Educational Data Mining

Cecily Heiner (Co-chair)
Neil Heffernan (Co-chair)
Tiffany Barnes (Co-chair)

Supplementary Proceedings of the 13th International Conference of Artificial Intelligence in Education.

Marina del Rey, CA. USA. July 2007

Educational Data Mining Workshop

(http://www.educationaldatamining.org/)

Cecily Heiner; University of Utah; cecily@cs.utah.edu (Co-chair)
Neil Heffernan; Worcester Polytechnic Institute; nth@wpi.edu (Co-chair)
Tiffany Barnes; University of North Carolina at Charlotte, tbarnes2@uncc.edu (Co-chair)

Introduction
Educational data mining is the process of converting raw data from educational systems to useful information that can be used to inform design decisions and answer research questions. Data mining encompasses a wide range of research techniques that includes more traditional options such as database queries and simple automatic logging as well as more recent developments in machine learning and language technology.
Educational data mining techniques are now being used in ITS and AIED research worldwide. For example, researchers have used educational data mining to:

- Detect affect and disengagement
- Detect attempts to circumvent learning called "gaming the system"
- Guide student learning efforts
- Develop or refine student models
- Measure the effect of individual interventions
- Improved teaching support
- Predict student performance and behavior

However, these techniques could achieve greater use and bring wider benefits to the ITS and AIED communities. We need to develop standard data formats, so that researchers can more easily share data and conduct meta-analysis across tutoring systems, and we need to determine which data mining techniques are most appropriate for the specific features of educational data, and how these techniques can be used on a wide scale. The workshop will provide a forum to present preliminary but promising results that can advance our knowledge of how to appropriately conduct educational data mining and extend the field in new directions.

Topics
They include, but are not limited to:

- What new discoveries does data mining enable?
- What techniques are especially useful for data mining?
- How can we integrate data mining and existing educational theories?
- How can data mining improve teacher support?
- How can data mining build better student models?
- How can data mining dynamically alter instruction more effectively?
- How can data mining improve systems and interventions evaluations?
- How do these evaluations lead to system and intervention improvements?
- How is data mining fundamentally different from other research methods?

TABLE OF CONTENTS

Evaluating Problem Difficulty Rankings Using Sparse Student Response Data

Ari BADER-NATAL [1], Jordan POLLACK
DEMO Lab, Brandeis University

Abstract. Problem difficulty estimates play important roles in a wide variety of educational systems, including determining the sequence of problems presented to students and the interpretation of the resulting responses. The accuracy of these metrics are therefore important, as they can determine the relevance of an educational experience. For systems that record large quantities of raw data, these observations can be used to test the predictive accuracy of an existing difficulty metric. In this paper, we examine how well one rigorously developed – but potentially outdated – difficulty scale for American-English spelling fits the data collected from seventeen thousand students using our SpellBEE peer-tutoring system. We then attempt to construct alternate metrics that use collected data to achieve a better fit. The domain-independent techniques presented here are applicable when the matrix of available student-response data is sparsely populated or non-randomly sampled. We find that while the original metric fits the data relatively well, the data-driven metrics provide approximately 10% improvement in predictive accuracy. Using these techniques, a difficulty metric can be periodically or continuously re-calibrated to ensure the relevance of the educational experience for the student.

1. Introduction

Estimates of student proficiency and problem difficulty play central roles in Item Response Theory (IRT) [11]. Several current educational systems make use of this theory, including our own BEEweb peer-tutoring activities [2,8,9,13]. IRT-based analysis often focuses on estimating student proficiency in the task domain, but the challenge of estimating problem difficulty should not be overlooked. While student proficiency estimates can inform assessment, problem difficulty estimates can be used to refine instruction: these metrics can affect the selection and ordering of problems posed and can influence the interpretation of the resulting responses [6]. It is therefore important to choose a good difficulty metric initially and to periodically evaluate the accuracy of a chosen metric with respect to available student data. In this paper, we examine how accurately one rigorously developed – but potentially outdated – difficulty scale for the domain of American-English spelling predicts the data collected from students using our SpellBEE system [1]. The defining challenge in providing this assessment lies in the nature of the data. As SpellBEE is a peer-tutoring system, the challenges posed to students are determined by other students, resulting in data that is neither random nor complete. In this

[1]Correspondence to: Ari Bader-Natal, Brandeis University, Computer Science Department – MS 018. Waltham, MA 02454. USA. Tel.: +1 781 736 3366; Fax: +1 781 736 2741; E-mail: ari@cs.brandeis.edu.

work, we rely on a pairwise comparison technique designed to be robust to data with these characteristics. After assessing the relevance of this existing metric (in terms of predictive accuracy), we will examine some related techniques for initially constructing a difficulty metric based on non-random, incomplete samples of observed student data.

