and Haberman (2011). Or one might model the series of individual raw-to-scale conversions over many administrations using a regression model with autoregressive moving-average (ARMA) errors (see Li, Li, & von Davier, 2011).

Then, one may consider applying a change-point model or a hidden Markov model to detect a point in time when the test results might contain a significant change (see Lee & von Davier, 2011). The main tasks of change-point detection are first to decide whether there has been a change, and if so, to estimate the time at which it occurred.

One might be interested in mining the data further by identifying patterns of test scores per subgroups of test takers. Luo, Lee, and von Davier (2011) investigated a multivariate weighted mixed model where the means of scaled scores are predicted by several background variables and the Test Administration variable, which is defined by specific sample compositions at each administration.

3. CONCLUSIONS

This paper presents a new perspective on quality control in assessments that is appropriate for the new generation of tests that have a continuous or almost continuous administration mode and that are delivered on the computer (and therefore, allow for the collection of additional information, such as time responses). These types of assessments include linear tests but also computer adaptive tests, multi-stage adaptive tests, and linear on-the-fly tests. Moreover, the tools described here can be applied to other assessment variables of interest. These tools can support the validity of the test overtime through timely identifying security breaches, administration errors, or demographic changes. As with all new applications, the approaches described here require more in-depth analyses to refine the approaches for matching the type of data from educational assessments. The theoretical and practical implications of the issues discussed in this paper are crucial for all standardized assessments with nontraditional equating designs and features.

REFERENCES

- LEE, Y.-H., & HABERMAN, S. (2011). *Application of harmonic regression to monitor scale stability*. Manuscript in preparation.
- LEE, Y.-H., & VON DAVIER, A. A. (2011, April). *Monitoring scale scores over time via quality control tools and time series techniques*. Paper presented at the annual meetings of the American Educational Research Association (AERA) and the National Council on Measurement in Education (NCME), New Orleans, LA.
- LI, D., LI, S., & VON DAVIER, A. A. (2011). Applying time-series analysis to detect scale drift. In A. A. von Davier (Ed.), *Statistical models for test equating, scaling, and linking* (pp. 327–346). New York, NY: Springer-Verlag.
- LUO, L., LEE Y.-H., & VON DAVIER, A. A. (2011). *Pattern detection for scaled score means of subgroups across multiple test administrations*. Paper presented at the annual meetings of the American Educational Research Association (AERA) and the National Council on Measurement in Education (NCME), New Orleans, LA

eLAT: An Exploratory Learning Analytics Tool for Reflection and Iterative Improvement of Technology Enhanced Learning

A.L. DYCKHOFF, D. ZIELKE, M.A. CHATTI, U. SCHROEDER
RWTH Aachen University, Germany

Educators are in need for powerful Learning Analytics tools in order to improve the effectiveness of their courses and to enhance their students' performance. In order to help educators to self-reflect on their technology-enhanced teaching and learning scenarios and to help them identify opportunities for interventions and improvements, it is important to provide comprehensible indicators for determining quality and efficiency. In this paper, we present an overview on the goals, requirements, and design of eLAT, a Learning Analytics toolkit which enables teachers to explore and correlate content usage, user properties, user behavior, as well as assessment results based on graphical indicators. The primary aim of eLAT is to process large data sets in real time with regard to individual research interests of teachers and data privacy issues.

Key Words and Phrases: Learning Analytics, Improving educational software, Evaluating teaching interventions, Improving teacher support

1. INTRODUCTION

The exploratory Learning Analytics Tool (eLAT) serves teachers to explore and correlate content usage, user properties, user behavior, as well as assessment results. Based on individually selected graphical indicators it supports reflection on and improvement of online teaching methods based on personal interests and observations. Therefore, eLAT has to provide a clear and usable interface while, at the same time, being powerful and flexible enough for individual information exploration purposes (Dyckhoff, 2011).

2. ELAT: EXPLORATORY LEARNING ANALYTICS TOOL

The requirements analysis of eLAT led to the following software design goals:

- *Usability*: provide an understandable user interface and appropriate methods for data visualization.
- *Interoperability*: ensure compatibility for any kind of LMS by allowing for integration of different data sources.
- *Extensibility*: allow the incremental extension of analytics functionality after the system has been deployed without rewriting code.
- *Reusability*: use a building-block approach to make sure that more complex functions can be implemented by re-using simpler ones.
- *Real-time operation*: make sure that the toolkit can return answers within seconds to allow an exploratory user experience.
- *Data Privacy*: preserve confidential user information and protect the identities of the users at all times.

eLAT is supposed to indicate certain facts about the usage and properties of a learning environment and visualize them appropriately. Therefore, we have introduced the concept of *indicators*, which can be described as specific calculators with corresponding visualizations. Indicators are arranged according to *analytics contexts* which correspond to the subject of a specific question a teacher might have.

One possible analytics context is “Content Activity”. It contains indicators, such as “Unique student activity for areas”, displaying an interactive visualization for content usage of different course areas during a particular period of time, while providing an overlay with the event of that course. Other analytics contexts are, e.g., „Assessment“, „Collaboration“, or „User Activity“. For every indicator any number of parameters may be added to narrow down a question according to certain date ranges, content areas or user properties. The set of available indicators for each analytics context is dynamically calculated dependent on the currently selected learning environment. This is necessary, as not all indicators can be applied to all kinds of course types in a meaningful way.

A typical use case of eLAT would consist of a teacher with a specific question in mind. With the selection of a context, a list of available indicators will be displayed. If required, the indicator will generate user interfaces to let the teacher supply parameters. After validating the configuration, a website will generate an indicator report along with some environment variables, store it in report tables, and send an evaluation request for that report to the evaluation service queue. After the evaluation has been completed, the user will be notified and the default visualizer for this indicator will retrieve the report and generate a result view.

A crucial part of eLAT’s system architecture handles the extensibility and reusability of the existing code base, so that the scope of operations can be increased with the need for new questions that naturally arise after one question has been answered. A single indicator implementation makes use of smaller parts in the form of *expressions* that are performance-optimized database queries to retrieve specific result sets, which can be useful for other indicators as well. The same practice is applied to the dynamic user interface generation for indicator parameters and the visualizers which operate on standardized datasets and are therefore generic to the data inside the report. This leads to a small effort for implementing new indicators. To keep the eLAT implementation independent of a particular LMS, we have developed a neutral data model which supports all major data types as well as an extension model to fit in special types.

During winter term 2010/2011, we selected four courses of RWTH Aachen University that differed in course size, learning technologies and teaching styles. We logged the students’ activities, interactions and assessment data for a period of three months. The collected data was pseudonymised to preserve the privacy of students. Due to this procedure, it was possible to get immediate feedback on early prototype stages.

3. CONCLUSION AND OUTLOOK

The main goal of eLAT is the improvement of teacher support with graphical analytics. Currently, eLAT has been primarily developed with the intention to support teachers in their ongoing reflection, evaluation and improvement activities. eLAT has been successfully tested with data collected from four courses. In the future, we plan to test eLAT with more courses from different disciplines and enhance eLAT in ways that students can use it as well. Future work will also include the evaluation of eLAT from a usability point of view and its enhancement with a recommendation component.

ACKNOWLEDGEMENTS

This project was partly funded by the Excellence Initiative of the Federal and State Governments.

REFERENCES

- DYCKHOFF, A.L. 2011. Implications for Learning Analytics Tools: A Meta-Analysis of Applied Research Questions. *International Journal of Computer Information Systems and Industrial Management Applications*, Volume 3 (2011), pp. 594-601.

Predicting graduate-level performance from undergraduate achievements

J. ZIMMERMANN, K.H. BRODERSEN, J.-P. PELLET, E. AUGUST, and J.M. BUHMANN, ETH Zurich

One of the principal concerns in graduate admissions is how future performance in graduate studies may be predicted from a candidate's undergraduate achievements. In this study, we examined the statistical relationship between B.Sc. and M.Sc. achievements using a dataset that is not subject to an admission-induced selection bias. Our analysis yielded three insights. First, we were able to explain 55% of the variance in M.Sc. performance using a small set of highly discriminative B.Sc. achievements alone. Second, we found the final B.Sc. year to be most informative for future M.Sc. performance. Third, when a failed exam was repeated, the crucial piece of information was the grade achieved in the final attempt. We envisage that results such as the ones described in this study may increasingly improve the design of future admission procedures.

1. INTRODUCTION

Throughout continental Europe, the last decade has seen the gradual adoption of a three-tier education system, consisting of Bachelor's, Master's, and Doctoral programmes. Due to its modularity, the system places high demands on the admission process. Several previous studies have examined the utility of undergraduate GPA (UGPA) scores in predicting graduate-level performance [Lane et al. 2003; Owens 2007]. However, because the UGPA is frequently used as an admission criterion in its own right, most studies to date are based on data with an inherent selection bias and may have underestimated the predictive power of undergraduate performance [Dawes 1975].

In this study, we analysed a dataset that exhibits no selection bias. Specifically, we acquired data from Computer Science undergraduates at ETH Zurich, all of whom were subsequently admitted to the M.Sc. programme, regardless of their undergraduate achievements. We investigated (i) what proportion of the variance of graduate performance could be explained by B.Sc. grades; (ii) whether achievements in the competitive first year or achievements in the final year of the B.Sc. programme proved most predictive; and (iii) the informativeness of first attempts versus final attempts when failed exams had been repeated.

2. METHODS

Data. Data were collected from the B.Sc. and the consecutive M.Sc. programme in Computer Science at ETH Zurich over a seven-year time period. Most B.Sc. courses were mandatory, while the M.Sc. programme granted more freedom of choice. A data matrix was constructed on the basis of 176 students, 125 predictor variables, and one target variable (the GPA of the M.Sc. programme achievements, GGPA). Predictor variables included: gender, age at enrolment, rate of progress, single course achievements (first and final examination attempts, measured on the Swiss 6-point grading scale), several GPA's (precision: two decimal places), and study duration.

Methodology. A random-forest algorithm was used to estimate decision trees for regression on random partitions of the data. Predictions were evaluated using an out-of-bag scheme. We used the canonical variable-importance measure of random forests for feature selection, and we used the pseudo- R^2 statistic for model selection.

Author's address: J. Zimmermann, Department of Computer Science, ETH Zurich, Universitaetstrasse 6, 8092 Zurich, Switzerland. E-mail: judith.zimmermann@inf.ethz.ch

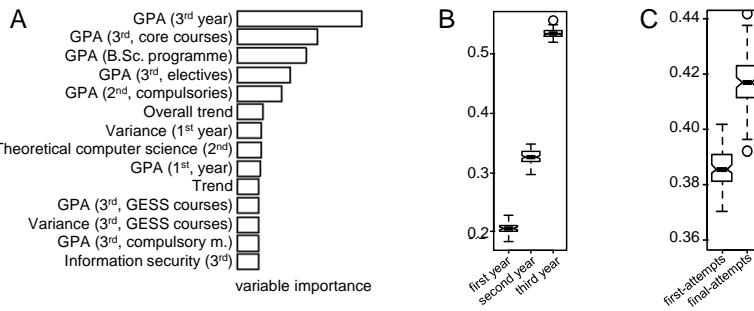


Fig. 1. (a) Degree of importance (x -axis) of individual undergraduate variables in predicting graduate-level performance. (b) Box-plots of 100 pseudo- R^2 estimates for the first, the second, and the third B.Sc. study year. (c) Box-plots of 100 Pseudo- R^2 estimates of single course achievements related to either first attempts or final attempts of exams.

3. RESULTS

Prediction performance and underlying predictors. Regarding the question of overall predictability, a small set of highly discriminative predictor variables explained 55% of the GGPA variance (Figure 1a).

Informativeness of undergraduate years. Concerning the relative importance of different study years, the third undergraduate year was most informative for future performance (Figure 1a,b).

Repeated exams. Regarding failed and repeated exams, models based on grades from final attempts yielded a significantly higher prediction accuracy than models based on grades from first attempts (Figure 1c).

4. DISCUSSION

Our analysis yielded three insights. First, we showed that it is feasible to explain as much as 55% of the variation in graduate performance purely on the basis of undergraduate achievements. This result outperforms previous attempts in the literature, and it highlights the significance of undergraduate achievements as criteria for M.Sc. admission decisions.

Second, we found third-year achievements to be more predictive for future M.Sc. performance than first-year grades. This is an important result, given that one might intuitively overestimate the predictive power of the highly competitive first-year courses.

Third, when exams were failed and repeated, our results indicate that final-attempt grades are more informative than first-attempt grades. Members of admission committees might feel tempted to ask for more information on failed exams, but our study suggests that results commonly reported in academic transcripts may be exhaustive. This observation also indicates that success in subsequent studies may not critically rely on the speed with which students have mastered their material. Rather, the key factor appears to be the amount of knowledge they have acquired at the time of completing an undergraduate degree.

An open question is to what extent the statistical approach adopted here can be extended to predict performance across universities and across countries. We will explore this question in a future study.

REFERENCES

- DAWES, R. 1975. Graduate Admission Variables and Future Success. *Science* 187, 4178, 721–723.
- LANE, J., LANDE, A., AND COCKERTON, T. 2003. Prediction of Postgraduate Performance from Self-Efficacy, Class of Degree and Cognitive Ability Test Scores. *Journal of Hospitality, Leisure, Sport and Tourism Education* 2, 1, 113–118.
- OWENS, M. K. 2007. Executive Education: Predicting Student Success in 22 Executive MBA Programs. *GMAC® Report Series*.

Mining Assessment and Teaching Evaluation Data of Regular and Advanced Stream Students

IRENA KOPRINSKA
University of Sydney, Australia

This paper investigates the effect of the stream (regular or advanced) on the student evaluation of teaching and the course marks. It presents a case study in a third year Computer Science course at an Australian university. The results show that there were no significant differences between the two groups in their perception of teaching and learning. However, the two groups significantly differed in their assessment results in all assessment components and also in the most important predictors of the final mark.

1. INTRODUCTION

Some Australian universities offer the same Units of Study (UoS) in two versions: regular and advanced. In the advanced stream the material is more demanding and the students have higher previous academic results. The aim of this study is to investigate how the differences between the two streams (previous academic performance and UoS demands) affect the student evaluation of teaching and the UoS assessment results.

Our study was conducted in a third year Computer Science course. The two streams had shared lectures (for the common material), separate labs/tutorials (for the more challenging material for the advanced stream) and common and different assessment components (more challenging and open ended for the advanced stream). The number of all enrolled students was 48, 18 advanced and 30 regular.

2. EFFECT OF THE STREAM ON STUDENT EVALUATION OF TEACHING

At the end of the semester students completed a survey similar to [USE] which measures their perception of teaching on a five-point Likert scale. Student evaluation of teaching has been an active area of research [Richardson 2005]. There is convincing evidence that it is reliable, valid and stable over time, and also relatively independent of the class size and expected grades. We extend previous research by studying if there are differences in the perception of the advanced and regular students doing the same UoS. Some possible differences are in the workload, clarity of explanation and adequacy of previous UoSSs. E.g., it may be difficult to find the right balance between the common and different content and assessment, resulting in much higher workload for the advanced stream. Also, the teaching methods and resources may not be equally efficient for both groups.

Using the Mann-Whitney test we found that overall there were no statistically significant differences between the two streams in their perception of teaching, with both groups being positive. It was particularly encouraging to see that the workload imposed by the more challenging assignments for the advanced stream was not perceived as too high. The findings can be used by teaching management to justify the existence of two streams within a single UoS as an alternative to two independent streams.

3. EFFECT OF THE STREAM ON ASSESSMENT RESULTS

We investigate two questions: 1) if there are significant differences between the marks of the two groups in the individual assessment components and the final mark and 2) which assessment components are the best predictors of the final mark for each groups. Previous work on mining assessment data includes McNamara [2004] whose goal was competency mapping and Pechenizkiy et al. [2008] who investigated if test questions and feedback on them matched the individual student needs.

We used the student marks on all three assessment components: homeworks, assignments and exam. The weighting of these components was 9%, 26% and 65%. The

weekly homeworks involved small problem solving tasks. The two assignments were project-based and involved writing a computer program to solve a real-world problem and a report to discuss the results. The end-of-semester exam involved problem solving tasks organised in 9 questions. The difference in assessment between the two streams was 25% in total: 0% in the homeworks, 10% in the assignments and 15% in the exam.

As the weighting of the assessment components was different, all marks were converted into percentages of the individual components. Table I shows the mean and standard deviations for the assessment components. The advanced group performed better than the regular group in all three assessment components and the differences were statistically significant. While the differences in the homework and exam marks were consistent (19%), there was a huge difference in the assignments mark (33%). A closer examination showed that the assignments engaged better the advanced students who implemented additional features in their programs and analysed the results much better. Another interesting observation is that there was significant difference between the two groups in the common part of the exam but not in the different part. This can be interpreted as a well chosen different part, not disadvantaging any of the two groups.