2. American-English spelling: A sample task domain

The educational system examined here, SpellBEE, was designed to address the task domain of American-English spelling [1]. SpellBEE is the oldest of a growing suite of web-based reciprocal tutoring systems using the Teacher's Dilemma as a motivational mechanism [2]. For the purposes of this paper, however, the mechanisms for motivation and interaction can be ignored, and the SpellBEE system and the difficulty metric used by it can be specifically re-characterized for an educational data mining audience.

2.1. Relevant characteristics of the SpellBEE system

Students access SpellBEE online at SpellBEE.org from their homes or schools. As of May 2007, over 17,000 students have actively participated. After creating a user account, a student is able to log in, choose a partner, and begin the activity.[2] During the activity, students take turns posing and solving spelling problems. When posing a problem, the student selects from a short list of words randomly drawn from the database of word-challenges. This database is comprised of 3,129 words drawn from Greene's New Iowa Spelling Scale (NISS), which will be discussed in the next section [12].[3] When responding to a problem, the student types in what they believe to be the correct spelling of the challenge word. The accuracy of the response is assessed to be either correct or incorrect. Figure 1 presents a list of the relevant data stored in the SpellBEE server logs.

To date, we have observed over 64,000 unique (case-insensitive) responses to the challenges posed,[4] distributed across over 22,000 completed games consisting of seven questions attempted per student. Student participation, measured in games completed, has not been uniform, however. Of the challenges in the space, most students have only attempted a very small fraction. In fact, when examining the response matrix of every student by every challenge, less than 1% of the matrix data is known. An important characteristic of the SpellBEE data, then, is that the response matrix is remarkably sparse. Given that the students acting as tutors are able to – and systemically motivated to – express their preferences and hunches through the problems that they select, another important characteristic of the SpellBEE data is that the data present in the student-challenge response matrix is also biased. The effects of this bias can be found in the following example: 16% of student attempts to spell the word "file" were correct, while 66% of attempts to spell the word "official" were correct. The average grade level among the first set of students was 3.9, while for the second set it was 6.4. In Section 3.2 we

[2] In the newer BEEweb activities, if no one else is present, a student can practice alone on problems randomly drawn from the database of challenges posed in the past.

[3] In SpellBEE, the word-challenges are presented in the context of a sentence, and so of the words in Greene's list, we only use those found in the seven public-domain books that we parsed for sentences.

[4] Of these, 17,391 were observed more than once. In this paper, we restrict the set of responses that we consider to this subset. See Footnote 7 for the rationale behind this.

Figure 1. The SpellBEE server logs data about each turn taken by each student, as shown in the first list. The data in the first list is sufficient to generate the data included in the second list.

1. `time` : a time-stamp allows responses to be ordered
2. `game` : identifies the game in which this turn occurred
3. `tutor` : identifies the student acting as the tutor in this turn
4. `tutee` : identifies the student acting as the tutee in this turn
5. `challenge` : identifies the challenge posed by the tutor
6. `response` : identifies the response offered by the tutee

1. `difficulty` : the difficulty rating of the challenge posed by the tutor
2. `accuracy` : the accuracy rating of the response offered by the tutee

will present techniques designed to draw more meaningful difficulty information from this type of data.

2.2. Origin, use, and application of the problem difficulty metric

When trying to define a measure of problem difficulty for the well-studied domain of American-English spelling, we were able to benefit from earlier research in the field. Greene's "New Iowa Spelling Scale" provides a rich source of data on word spelling difficulty, drawn from a vast study published in 1954. Greene first developed a methodology for selecting words for his list (5,507 were eventually used.) Approximately 230,000 students from 8,800 classrooms (grades 2 through 8) around the United States participated in the study, totally over 23 million spelling responses [12]. From these, Greene calculated the percentage of correct responses for each word for each grade. This table of success rates is used in SpellBEE to calculate the difficulty of each spelling problem for students, whose grade level is known.