Table I. Assessment data (** - Mann-Whitney stat. significant differences, $p<0.05$)

Mark	Homeworks **	Assign- ments **	Exam common **	Exam different	Exam total **	Final **
Reg.	58.9±27.8	58.2±24.9	65.6±16.2	68.9±28.3	66.4±17.4	63.6±18.3
Adv.	77.3±26.6	91.2±11.4	81.5±13.0	71.9±22.4	79.3±13.7	82.2±10.4

To examine the predictive power of the individual assessment components on the final mark we used correlation analysis. We found that for the regular stream all assessment components correlated highly with the final mark ($r=0.806-0.951$). For the advanced stream the exam marks correlated highly ($r=0.702-0.931$), the homework mark correlated moderately ($r=0.604$) and the assignment mark correlated weakly ($r=0.229$). All correlations except the last one were statistically significant at $p<0.05$.

To investigate the predictive power of the individual exam questions on the final mark we used multiple regression. The exam is the most comprehensive assessment component; it tests all topics while the other assessment components are focused on selected topics. Initially all 9 questions were entered as independent variables and at each step the most non-significant independent variable was removed ($p>0.05$) until only significant predictors were left. The results showed that for the regular stream all exam questions except two relatively easy questions were good predictors of the final mark. For the advanced stream the most important predictors were two non-trivial questions.

In summary, we showed how correlation and regression analysis can be used to gain a better understanding of the assessment results. The advanced and regular streams differed in the assessment results and the predictors of the final marks. As expected the advanced students achieved higher marks in all assessment components and this difference was highest in the project-based assignments. The results can be used to improve future offerings of the course and provide timely feedback to students during the semester, e.g. by predicting the final mark based on a progress mark during the semester.

REFERENCES

- USE. <http://www.itl.usyd.edu.au/use/questionnaires.htm>
 McNAMARA, R.A. (2004). Evaluating assessment with competency mapping. In *Proc. ACE'2004*.
 RICHARDSON, J.T.E. (2005). Instruments for obtaining student feedback: a review of the literature. *Assessment and Evaluation in Higher Education* 30(4):387–415.
 PECHENIZKIY, M., CALDES, T., VASILYEVA, E., AND DE BRA, P. (2008). Mining the student assessment data: lessons drawn from a small scale case study. In *Proc. EDM'2008*, 187-191.

Investigating Usage of Resources in LMS with Specific Association Rules

A. MERCERON

Beuth University of Applied Sciences, Berlin, Germany.

Learning Management Systems (LMS) are widely used to support face to face as well as on-line teaching. In several courses we have observed that students make decreasing use of the resources uploaded for them in the system as the semester progresses. We want to investigate whether a core group of students emerges that keep using the resources or whether, on the contrary, students are eclectic in their choice, consulting resources randomly though they use them less as the semester progresses. This paper introduces specific association rules to investigate this pattern. High confidence of these rules, confirmed by a good rating of other interestingness measures, means that a core group emerges. Support of these rules does not play any role; in fact these rules could be rare.

In the Beuth University of Applied Sciences in Berlin teaching is supported by the use of a Learning Management Systems (LMS), in our case Moodle. Some teachers are interested in knowing how their students learn with the help of all the resources they put on-line for them. A first step in answering this question is to calculate some simple statistics showing the use of resources by students. While displaying these statistics we have come across an interesting pattern like the one depicted in Figure 1 concerning non-compulsory self-tests, here `ex1` to `ex7`, that teachers make available during the semester. This pattern indicates that the number of students attempting these self-tests decreases during the semester.

We are interested in investigating this pattern to uncover the strategy adopted by students: Are they gradually giving up completely, which means that the students who

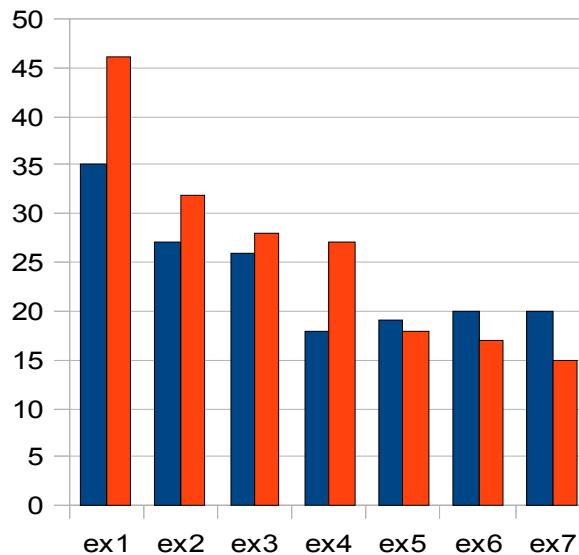


Fig. 1. Number of students attempting self-tests in 2 courses. Left: foundation of computer science , right: Java

Author's addresses: A. Merceron, Department of Computer Science and Media, Beuth University of Applied Sciences, Luxemburgerstraße 10, D-13353 Berlin, Germany. E-mail: merceron@beuth-hochschule.de. This work is partially supported by the European Social Fund for the Berlin state.

attempt the self-test of week i is roughly a sub-group of the students who attempted the self-test of week $i-1$? Or are they eclectic in their choice, which means students attempt randomly some self-tests during the semester though they attempt them less as the semester progresses? We suggest investigating this pattern with the help of specific association rules. First we propose to extract local rules of the form $X_{i+1} \rightarrow X_i$ which mean the following: If students attempt self-test $i+1$, then they also attempted the preceding one. High confidence of these rules denotes that the students who attempt self-test $i+1$ form almost a subgroup of those who attempted the preceding self-test, since we are in the context in which more students have attempted self-test i than self-test $i+1$. When confidence of these local rules is high we propose to go ahead extracting global rules of the form $X_{i+1} \rightarrow X_i, X_{i-1}, \dots, X_1$, which mean the following: If students attempt self-test $i+1$, then they also attempted all the preceding ones. High confidence of these rules denotes the emergence of a core group that keeps attempting the self-tests. Support of these specific rules is not relevant. It could be low leading to the discovery of rare association rules [KohRountree].

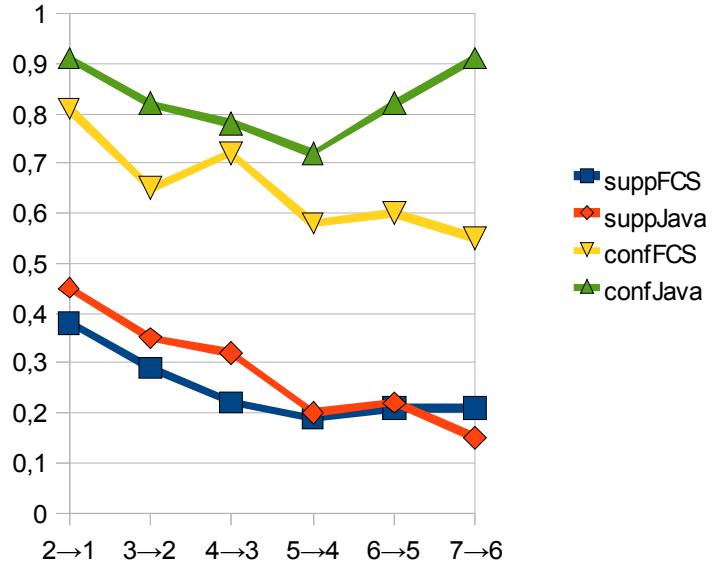


Fig. 2. Support and confidence of local rules.

Figure 2 shows support and confidence of local association rules. $2 \rightarrow 1$ means if students attempt the second self-test, they attempt the first one. One notices at a glance that support for both courses keeps decreasing as expected. Confidence is high for the Java course suggesting the emergence of a core group, which is less true for the foundation course. Local rules of the Java course have been rated as interesting by three other measures of interestingness. Confidence of the global rules for the foundation course is low while it is mainly above 0.6 for the Java course. Students have adopted different strategies in these two courses. Local and global association rules can be used for any kind of resources when usage over time indicates a drop similar to the one of Figure 1.

REFERENCES

KOH, Y. S., AND ROUNTREE, N. (EDS.).2010. RARE ASSOCIATION RULE MINING AND KNOWLEDGE DISCOVERY: TECHNOLOGIES FOR INFREQUENT AND CRITICAL EVENT DETECTION. IGI GLOBAL, ISBN9781605667546

Towards Parameter-Free Data Mining: Mining Educational Data with *yacaree*

MARTA E. ZORRILLA and DIEGO GARCÍA-SAIZ and JOSE L. BALCÁZAR, University of Cantabria

The need of parameter-free alternatives for data mining algorithms is widely recognized, and is particularly acute in the web-based educational field, where instructors involved in the teaching process are interested in improving their virtual courses and adapting these to the learners' behavior: most often, these instructors are not expected to know about data mining technologies. We report on a quantitative comparison of several algorithms for association rules on educational datasets, including in the comparison both well-established implementations and a recent parameter-free association miner; our purpose is to clarify whether this newer approach is actually useful for the educational data mining field. Our results indicate that it has indeed a high potential, and allows us to identify some important aspects that must be improved.