3. Techniques for assessing relative challenge difficulty

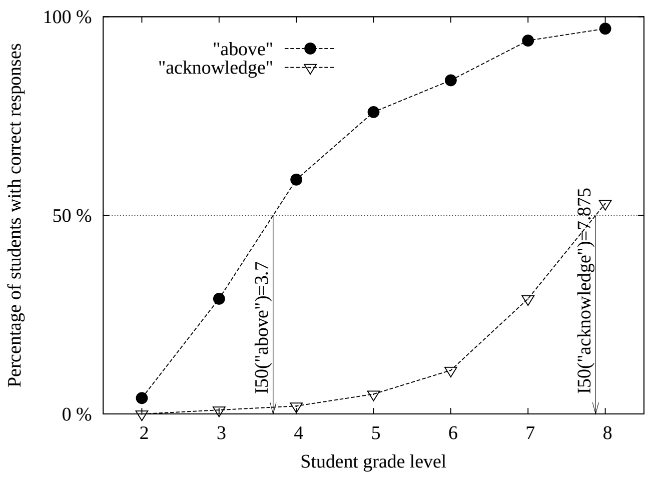
The research questions addressed in this paper focus on the fit of the difficulty model based on the NISS data to the observed SpellBEE student data. Two different techniques are involved in the calculating this fit. The first converts the graded NISS data to a linear scale. The second identifies from the observed student data a difficulty ordering over pairs of problems, in a manner appropriate for a sparse and biased data matrix. Both will be employed to address the research questions in the following sections.

3.1. Linearly ordering challenges using the difficulty metric

Many subsequent studies have explored various aspects of Greene's study and the data that it produced. Cahen, Craun, and Johnson [5] and, later, Wilson and Bock [14] explore the degree to which various combinations of domain-specific predictors could account for Greene's data. Initially starting with 20 predictors, Wilson and Bock work down to a regression model with an adjusted R^2 value of 0.854.[5] Here, we not interested in

[5]The most influential of which being the length of the word.

Figure 2. Difficulty data for two words from the NISS study are plotted, and the I_{50} statistics are calculated.



predicting the NISS results, but instead are interested in assessing the fit (or predictive power) of the 1954 NISS results to observations made of students using SpellBEE over 50 years later. We will drawn upon one statistic used by Wilson and Bock: the one-dimensional flattening of the seven-graded NISS data. This statistic, which they refer to as the "location" of the word, is the (fractional) grade level at which 50% of the NISS students correctly spell word w.[6] We denote this as $I_{50}(w)$. Figure 2 illustrates how the graded difficulty data that is used to derive this statistic for two different words. The value of this statistic is that it provides a single grade-independent difficulty value for a word that can be compared directly to that of other words.

3.2. Identifying pairwise difficulty orderings using observed student data

Given the characteristics of the data collected from the SpellBEE system, identifying the more difficult of a pair of problems based on this data is not trivial. The percentage of correct responses to a challenge, the calculation used to generate the NISS data, is not appropriate here, as the assignment of challenges to students was done in a biased, non-random manner (recall the "file"/"official" example from Section 2.1.) Tutors, in fact, are motivated to base their challenge selection on the response accuracies that they anticipate. A more appropriate measure, rooted in several different literatures, is to assess pairwise problem difficulties on distinctions indirectly indicated by the students. In the statistics literature, McNemar's test provides a statistic based on this concept [10], in the IRT literature, this is used as a data reduction strategy for Rasch model parameter estimation [7], and the Machine Learning literature includes various approaches to learning

[6]Wilson and Bock calculate the 50% threshold based on a logistic model fit to the discrete grade-level data, while we calculate the threshold slightly differently, based on a linear interpolation of the grade-level data.

Table 1. While the I_{50} metric flattens the grade-specific NISS data to a single dimension, the relative difficulty ordering of most word-pairs based on the graded NISS data is the same as when based on the I_{50} scale. In this table, we quantify the amount of agreement between I_{50} and each set of grade-specific NISS data using Spearman's rank correlation coefficient. The strong correlations observed suggest that the unidimensional scale sufficiently captures the relative difficulty information from the original NISS dataset. (The number of words, N, varies by grade, as the NISS study did not show several of the harder words to the younger students.)