1. INTRODUCTION

In the educational arena, Data Mining techniques are acquiring a major importance since the appearance of the e-learning environments. These systems log all the activity carried out by students and instructors, and this raw data, adequately processed, may offer useful knowledge about the learning process for instructors.

But data mining techniques are out of the reach of most teachers, e.g., for humanities or law studies. Thus, if we want to help users of all disciplines, we need to work out data mining tools that do not require much tuning or technical understanding from the user. In particular, this is relevant for the case of association rules: all the available algorithms up to recent work depend on one or more parameters (confidence, support,etc) whose value is to be set by the user, and whose semantics may not be easy to grasp. Likewise, the number of rules which obtain as output is often large, and most of them are redundant and non-interesting for decision making [García et al. 2007].

There is, thus, a clear need to design and implement parameter-free data mining algorithms addressed to “non-experts”, and they must stand reasonably well a comparison with other “expert”-oriented algorithms. To the best of our knowledge, *yacaree* [Balcázar 2011] is the first parameter-free association miner implemented. Here, we compare this system with other three well-known association rule miners: the Apriori [Agrawal and Srikant 1994] and Predictive Apriori [Scheffer 2001] tools from Weka, and Borgelt’s Apriori implementation [Borgelt 2003].

2. MAIN RESULTS

Yacaree is a parameter-free algorithm which mines frequent closed itemsets and constructs association rules from them; a main property is that it reduces the number of rules shown to the user by means of a parameter, called confidence boost, which eliminates redundant rules. Essentially, a rule is considered redundant with respect to another if it has larger antecedent or smaller consequent, and simultaneously the ratio of their confidences falls below a given threshold. This algorithm is available at <http://sourceforge.net/projects/yacaree/>.

Thus, this algorithm tends to generate rules with small antecedents and larger consequents, unlike the original association rules and their implementation by Borgelt. In fact, whereas a rule like $X \rightarrow AB$ implies

This work is partially financed by the Spanish Government grants TIN2007–66523 and TIN2008–05924.

Author's addresses: M. E. Zorrilla, D. García-Saiz and J. L. Balcázar, Mathematics, Statistics and Computation Dept., Univ. of Cantabria.

Table I. Number of rules obtained on our datasets with the four algorithms

	Number of rules at support 1% and confidence 66%			
	Weka Apriori	Predictive Apriori	Borgelt Apriori	yacaree
Dataset1	2272	1730	617	24
Dataset2	7523	over 10000	3751	214
Dataset3	4249	over 10000	1876	88

both rules $X \rightarrow A$ and $X \rightarrow B$, the converse is not true, and, therefore, *yacaree* chooses to keep larger consequents whenever possible as they furnish the most informative configuration. This program keeps a fixed confidence threshold, defaulting at $2/3$, and it reports only rules that belong to a certain “basis” of irredundant rules corresponding to the threshold. It has very generous start values for support and confidence boost thresholds; but a key property is that the algorithm “tunes” on itself these two thresholds along the computation, by monitoring lift, memory consumption, and other parameters.

Our three educational datasets come from logs of a virtual course, and relate sessions and materials employed by the students; they have, respectively, 407 transactions on 22 items (Dataset1), 2486 transactions on 27 items (Dataset2), and 2346 transactions on 26 items. As *yacaree* self-tunes the support, we performed a brief preprocessing to tune manually the rest of algorithms and guarantee a fair comparison. We decided to fix at 1% the support threshold for all the computations, and at $2/3$ (or 66%) the confidence threshold. The limit on the number of rules in the Weka tools was set very high (at 10000 rules), and left unbounded in Borgelt’s Apriori and *yacaree*. We show the number of rules obtained for each case in Table I.

The first consideration that we can highlight is that *yacaree* provides a reasonable size of the output. In general these rules contain good knowledge without overwhelming the user. Furthermore, these rules are, intuitively, reasonably irredundant (“they say different things”). Instead, both Apriori implementations in Weka and the one by Borgelt lead to more voluminous and redundant output. Predictive Apriori tends to choose first rules of a support rather lower than the user would like to, tends to create overwhelming output sizes, and leaves room for quite a degree of redundancy. Its running time tends to be unacceptably high, and the “expected predictive accuracy” parameter is less interpretative than support and confidence for the end-user. On the other hand, all of these well-established algorithms do return rules of 100% confidence, something that *yacaree* does not. We hope to add this feature in the near future, as this experiment clearly marked this as the issue that needs remedy most urgently.

In the opinion of the instructor involved in the virtual course analyzed (prof. Rafael Menéndez), the results of *yacaree* are superior in comparison with the rest of the algorithms used in our case study, in terms of subjective usefulness for the teacher. In summary, *yacaree* seems particularly well-suited to educational datasets which seem to require a low support threshold, but do include items of rather high support, as this combination seriously hinders the ability of traditional association miners to offer interesting output.

REFERENCES

- AGRAWAL, R. AND SRIKANT, R. 1994. Fast algorithms for mining association rules in large databases. In *VLDB*, J. B. Bocca, M. Jarke, and C. Zaniolo, Eds. Morgan Kaufmann, 487–499.
- BALCÁZAR, J. L. 2011. Parameter-free association rule mining with *yacaree*. In *EGC*, A. Khenchaf and P. Poncelet, Eds. Revue des Nouvelles Technologies de l’Information Series, vol. RNTI-E-20. Hermann-Éditions, 251–254.
- BORGELT, C. 2003. Efficient implementations of apriori and eclat. In *FIMI*, B. Goethals and M. J. Zaki, Eds. CEUR Workshop Proceedings Series, vol. 90. CEUR-WS.org.
- GARCÍA, E., ROMERO, C., VENTURA, S., AND CALDERS, T. 2007. Drawbacks and solutions of applying association rule mining in learning management systems. In *Proceedings of the International Workshop on Applying Data Mining in e-Learning*. 13–22.
- SCHEFFER, T. 2001. Finding association rules that trade support optimally against confidence. In *In: 5th European Conference on Principles of Data Mining and Knowledge Discovery*. 424–435.

Factors Impacting Novice Code Comprehension in a Tutor for Introductory Computer Science

LEIGH ANN SUDOL-DeLYSER, Carnegie Mellon University
JONATHAN STEINHART, Tutor Technologies

Code comprehension activities have been strongly correlated with measures of student performance in computer science classes. This poster presents a model predicting student success based on features collected during the tutoring session. Model selection indicated that experimental condition combined with within tutor cumulative percent correct yielded the best model.

1. INTRODUCTION TO DATA

To test the effect of questions regarding the low-level details of a program compared to questions regarding the more abstract nature of the algorithms, a tutoring environment was constructed. Prior work has shown a strong correlation between students' perception of the global goals of a program in a code comprehension task and their ability to write code to solve problems[1; 2; 3]. Using these tutoring data, we model the students' learning progression through the activities and compare student performance on common tutoring questions. The models were evaluated for predictive factors in student performance. Performance on common questions was dependent upon the factors of the design as well as the students' performance and how many questions they had completed. These results can help give us insight into the type of practice students need to better comprehend code and points to types of feedback to explore in future work.

In the fall of 2010, introductory computer science students from Carnegie Mellon University (N=236) participated in a study to determine the effects of interacting with a code comprehension exercise. Students engaged with the tutor as a part of a homework assignment in their introductory computer science course. Three different instructors teach the course, and students were drawn from all classes.

Students were randomly assigned to conditions pairing Abstraction vs Tracing with Context vs. No-Context. In the contextualized conditions, the examples that students saw included a one-sentence description of the context, and variables names were changed to match the context.

As a part of the tutor, students interacted with 30 code comprehension questions. The 30 questions referred to about 10 different code segments, implementing various algorithms (e.g., sum, search, sort, rotate). Each code segment had a series of three questions associated with it. The first question was a tracing question and was seen by all students, regardless of condition, and is referred to as the common question. The second and third question varied by condition. In the Tracing condition, students were asked to trace the code with two other data sets. In the Abstract condition, students were asked questions about the global goals of the algorithm, or the role of its components. Code examples were presented to all students in the same order, and the first 6 were selected to be easier than the last 4.

2. DATA MODEL CONSTRUCTION

The following factors are included in the models used in this paper. *Tracing*: (categorical), tracing condition? (yes/no) *Context*: (categorical), context present? (yes/no) *Problem Number*: (ordinal), problem number, to account for order effect *Problem ID*: (categorical), to account for specific problem characteristics *Cumulative Percent Correct*: (numerical), the percent correct so far within tutor *Student*: a random intercept to account for random variation in ability between students

We used logistic regression to estimate students' mean probability of success, given characteristics that would be known to a "live" Intelligent Tutoring System. Logistic regression models the log odds of success of the i th student, π_i , as some unknown linear combination of the student's attributes.

Logistic regression models the log odds of success of the i th student, π_i is

$$\log\left(\frac{\pi_i}{1 - \pi_i}\right) = X_i\beta \Leftrightarrow \pi_i = \frac{1}{1 + e^{-X_i\beta}}$$

We performed model selection using an initial model fit (using `glm`) to the entire dataset and containing all of the aforementioned variables and all second-order interactions. We then performed stepwise selection (using `step`) to find a model which minimizes Akaike's Information Criterion (AIC). The resulting model kept all of the main effects except problem number, interactions between problem id and each of the other main effects, and the interaction between context and cumulative percent correct.

3. RESULTS

To validate our model, we performed five-fold cross-validation. We fit the aforementioned model to each training fold, which resulted in a predicted success probability for each training case. To convert predicted probabilities (continuous) into predicted success/failure (dichotomous), a threshold value was chosen to maximize Cohen's κ on the training data. The error rate and κ were then measured for each test fold. Then, the resulting five cross-validation error rates and κ values (i.e. one for each fold), were averaged for a mean CV error rate of 0.1294 and approximate κ of 0.4849. Our somewhat crude model seems to do a reasonable job of predicting success, although there is much room for improvement.

In addition to predicting success/failure, our fitted logistic regression model also offers some explanatory insight into the ways in which code base, context, tracing/abstraction, and student ability act and interact. Some observations from the fit include:

- Cumulative percent correct (a rough measure of student ability) generally increased the probability of success, and this boost was greater in the context conditions.
- Code base 2 was easier for those in the tracing conditions but – interestingly – harder overall for those with higher cumulative percent correct ($p=0.0528$)!
- Except for code base 2, though, tracing generally hurt performance for everyone, and particularly in code base 10.
- Context made code base 8 significantly easier, but otherwise it generally had no effect on success.

4. ACKNOWLEDGEMENTS

We would like to thank the PSLC, especially Brett Leber for help in constructing and deploying the tutor. This work was partially funded by the IES, US Department of Education, through Grant R305B040063 to Carnegie Mellon University. The opinions expressed are those of the authors and do not represent the views of the Institute or the US Department of Education.

REFERENCES

- R. J. Barker and E. A. Unger. A predictor for success in an introductory programming class based upon abstract reasoning development. pages 154–158, 1983.
- M. Lopez, J. Whalley, P. Robbins, and R. Lister. Relationships between reading, tracing and writing skills in introductory programming. In *ICER '08: Proceeding of the Fourth international Workshop on Computing Education Research*, pages 101–112, New York, NY, USA, 2008. ACM.
- A. Venables, G. Tan, and R. Lister. A closer look at tracing, explaining and code writing skills in the novice programmer. In *ICER '09: Proceedings of the fifth international workshop on Computing education research workshop*, pages 117–128, New York, NY, USA, 2009. ACM.

Investigating the Transitions between Learning and Non-learning Activities as Students Learn Online

P. S. INVENTADO¹, R. LEGASPI¹, M. SUAREZ² AND M. NUMAO¹

¹The Institute of Scientific and Industrial Research, Osaka University, Japan

²Center for Empathic Human-Computer Interactions, De La Salle University - Manila, Philippines

Many students today utilize the internet to help them accomplish their learning goals. However, when they learn at home they are in total control and it is easy for them to visit websites not related to learning when they lose focus or are not motivated enough to learn. Observing affect further will help us understand the transitions between learning and non-learning activities when students learn online. To achieve this, we collected affect and activity transition data from students learning online at home. D'Mello's likelihood metric was modified to compute the likelihood of transitioning between activities and their corresponding affective states. Results showed that students not only shift to non-learning activities after experiencing negative affective states, but also positive affective states plausibly when learning goals are completed. Also, despite engaging in non-learning activities, students resumed learning and even experienced positive affect which is beneficial to learning.

Key Words and Phrases: online, learning, non-learning activities, affective states, transition likelihood

1. INTRODUCTION

Information seeking is a major part of online learning since it allows students to find resources needed to accomplish their learning goals, using online tools and collaborating with peers [Smith and Caruso 2010]. Students can use these tools not only in school, but also at home while doing homework and projects or studying for quizzes. At home, however, students have complete control over their learning process and do not engage in learning related activities alone but also shift into non-learning activities.

In [Luo et al. 2011], students tasked to seek information on the internet experienced transitions in affective states which played a role in how they proceeded next. For example, students who experienced excitement when feeling they were on the right track proceeded with the task assigned to them, but showed confusion and frustration when they were not able to find what they expected and later gave up. Since learning online consists mostly of information seeking, we expect that it is influenced by affect. Also, since students have complete control in this domain, affect may not only influence learning but also their shifts to non-learning related activities. When experiencing frustration, they can easily browse game or media websites to de-stress and alleviate negative affect.

Observing affect will help us understand the transitions between learning and non-learning activities when students learn online. We collected data from students learning at home where there were no observers, specific goals or time frames given to ensure that they felt total control over their own learning. However, they were asked to install a web browser plugin which showed a popup window every time they viewed a webpage asking them to annotate the type of the activity (i.e., learning related or non-learning related) and the most pronounced affective state they were in (i.e., delighted, engaged, neutral, bored, confused, frustrated). Information about their browsing behavior and their annotations were sent via internet into a web server to collect the data.

Author's address: P. Inventado, R. Legaspi and M. Numao, 8-1 Mihogaoka, Ibaraki, Osaka, 567-0047, Japan; email: {inventado,roberto}@ai.sanken.osaka-u.ac.jp, numao@sanken.osaka-u.ac.jp; Merlin Suarez, Gokongwei Building, De La Salle University, 2401 Taft Avenue, Manila, Philippines; email: merlin.suarez@delasalle.ph

2. RESULTS AND ANALYSIS

The transitions between learning and non-learning activities with their corresponding affective states were analyzed using a modified version of D'Mello's likelihood metric [D'Mello et al. 2007]. Equation 1 measures the likelihood of transitioning between an activity and a corresponding affective state to another.

$$L_{Act1,AffSt1 \rightarrow Act2,AffSt2} = \frac{Pr(Act2, AffSt2 | Act1, AffSt1) - Pr(Act2, AffSt2)}{(1 - Pr(Act2, AffSt2))} \quad (1)$$

A value above 0 indicates a likely transition with increasing likelihood as it approaches 1. Zero indicates likelihood that is at the chance level. A value below 0 indicates that the transition is less likely than the base rate of the next activity with its corresponding affective state. The modified likelihood metric was used on all student data to compute the likelihood of transitioning between all combinations of activity and affective states. Likely transitions important to the study are shown below.

Table I. : Mean transition likelihoods of all student activities and affective states. (L = learning; NL = nonlearning; D = delighted; N = neutral; E = engaged; C = confused; B = bored; F = frustrated)

(a) L → L						(b) L → NL			(c) NL → L			
	LD	LN	LE	LC	LB	LD	NLD	NLB	NLF	LD	LB	LF
LD	0.01					0.03	0.02			0.04		0.01
LN	0.05						0.03		0.04			
LE	0.03	0.002	0.01	0.04	0.01		0.02	0.01				
LC	0.02			0.03								
LB					0.12							
LF						0.07						

Results show that as students transition between activities, they are likely to experience changes in affective states. Both positive (i.e., delighted, engaged) and negative affect (i.e. bored, confused, frustrated) were likely to persist and may either motivate students or cause them to give up as was described by [Luo et al. 2011]. When students experience confusion, they are likely to transition into non-learning activities which may indicate the point when they fail to accomplish their learning goal and give up. Interestingly however, even if students experienced positive affect while learning, it was likely for them to transition into non-learning activities. This may mean that either they engaged in non-learning activities after completing their learning goal, or non-learning activities served as distractions while they were learning. Non-learning seems to have helped students since it was likely for them to transition back to learning and they were even brought to a positive state (NLE → LD) which is advantageous to learning. However, it was also likely that non-learning activities may have caused the student to lose interest in learning (NLC → LB), or dislike learning (NLE → LF) because they were more engaged in non-learning. Although affect can provide possible explanations for activity transitions, further research is needed to verify them. Since affect does not fully explain the transitions, it is also worthwhile to explore other factors that may influence these transitions.