Grade	N	Spearman's ρ
2	2218	0.751
3	3059	0.933
4	3126	0.977
5	3129	0.974
6	3129	0.960
7	3129	0.935
8	3129	0.915

rankings based on pairwise preferences [4]. Assume that for some specific pair of problems, such as the spelling of the words "about" and "acknowledge", we first identify all students in the SpellBEE database who have attempted both words. Given that response accuracy is dichotomous, there are only four possible configurations of a student's response accuracy to the pair of challenges. In the cases where the student responds to both correctly or incorrectly, no distinction is made between the pair. But in the cases where the student correctly responds to one but incorrectly to the other, we classify this as a distinction indicating a difficulty ordering between the two problems.[7]

It is also worth stating that in this study, we assume a "static student" model, so we are not concerned with the order of these two responses. At the cost of some data loss, one could instead assume a "learning student" model, for which only a correct response on one problem followed by an incorrect response on the other would define a distinction. Had the incorrect response been observed first, we could not rule out the possibility that the difference was due to a change in the student's abilities over time, and not necessarily an indication of difference in problem difficulties.[8]

An example may clarify. If counting the number of both directional distinctions made by all students (e.g. 12 students in SpellBEE spelled "about" correctly and "acknowledge" incorrectly, while 2 students spelled "about" incorrectly and "acknowledge" correctly), we have a strong indication of relative problem difficulty. McNemar's test assigns a significance to this pair of distinction counts. In this work, we more closely follow the IRT approach, relying only the relative size of the two counts (and not the significance.) Thus, since 12 distinctions were found in one direction and only 2 in the other, we say that we observed the word "about" to be easier than the word "acknowledge" based on collected SpellBEE student data. If distinctions were available for every problem pair,

[7]We recognize that some distinctions are spurious, for which the incorrect response was not reflective of the student's abilities. Here we take a simplistic approach of identifying and ignoring non-responses (in which the student typed nothing) and globally-unique responses (which no other student ever responded, to any challenge.) Globally-unique responses encompass responses from students who don't yet understand the activity, responses from students who did not hear the audio recording, responses from student attempting to use the response field as a chat interface, and responses from students making no effort to engage in the activity.

[8]Another possible model is a "dynamic student" model, for which student abilities may get better or worse over time. Under this model, no distinctions can be definitively attributed to difference in problem difficulty.

a total of 3,129 $\times$ 3,128 = 9,787,512 pairwise problem orderings could be expressed. In our collected data so far, we have 3,349,602 of these problem pairs for which we have distinctions recorded. In the subsequent sections, we measure the fitness of a predictive model (like I_{50}) based on how many of these pairwise orderings are satisfied.[9]

4. Assessing the fit of the NISS-based I_{50} model to the SpellBEE student data

Given the NISS-based I_{50} difficulty model of problem difficulty and the data-driven technique for turning observed distinctions recorded in the SpellBEE database into pairwise difficulty orderings, we can now explore various methods to assess the applicability of the model to the data.

4.1. Assessing fit with a regression model

The first method is to construct a regression model that uses I_{50} to predict observed difficulty. Since observed difficulty is currently available only in pairwise form, this requires an additional step in which we flatten these pairwise orderings into one total ordering over all problems. As this is a highly non-trivial step, the results should be interpreted tentatively. Here, we accomplish a flattening by calculating, for each challenge, the percentage of available pairwise orderings for which the given challenge was the more difficult of the pair. So if 100 pairwise orderings involve the challenge word “acknowledge”, and 72 of these found “acknowledge” to be the harder of the pair, we would mark “acknowledge” as harder than 72% of other words. A regression model was then built on this, using I_{50} as a predictor of the pairwise-derived percentage. The model, after filtering out data points causing ceiling and floor effects (i.e. $I_{50}(w) = 2.0$ or $I_{50}(w) = 8.0$), had an adjusted R^2 value of 0.337 ($p < 0.001$ for the model). The corresponding scatterplot is shown in Figure 3.[10] The relatively low adjusted R^2 value is likely at least partially a result of the flattening step (rather than solely due to poor fit.) Had we flattened the data differently, this value would clearly change. In order to obtain a more reliable measure of model fitness, we seek to avoid any unnecessary processing of the mined data.

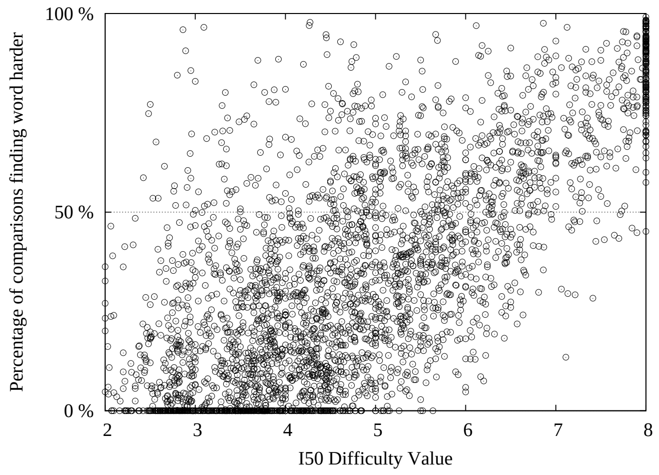
4.2. Assessing fit with as the percentage of agreements on pairwise difficulties