REFERENCES

- D'MELLO, S., TAYLOR, R. S., AND GRAESSER, A. 2007. Monitoring Affective Trajectories During Complex Learning. In *Proceedings of the 29th Annual Meeting of the Cognitive Science Society*. Cognitive Science Society, Austin, TX, USA.
- LUO, M. M., NAHL, D., AND CHEA, S. 2011. Uncertainty, Affect, and Information Search. In *Proceedings of the Hawaii International Conference on System Sciences*. IEEE Computer Society, Los Alamitos, CA, USA.
- SMITH, S. D. AND CARUSO, J. B. 2010. ECAR Study of Undergraduate Students and Information Technology, 2010 (Research Study, Vol. 6). Tech. rep., Boulder, CO: EDUCAUSE Center for Applied Research.

Learning parameters for a knowledge diagnosis tool in orthopedic surgery

S. LALLÉ

Joseph Fourier University, France

AND

V. LUENGO

Joseph Fourier University, France

We provide and illustrate a methodology for taking into account data for a knowledge diagnosis method in orthopaedical surgery, using Bayesian networks and machine learning techniques. We aim to make the conception of the student model less time-consuming and subjective. A first Bayesian network was built like an expert system, where experts (in didactic and surgery) provide both the structure and the probabilities. However, learning the probability distributions of the variables allows going from an expert network toward a more data-centric one. We compare and analyze here various learning algorithms with regard to experimental data. Then we point out some crucial issues like the lack of data.

Key Words and Phrases: Knowledge diagnosis, machine learning, Bayesian network, surgery

1. THE STUDY

TELEOS¹ (Technology Enhanced Learning Environment for Orthopaedical Surgery) is an Intelligent Tutoring System designed for the percutaneous orthopedic surgery [Vadcard and Luengo 2004]. A student model based on a Bayesian network was built after a long didactical analysis of the domain, as presented in Minh Chieu et al. [2010]. Bayesian networks for student modeling usually are expert system; in TELEOS, surgeons were implied for designing both the structure and the probability tables of the network. However it seems interesting to use a more automatic approach, as presented by Mayo and Mitrovic [2001]. First, the model includes a lot of variables and experts can roughly estimate all the parameters and are prone to error or approximation. Then surgeons sometimes work in a different way and taking into account their various points of view may be hard. In TELEOS, a robotic arm that records continuous data like the strength or the position is also used for the knowledge diagnosis. Surgeons are not used to deal with such data. Thus, our work aims to study some algorithms for learning the parameters of the Bayesian network.

Given data, the parameters can be learned, i.e. computed from a base of observations. Indeed, we can find in the literature several algorithms designed for learning the probabilities of a Bayesian network. However, they present different characteristics and have both strong and weak points – no guarantee of results can be offered anyway. We studied three of them:

- Maximum likelihood (ML) for counting facts in the database
- Maximum A Priori (MAP), based on ML algorithm but taking into account prior knowledge on the domain
- Estimation-Maximisation (EM) that can handle misses in the database

These algorithms are well known in the literature; see for example [Heckerman 1995].

¹ The TELEOS project is supported by the CNRS and by the French research agency (ANR-06-BLAN-0243).

Authors' addresses: S. Lallé, Laboratoire informatique de Grenoble, Université Joseph Fourier, Grenoble, France. E-mail: sebastien.lalle@imag.fr; V. Luengo, Laboratoire informatique de Grenoble, Université Joseph Fourier, Grenoble, France; E-mail : vanda.luengo@imag.fr

We need quality data, in term of quantity, coverage and representativeness, for learning the parameters in a pertinent way. Data was collected at the Grenoble's hospital in France. First, one surgeon and six interns realized a set of six exercises with TELEOS, each of them presenting various characteristics and difficulties. Then, the experimental team and a second surgeon both had to manually perform the knowledge diagnosis, based on the observation of the students. For each action, knowledge may be either brought into play in a valid way, in an invalid way, or not used whereas it should have been. We got at the end a database of only 3000 entries. Indeed collecting data is expensive in our domain, as we need at least one surgeon.

2. RESULTS

Data was used for learning the parameters and validating the network in two different manners. We first performed cross-validation in order to estimate the accuracy of the diagnosis (i.e. we trained the model with a part of the data and validated it on the other part). As the expert did the diagnosis only for almost 1/3 of the data, we partitioned the data in different ways for the cross-validation. In method I we blend expert and non-expert data (Table I), in method II we only used expert data for learning the parameters and non-expert data for the validation (Table II). Then we used a 3-folds method (Table III). Results are shown below.

Table I. Prediction accuracy (method I)

Algorithm	Prediction accuracy
ML	59.3%
MAP	63.4%
EM	59.8%

Table II. Prediction accuracy (method II)

Algorithm	Prediction accuracy
ML	51.3%
MAP	58.5%
EM	51.6%

Table III. Accuracy with 3-folds cross validation

Algorithm	Fold 1	Fold 2	Fold 3
ML	58.33%	60.2%	57.75%
MAP	63.3%	63.7%	66.18%
EM	59.36%	60.3%	59.97%

According to these results, the MAP estimation gives the best accuracy, probably due to a good prior distribution based on our knowledge of the domain. Since there is few misses in the database, EM algorithm gives almost the same results than the Maximum Likelihood. However, the accuracy is not really good. A first explanation may be the lack of data and the difference between expert and non-expert data. On the other hand various data reduces potential bias for the learning.

To conclude, we compared different ways to learn the parameters of a Bayesian network for the knowledge diagnosis that keep the deep didactical analysis of the domain (here the structure). In future works, we want to evaluate this data-centric approach in a more qualitative way, with new experiments at the hospital. We also wish to bring out a methodology that takes into account both expert knowledge and data.

REFERENCES

- HECKERMAN D. 1995. A tutorial on learning with Bayesian networks. In *Innovations in Bayesian Networks*, 33–82.
- MAYO M., and MITROVIC A. 2001. Optimising ITS behaviour with Bayesian networks and decision theory. In *International Journal of Artificial Intelligence in Education*, 12, 124–153.
- MINH CHIEU V., LUENGO V., VADCARD L., and TONETTI J. 2010. Student modeling in complex domains: Exploiting symbiosis between temporal Bayesian networks and fine-grained didactical analysis. In *Journal of Artificial Intelligence in Education*, 20, 269–301.
- VADCARD, L., and LUENGO V. 2004. Embedding knowledge in the design of an orthopaedic surgery learning environment. In *International Conference on Computer Aided Learning in Engineering education*.

Problem Response Theory and its Application for Tutoring

P. JARUŠEK and R. PELÁNEK, Faculty of Informatics, Masaryk University Brno

Problem solving is an important component of education. To make problem solving activities attractive, it is important to confront students with problems of suitable difficulty – neither too easy, nor too difficult. Since students vary in their skills, it is also crucial to make problem recommendations individually adaptive. We present a novel problem response theory, which predicts how much time will a student need to solve a given problem. Our theory is an analogy of the item response theory, but instead of probability of a correct answer we model problem solving time. We introduce a problem solving tutor, which uses the theory to make adaptive predictions and to recommend students problems of suitable difficulty.

1. PROBLEM RESPONSE THEORY

People enjoy the learning process most when facing problems of a challenging difficulty – neither boring, nor frustrating [Csikszentmihalyi 1975]. Our main aim in this paper is to predict a difficulty of problems, more specifically to predict time it will take a person to solve a problem. We aim to do the prediction based on previous data about problem solving activity of this and other persons (as opposed to prediction based on analysis of a problem structure). To this end we propose a “problem response theory”, which models a relation between a problem solving ability and a time to solve a problem. The theory is a variation of the standard item response theory [Baker 2001].

Item response theory is used mainly in testing. Main assumption is that a given test measures one latent ability θ , and models give a relation between this ability θ and the probability P that a test item is correctly answered. This relation is expressed by an item response function. The most common model is a 3 parameter logistic model, which has the following parameters: b is a basic difficulty of an item, a is a discrimination factor, and c is a pseudo-guessing parameter (see Fig. 1).

There are many extensions of the basic model, particularly models which take into account response times [Van Der Linden 2009]. But none of these models is directly applicable to the problem solving setting. Therefore, we propose a problem response theory, which models relation between problem solving ability and time to solve a problem.

Similarly to item response theory, we assume that a problem solving performance depends on one latent problem solving ability θ . We are interested in problem response function $f(\theta)$, which for a given ability θ gives an estimate of a time to solve a problem. More specifically, the function gives a probabilistic density of times (see Fig. 1).

To obtain a specific model we make the following two assumptions, which are grounded on data about human problem solving from our previous experiments [Jarušek and Pelánek 2011; Pelánek 2011]. Firstly, the distribution of solving times $f(\theta)$ for persons with a fixed ability θ is a log-normal distribution. Secondly, the mean and variance of the distribution $f(\theta)$ are exponentially dependent on θ .

Our basic model is a 3 parameter model in which the intuitive meaning of the parameters is the following (we intentionally use notation analogical to item response theory): a is a discrimination factor, b is a basic difficulty of a problem, and c is a randomness factor. The problem response function, i.e., the probability density that a person with ability θ will solve a problem with logarithm of time $\ln t$, is given by a normal

This work is supported by GAČR grant no. P202/10/0334.

Author's address: Botanická 68a, 602 00 Brno, Czech Republic

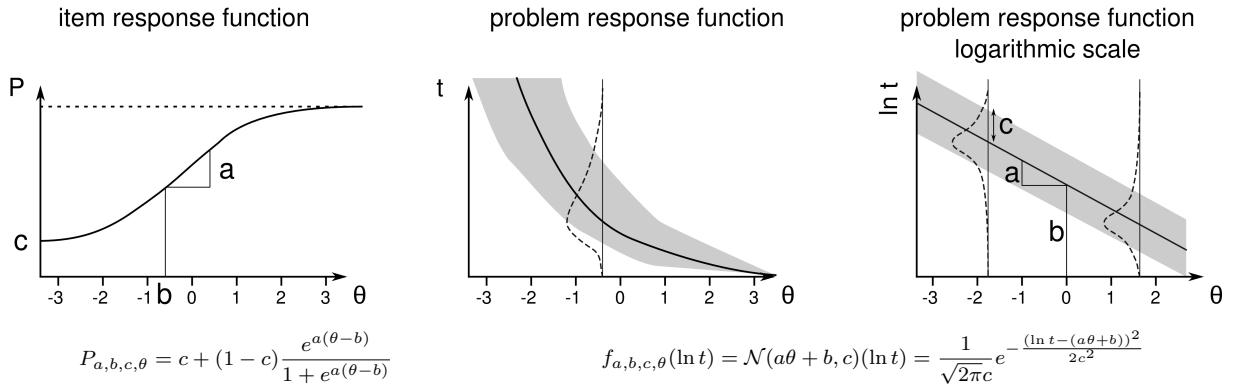


Fig. 1. Intuitive illustration of item response function, general problem response function, and a specific problem response function under our assumptions. Dashed lines illustrate distributions for certain skill θ ; solid lines denotes the median of a time distribution, grey areas depict the area into which most attempts should fall.

distribution with a mean $b + a\theta$ and a variance c^2 . This model and intuition behind its parameters are illustrated in Fig. 1.

2. PROBLEM SOLVING TUTOR

Intelligent tutoring systems [Anderson et al. 1985] are computer programs used to make learning process more adaptive and student oriented. We apply our theory in development of a “Problem solving tutor” – a web portal for practicing problem solving skills, which is available at tutor.fi.muni.cz. The tutor contains large set of problems of different types (math and programming problems, logic puzzles).

Problem parameters a, b, c and user skills θ are estimated using an iterative computation: problem parameters are computed using estimates of user skills; user skills are improved using estimates of problem parameters (both direction are computed by maximum likelihood estimation); and this process continues until requested precision is reached. Based on these estimates the system predicts problem solving times and recommends a suitable problem to solve. The collected data problem solving data are continuously used to further improve parameters estimates and problem recommendations.

Preliminary evaluation shows that predictions based on the problem response theory bring significant improvement over a baseline prediction algorithm (using mean times). Detailed evaluation will be presented in a future paper.

REFERENCES

- ANDERSON, J., BOYLE, C., AND REISER, B. 1985. Intelligent tutoring systems. *Science* 228, 4698, 456–462.
- BAKER, F. 2001. *The basics of item response theory*. University of Wisconsin.
- CSIKSZENTMIHALYI, M. 1975. *Beyond boredom and anxiety*. Jossey-Bass.
- JARUŠEK, P. AND PELÁNEK, R. 2011. What determines difficulty of transport puzzles? In *Proc. of Florida Artificial Intelligence Research Society Conference (FLAIRS 2011)*.
- PELÁNEK, R. 2011. Difficulty rating of sudoku puzzles by a computational model. In *Proc. of Florida Artificial Intelligence Research Society Conference (FLAIRS 2011)*.
- VAN DER LINDEN, W. 2009. Conceptual issues in response-time modeling. *Journal of Educational Measurement* 46, 3, 247–272.
- VANLEHN, K. 2006. The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education* 16, 3, 227–265.

Towards Better Understanding of Transfer in Cognitive Models of Practice

MICHAEL V. YUDELSON, PHILIP I. PAVLIK JR., and KENNETH R. KOEDINGER, Carnegie Mellon University

Achieving transfer – the ability to apply acquired skills in contexts different from those contexts the skills were mastered in – is, arguably, the sine qua non of education. Capturing transfer of knowledge has been addressed by several user modeling and educational data mining approaches (e.g., AFM, PFA, CFA). While similar, these approaches use different underlying structures to model transfer: Q-matrices and T-matrices. In this work, we compare of a more traditional Q-matrix-based method and the relatively new and more complex T-matrix based method. Comparisons suggest that the T-matrix, although demonstrating only marginally better fits, offers a more interpretable and consistent picture of learning transfer.

Transfer is, arguably, the sine qua non of education, in which the primary goal for the learner is to be able to apply new skills in contexts often different from the ones they were mastered in. Because of this, achieving transfer from the math skills trained by computer-aided educational systems to the math abilities used later in a student's life is a litmus test for determining the success of such systems as a whole.

Educational systems access the transfer by tracking students' learning with help of some model (usually math based). This model is also used to make instructional decisions. If the model can be simple or complex, but if it does not provide practice with transfer in mind, it is unlikely that long term-learning will be strong. In psychology, for example, it is well known that what optimizes immediate performance is unlikely to be the practice that optimizes long-term retention or transfer [Schmidt and Bjork 1992]. Due to this paradox of "desirable difficulties" an educational software system needs a model that is clever enough to see not only the effect of practice on repetition, but also see the effect of practice on transfer. This sort of model is unlikely to be easy to configure, since understanding transfer is paramount to understanding education itself, because of how crucial transfer is to a flexible education.

This problem has been addressed by the user modeling and educational data mining communities. While many approaches have been attempted over the years to handle transfer, the Q-matrix method of assigning latent knowledge components (KCs) to particular problems or problem steps has begun to have a large following, even being supported with a suite of logging and analysis tools [Koedinger et al. 2008]. Q-matrix (or question matrix) methods have a long history [Birenbaum et al. 1992; Tatsuoka 1983]. They assign rules to questions so as to determine the aggregate difficulty of items in an instructional situation. The Q-matrix method is notable for the way it specifies these rules abstractly as latent variables that are contained in one or more questions. In contrast, a newer "T-matrix" method assumes learning is less abstract by not describing latent variables, but rather looking at the transfer effects with a question by question matrix where each question causes learning that effects other questions directly [Pavlik Jr. et al. 2011].

In order to better understand transfer, we decided to meticulously compare these two approaches. This contrast is interesting, because the Q-and T-matrix models make clearly different assumptions about transfer.

This work is supported by the U.S. Department of Education (IES-NCSER) #R305B070487 and was also made possible with the assistance and funding of Carnegie Learning Inc., the Pittsburgh Science of Learning Center, DataShop team (NSF-SBE) #0354420 and Ronald Zdrojowski.

Author's addresses: M. V. Yudelson, P.I. Pavlik, and K.R. Koedinger, Human Computer Interaction Institute, School of computer Sciences, Carnegie Mellon University.

In the Q-matrix case, the method specifies transfer by the sharing of latent construct between 2 items. Because of this sharing of the latents, it is seldom the case that transfer is asymmetric in the model. On the other hand, in the T-matrix case, there is no sharing of latents. Instead, the T-matrix - a more complex method - specifies each directional pairwise relationship between a transferred-from item and a transferred-to item individually. Therefore, the T-matrix model is more fit for capturing asymmetry of transfer.

If we trace these assumptions of Q-and T-matrix-based models to the learning science and psychology literature, we can see a problem that inspires our comparison. Specifically, there are notable examples in the literature where asymmetrical transfer occurs strongly (e.g. [Bassok and Holyoak 1989]). Similarly, we find cases where learning is optimal for one condition (e.g. concrete task), but when transfer is analyzed, another condition is more beneficial (e.g. a more abstract task) [Goldstone and Son 2005]). However, there exist other examples of transfer being successfully modeled as latent skills (rf. [Singley and Anderson 1989]). Therefore, the question is, given the prior success of the Q-matrix-based methods and the potential benefits of a relatively new T-matrix method, can we find the evidence that warrants coping with T-matrix complexity for the benefit of its greater flexibility?