The second method that we explore provides a more direct comparison, without any further flattening of the student data. Here, we simply calculate the percentage of observed pairwise difficulty orderings (across all challenges) for which the I_{50} model correctly predicts the observed pairwise difficulty ordering. When we do this across all of the 3,349,602 difficulty orderings that we have constructed from the student data, we find that the I_{50} model correctly predicts 2,534,228 of these pairwise orderings, providing a 75.66% agreement with known pairwise orderings from the mined data. Remarkably, we found that the predictive accuracy of the I_{50} model did not significantly change as the

[9]Note that it is not be possible to achieve a 100% fit, as some cycles exist among these pairwise orderings.

[10]The outliers in this plot mark the problems that are ranked most differently by the two measures. The word “arithmetic”, for example, was found to be difficult by SpellBEE students, but was not found to be particularly difficult for the students in the NISS study. Variations like this one may reflect changes in the teaching or in the frequency of usage since the NISS study was performed 50 years ago.

Figure 3. Words are plotted by their difficulty on the I_{50} scale and by the percentage of other words for which the observed pairwise orderings found the word to be the harder of the pair. An adjusted R^2 value of 0.490 was calculated for this model. (When ignoring the words affected by a ceiling or floor effect in either variable, the adjusted R^2 value drops to 0.377.)



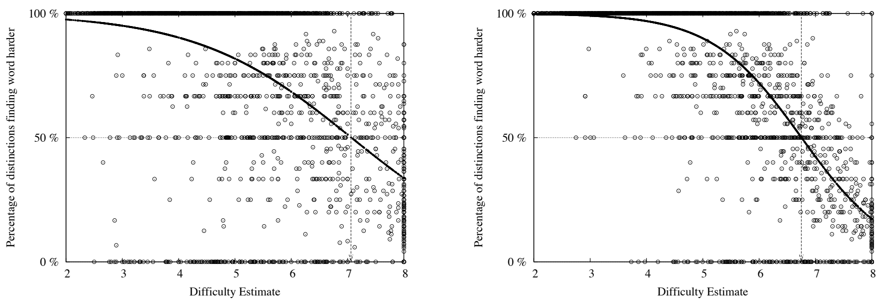
quantity of student data used for the distinction varied. 75.1% of predictions based on one distinction were accurate, while 74.7% of predictions based on 25 distinctions were accurate (intermediate values ranged from 71.0% to 77.6%). This flat relationship suggests that pairwise difficulty orderings constructed from a minimal amount of observed data may be just as accurate, in the aggregate, as those orderings constructed when additional data is available.

5. Incorporating SpellBEE student data in a revised difficulty model

We now know that there is a 75.66% agreement in pairwise difficulty orderings between the I_{50} difficulty metric derived from the NISS data and the observed pairwise preferences mined from the SpellBEE database. Can we improve upon this? We will present an approach that iteratively updates the I_{50} problem difficulty estimates using the mined data and a logistic regression model. Rather than producing a single predictive model, we construct one logistic model for each challenge, and use these fitted model to update our estimates of the problem difficulty. Applied iteratively, we hope to converge on problem difficulty metric that better fits the observed data. This process is inspired by the parameter estimation procedures for Rasch models [11], which may not be directly applicable due to the large size of our problem space.

For a given challenge c_1 (e.g. "acknowledge"), we can first generate the list of all other challenges for which SpellBEE students have expressed distinctions (in either direction.) In Section 3.2, we chose to censor these distinctions in order to generate a bi-

Figure 4. A logistic regression model is used to estimate the difficulty of the word "abandon." At left, the first estimate is based on the original I_{50} difficulty values. At right, the third iteration of the estimate is constructed based on data from the previous best estimate. The point estimate dropped from 8.0 (from I_{50}) to 7.06 (from iteration 1) to 6.81 (from iteration 3.)



nary value representing the difficulty ordering. Here we will make use of the actual distinction counts in each direction. For each challenge with which pairwise distinctions for c_1 are available, we note our current-best estimate of the difficulty of c_2 (initially, using I_{50} values), and note the number of distinctions indicating that c_1 is the more difficult challenge. We can then regress the grouped distinction data on the problem difficulty estimate data to construct a logistic model relating the two. For some c_1, if the relationship is statistically significant, we can use it to generate a revised estimate for the difficulty of that challenge. By solving the regression equation for the c_2 problem difficulty value for which 50% of distinctions find c_1 harder, we can calculate the difficulty of a problem for which relative-difficulty distinctions are equally likely in either direction. This provides a revised estimate for the difficulty of the original problem, c_1. We use this procedure to calculate revised estimates for every challenge in the space (unless the resulting logistic regression model is statistically not significant, in which case we retain our previous difficulty estimate.) This process can be iteratively repeated, using the revised difficulty estimates as the basis of the new regression models. Figure 4 plots this data for one word, using the difficulty estimates resulting from the third iteration of the estimation.

A second approach towards incorporating observed distinction data into a unified problem difficulty scale is briefly introduced and compared to the other metrics. Here, we recast the estimation problem as a sorting problem, and use a probabilistic variant of the bubble-sort algorithm to reorder consecutive challenges based on available distinction data. Initially ordering the challenge words alphabetically, we repeatedly step through the list, reordering challenges at indices i and $i+1$ with a probability based on the proportion of distinctions finding the first challenge harder than the second.[11] After "bubbling" through the ordered list of challenges 200,000 times, we interpret the rank-order of each challenge as a difficulty index. These indices provide a metric of difficulty (which we refer to as $ProbBubble$), and a means for predicting the relative difficulty of any pair of challenges (based on index ordering.)

[11]If distinctions have been observed in both directions, the challenges are reordered with a probability determined by the proportion of distinctions in that direction. If no distinctions in either direction have been observed, the challenges are reordered with a probability of $p = 0.5$. If distinctions have been observed in one direction but not the other, the challenges are reordered with a fixed minimal probability ($p = 0.1$).

Table 2. Summary table for the predictive accuracy of various difficulty metrics. For each metric, the percentage of accurate predictions of pairwise difficulty orderings is noted. The accuracy of the I_{50} metric is measured against all of the 3,349,602 pairwise orderings identified by student distinctions. The accuracy of the data-driven metrics ($I_{50}rev.1$ and $ProbBubble$) are based on the average results from a 5-fold cross-validation, in which the metrics are constructed or trained on a subset of the pairwise distinction data and are evaluated on a different set of pairwise data (the remaining portion.)

Difficulty Model	*Predictive Accuracy*
I_{50}	75.66%
$I_{50}rev.1$	84.79%
$ProbBubble$	84.98%

Table 3. Spearman's rank correlation coefficient between pairs of problem difficulty rank-orderings ($N = 3129$, $p < 0.01$, two-tailed.)

Metric 1	Metric 2	Spearman's ρ
I_{50}	$I_{50}rev.3$	0.677
I_{50}	$ProbBubble$	0.673
$I_{50}rev.3$	$ProbBubble$	0.908

Given the pairwise technique used in Section 4.2 for analyzing the fit of a difficulty metric for a set of pairwise difficulty orderings, we can examine how these two data-driven models compare to the original I_{50} difficulty metric. Table 2 summarizes our findings. Here we observe that the data-driven approaches provide an improvement of almost 10% accuracy with regard to the prediction of pairwise difficulty orderings. As was noted earlier, cycles in the observed pairwise difficulty orderings prevent any linear metric from achieving 100% prediction accuracy, and the maximum achievable accuracy for the SpellBEE student data is not know. We do note that two different data-driven approaches, logistic regression-based iterative estimation and the probabilistic sorting, arrived at very similar levels of predictive accuracy. Table 3 uses Spearman's rank correlation coefficient as a tool to quantitatively compare the three metrics. One notable finding here is the extremely high rank correlation between the $ProbBubble$ and $I_{50}rev.3$ data-driven metrics.

6. Conclusion

The findings from the research questions posed here are both reassuring and revealing. Although the NISS study was done over 50 years ago, much of its value seems to have been retained. The NISS-based I_{50} difficulty metric was observed to correctly predict 76% of the pairwise difficulty orderings mined from SpellBEE student data. Many of the challenges for which the difficulty metric achieved low predictive accuracies corresponded with words whose cultural relevance or prominence has changed over the past few decades. The data-driven techniques presented in Section 5 offers a means for incorporating these changes back into a difficulty metric. After doing so, we found the predictive accuracy increased approximately 10%, to the 85% agreement level.