In this work we attempt to answer this question by comparing Q-matrix models and T-matrix models to determine which fit the data better and which provide a richer qualitative model of the data. This comparison allows us to establish whether the added complexity of the T-matrix method is justified. While the overall fit is important, we also keep in mind that the goal of educational data mining is not just to produce the optimal model, but also to produce a model that has educational implications. Because of this, we consider the different practical implications of each model to help us establish which one is qualitatively more useful for understanding the educational data.

Our comparison of user modeling methods based on Q-matrix and T-matrix suggest that the latter, although demonstrating only marginally better fits, offers a more interpretable and consistent picture of learning transfer. The results support the claim that in specific cases there is a need for the types of analysis where asymmetric transfer is specifically modeled. Despite the fact that T-matrix-based models are not quantitatively better overall, we found that there were consistent qualitative patterns of asymmetric transfer revealed by the T-matrix model. By comparing the model parameters that governed transfer in each model we were able to see that often Q-matrix-based method were producing bidirectional transfer models by averaging. At the same time the T-matrix-based model was able to show asymmetry clearly.

REFERENCES

- BASSOK, M. AND HOLYOAK, K. 1989. Interdomain transfer between isomorphic topics in algebra and physics. *Journal of Experimental Psychology: Learning, Memory, and Cognition* 15, 153–166.
- BIRENBAUM, M., KELLY, A., AND TATSUOKA, K. K. 1992. *Diagnosing knowledge states in algebra using the rule space model*. Educational Testing Service. NCTM, Princeton, NJ.
- GOLDSTONE, R. AND SON, J. 2005. The transfer of scientific principles using concrete and idealized simulations. *Journal of the Learning Sciences* 14, 69–110.
- KOEDINGER, K., CUNNINGHAM, K., SKOGSHOLM, A., AND LEBER, B. 2008. An open repository and analysis tools for fine-grained, longitudinal learner data. In *Proceedings of the 1st International Conference on Educational Data Mining*, R. Baker and J. Beck, Eds. 157–166.
- PAVLIK JR., P. I., YUDELSON, M. V., AND KOEDINGER, K. R. 2011. Using contextual factors analysis to explain transfer of least common multiple skills. In *Proceedings of the 15th International Conference on Artificial Intelligence in Education (AIED 2011)*, J. Kay, S. Bull, and G. Biswas, Eds. Springer-Verlag, New York, NY (accepted).
- SCHMIDT, R. AND BJORK, R. 1992. New conceptualizations of practice: Common principles in three paradigms suggest new concepts for training. *Psychological Science* 3, 207–217.
- SINGLEY, M. AND ANDERSON, J. 1989. *Transfer in the ACT* theory*. In *The transfer of cognitive skill*. Harvard University Press VIII, Cambridge, MA.
- TATSUOKA, K. 1983. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of Educational Measurement* 20, 4, 345–354.

List of authors

A

Akçapinar, Gökhan	259
Al-Qaraghuli, Ammar	111
Altun, Arif	259
Anaya, Antonio R.	317
Anjewierden, Anjo	341
Antunes, Claudia	329
August, Elias	357

B

Baker, Ryan S.J.D.	101, 179, 189, 199, 315
Balcázar, Jose	363
Barker-Plummer, Dave	51
Barnes, Tiffany	289, 349
Barracosa, Joana	329
Beck, Joseph	81, 307
Blikstein, Paulo	235
Boticario, Jesús G.	317
Brodersen, Kay H.	357
Brunskill, Emma	217, 327
Buhmann, Joachim M.	357

C

Cambranes-Martinez, Edgar	339
Celikovic, Milan	265
Chatti, Mohamed Amine	355
Chi, Min	61, 199
Cobo, Germán	253
Cohen, William	31
Conati, Cristina	159
Corbett, Albert	179
Cosgun, Erdal	259
Cox, Richard	51
Croy, Marvin	289

D

Dale, Robert	51
De Hoog, Robert	341
Desmarais, Michel	41
Desmet, Piet	121, 247
Duan, Weisi	169
Dyckhoff, Anna Lea	355

E

Eagle, Michael	349
Estévez, Román	343

F

Fancsali, Stephen	331
Feng, Mingyu	295

G

García-Solórzano, David	253
García-Saiz, Diego	323, 363
Gijlers, Hannie	341
Gobert, Janice	315
Goguadze, George	301
Gong, Yue	81
González-Brenes, José	149, 169
González, Carina	333, 343
Good, Judith	339
Gordon, Geoff	61
Gowda, Sujith	179, 189, 199

H

He, Qiwei	325
Heffernan, Cristina	295
Heffernan, Neil	101, 129, 139, 189, 295, 351
Hershkovitz, Arnon	315
Hong Tan, Bao	149
Horváth, Tomáš	11

I

Ifland, Marianus	345
Ignatov, Dmitry	223, 229
Inoue, Hitoshi	335
Inventado, Paul Salvador	367
Isotani, Seiji	301
Ivancevic, Vladimir	265

J

Jarušek, Petr	371
Johnson, Matthew	349
Jordan, Pamela	61
Joseph, Leena	349

K	
Kabakchieva, Dorina	347
Kardan, Samad	159
Kay, Judy	111
Kharrufa, Ahmed	111
Kisimov, Valentin	347
Koedinger, Kenneth	31, 61, 71, 91, 199, 373
Kolotova, Elena	229
Koprinska, Irena	359

L	
Lallé, Sébastien	369
Lammers, Thomas	5
Legaspi, Roberto	367
Lemmerich, Florian	345
Li, Nan	31
Lu, Hanyuan	139
Luengo, Vanda	369
Lukovic, Ivan	265
van der Linden, Erik-Jan	5

M	
Maldonado, Roberto Martinez	111
Mamedova, Serafima	223
Marquez-Vera, Carlos	271
Matsuda, Noboru	31
McLaren, Bruce	301
Melenchón, Javier	253
Menéndez Domínguez, V.H.	321
Merceron, Agathe	361
Monzo, Carlos	253
Morán, Jose Antonio	253
Moreno, Lorenzo	343
Mostafavi, Behrooz	289
Mostow, Jack	149, 169, 241, 283, 337
Munna, Mdahaduzzaman	283

N	
Nixon, Tristan	91, 211
Nooraei B., Bahador	101
Numao, Masayuki	367
Nwaigwe, Adaeze	71

P	
Pardos, Zachary	101, 129, 139, 189, 295
Pavlik, Philip I. Jr.	91, 277, 373
Pelánek, Radek	371
Pellet, Jean-Philippe	357
Perez, Rafael Pedraza	319
Popescu, Beatrice	343
Prieto, Manuel	321
Puppe, Frank	345

Q	
Qi, Yingmei	139
Qiu, Yumeng	139

R	
Rabbany Khorasgani, Reihaneh	21
Rai, Dovan	307
Ritter, Steven	91
Romashkin, Nikita	223, 229
Romero, Cristobal	271, 319, 321
Rowe, Jonathan P.	199
Russell, Stuart	327

S	
Saab, Nadira	341
Santamaría, Eugènia	253
Sárközy, Gábor	129
Schmidt-Thieme, Lars	11
Schroeder, Ulrik	355
Shamshurin, Ivan	223
Smyth, Barry	3
Sosnovsky, Sergey	301
Stamper, John	7, 91
Stefanova, Kamelia	347
Steinhart, Jonathan	365
Suarez, Merlin	367
Sudol DeLyser, Leigh Ann	365
Sumiya, Takahiro	335

T	
Tagawa, Takahiro	335
Takaffoli, Mansoureh	21

Thai-Nghe, Nguyen	11
Toledo, Pedro A.	333
Towle, Brendon	211
Trivedi, Shubhendu	129

V

Van Den Noortgate, Wim	121,247
Vanlehn, Kurt	61
Veldkamp, Bernard	325
Ventura, Sebastián	271,319
Von Davier, Alina	353
Vuong, Annalies	211

W

Wang, Yutao	351
Wauters, Kelly	121,247
Westerhof, Gerben	325
Wijffelaars, Martijn	5
Wixon, Michael	315
Worsley, Marcelo	235
Wu, Sue-Mei	277

X

Xu, Yanbo	241,283,337
-----------	-------------

Y

Yacef, Kalina	111
Yamakawa, Osamu	335
Yastake, Koichi	335
Yudelson, Michael	373

Z

Zaïane, Osmar	21
Zapata Gonzalez, Alfredo	321
Zielke, Dennis	355
Zimmermann, Judith	357
Zorrilla Pantaleón, Marta	323,363