The key technique used here to enable the assessment and improvement of problem difficulty estimates works even when not all students have attempted all challenges or when the selection of challenges for students is highly biased. It is data-driven, based on identifying and counting pairwise distinctions indicated indirectly through observations of student behavior over the duration of use of an education system. The pairwise distinction-based techniques for estimating problem difficulty information explored here is a part of a larger campaign to develop methods for constructing educational systems that require a minimal amount of expert domain knowledge and model-building. Our BEEweb model is but one such approach, the Q-matrix method is another [3], and most the IRT-based systems discussed in the introduction are, also. Designing BEEweb activities only requires domain knowledge in the form of a problem difficulty function and a response accuracy function. The latter can usually be created without expertise, and the former can now be approached, even when collected data is sparse and biased, using the techniques discussed in this paper.

References

[1] Ari Bader-Natal and Jordan B. Pollack. Motivating appropriate challenges in a reciprocal tutoring system. In C.-K. Looi, G. McCalla, B. Bredeweg, and J. Breuker, editors, *Proceedings of the 12th International Conference on Artificial Intelligence in Education (AIED-2005)*, pages 49–56, Amsterdam, July 2005. IOS Press.

[2] Ari Bader-Natal and Jordan B. Pollack. BEEweb: A multi-domain platform for reciprocal peer-driven tutoring systems. In M. Ikeda, K. Ashley, and T.-W. Chan, editors, *Proceedings of the 8th International Conference on Intelligent Tutoring Systems (ITS-2006)*, pages 698–700. Springer-Verlag, June 2006.

[3] Tiffany Barnes. The q-matrix method: Mining student response data for knowledge. Technical Report WS-05-02, AAAI-05 Workshop on Educational Data Mining, Pittsburgh, 2005.

[4] Klaus Brinker, Johannes Fürnkranz, and Eyke Hüllermeier. Label ranking by learning pairwise preferences. *Journal of Machine Learning Research*, 2005.

[5] Leonard S. Cahen, Marlys J. Craun, and Susan K. Johnson. Spelling difficulty – a survey of the research. *Review of Educational Research*, 41(4):281–301, October 1971.

[6] Chih-Ming Chen, Chao-Yu Liu, and Mei-Hui Chang. Personalized curriculum sequencing utilizing modified item response theory for web-based instruction. *Expert Systems with Applications*, 30, 2006.

[7] Bruce Choppin. A fully conditional estimation procedure for rasch model parameters. CSE Report 196, Center for the Study of Evaluation, University of California, Los Angeles, 1983.

[8] Ricardo Conejo, Eduardo Guzmán, Eva Millán, Mónica Trella, José Luis Pérez-De-La-Cruz, and Antonia Ríos. Siette: A web-based tool for adaptive testing. *International Journal of Artificial Intelligence in Education*, 14:29–61, 2004.

[9] Michel C. Desmarais, Shunkai Fu, and Xiaoming Pu. Tradeoff analysis between knowledge assessment approaches. In C.-K. Looi, G. McCalla, B. Bredeweg, and J. Breuker, editors, *Proceedings of the 12th International Conference on Artificial Intelligence in Education (AIED-2005)*. IOS Press, 2005.

[10] B. S. Everitt. *The Analysis of Contingency Tables*. Chapman and Hall, 1977.

[11] Gerhard H. Fischer and Ivo W. Molenaar, editors. *Rasch Models: Foundations, Recent Developments, and Applications*. Springer-Verlag, New York, 1995.

[12] Harry A. Greene. *New Iowa Spelling Scale*. State University of Iowa, Iowa City, 1954.

[13] Jeff Johns, Sridhar Mahadevan, and Beverly Woolf. Estimating student proficiency using an item response theory model. In M. Ikeda, K. Ashley, and T.-W. Chan, editors, *Proceedings of the 8th International Conference on Intelligent Tutoring Systems (ITS-2006)*, pages 473–480, 2006.

[14] Mark Wilson and R. Darrell Bock. Spellability: A linearly ordered content domain. *American Educational Research Journal*, 22(2):297–307, Summer 1985.

Toward the extraction of production rules for solving logic proofs

Tiffany Barnes, John Stamper
Department of Computer Science, University of North Carolina at Charlotte
tbarnes2@uncc.edu, jcstampe@uncc.edu

Abstract: In building intelligent tutoring systems, it is critical to be able to understand and diagnose student responses in interactive problem solving. However, building this understanding into the tutor is a time-intensive process usually conducted by subject experts. Much of this time is spent in building production rules that model all the ways a student might solve a problem. We propose a novel application of Markov decision processes (MDPs), a reinforcement learning technique, to automatically extract production rules for an intelligent tutor that learns. We demonstrate the feasibility of this approach by extracting MDPs from student solutions in a logic proof tutor, and using these to analyze and visualize student work. Our results indicate that extracted MDPs contain many production rules generated by domain experts and reveal errors that experts do not always predict. These MDPs also help us identify areas for improvement in the tutor.

Keywords: educational data mining, Markov decision processes

1. Introduction

According to the ACM computing curriculum, discrete mathematics is a core course in computer science, and an important topic in this course is solving formal logic proofs. However, this topic is of particular difficulty for students, who are unfamiliar with logic rules and manipulating symbols. To allow students extra practice and help in writing logic proofs, we are building an intelligent tutoring system on top of our existing proof verifying program. Our experience in teaching discrete math, and in student surveys, indicate that students particularly need feedback when they get stuck.

The problem of offering individualized help and feedback is not unique to logic proofs. Through adaptation to individual learners, intelligent tutoring systems (ITS) can have significant effects on learning [1]. However, building one hour of adaptive instruction takes between 100-1000 hours of work of subject experts, instructional designers, and programmers [2], and a large part of this time is used in developing production rules that are used to model student behavior and progress. A variety of approaches have been used to reduce the development time for ITSs, including ITS authoring tools (such as ASSERT and CTAT), or building constraint-based student models instead of production rule systems. ASSERT is an ITS authoring system that uses theory refinement to learn student models from an existing knowledge base and student data [3]. Constraint-based tutors, which look for violations of problem constraints, require less time to construct and have been favorably compared to cognitive tutors, particularly for problems that may not be heavily procedural [4].

Some systems, including RIDES, DIAG, and CTAT use teacher-authored or demonstrated examples to develop ITS production rules. RIDES is a "Tutor in a Box" system used to build training systems for military equipment usage, while DIAG was built as an expert diagnostic system that generates context-specific feedback for students [2]. These systems cannot be easily generalized, however, to learn from student data. CTAT has been used to develop "pseudo-tutors" for subjects including genetics, Java, and truth tables [5]. This system has also been used with data to build initial models for an ITS, in an approach called Bootstrapping Novice Data (BND) [6].

Similar to the goal of BND, we seek to use student data to directly create student models for an ITS. However, instead of feeding student behavior data into CTAT to build a production rule system, we propose to generate Markov Decision Processes that represent all student approaches to a particular problem, and use these MDPs directly to generate feedback. We believe one of the most important contributions of this work is the ability to generate feedback based on frequent, low-error student solutions.

We propose a method of automatically generating production rules using previous student data to reduce the expert knowledge needed to generate intelligent, context-dependent feedback. The system we propose is capable of continued refinement as new data is provided. We illustrate our approach by applying MDPs to analyze student work in solving formal logic proofs. This example is meant to demonstrate the applicability of using MDPs to collect and model student behavior and generate a graph of student responses that can be used as the basis for a production rule system.

2. Background and Proofs Tutorial Context

Several computer-based teaching systems, including Deep Thought [7], CPT [8] and the Logic-ITA [9] have been built to support teaching and learning of logic proofs. Of these, the Logic-ITA is the most intelligent, verifying proof statements as a student enters them, and providing feedback after the proof is complete on student performance. Logic-ITA also has facilities for considerable logging and teacher feedback to support exploration of student performance [9], but does not offer students help in planning their work. In this research, we propose to augment our own existing Proofs Tutorial, with a cognitive architecture derived using educational data mining, that can provide students feedback to avoid error-prone solutions, find optimal solutions, and inform students of other student approaches.

In [10], the first author has applied educational data mining to analyze completed formal proof solutions for automatic feedback generation. However, this work did not take into account student errors, and could only provide general indications of student approaches, as opposed to feedback tailored to a student's current progress. In this work, we explore all student attempts at proof solutions, including partial proofs and incorrect rule applications, and use visualization tools to learn how this work can be extended to automatically extract a production rule system to add to our logic proof tutorial. In [11], the second author performed a pilot study to extract Markov decision processes for a simple proof from three semesters of student data from Deep Thought, and verified that the rules extracted by the MDP conformed with expert-derived rules and generated buggy rules that surprised experts. In this work, we apply the technique and extend it with visualization tools to new data from the Proofs Tutorial.

The Proofs Tutorial is a computer-aided learning tool implemented on NovaNET (http://www.pearsondigital.com/novanet/). This program has been used for practice and feedback in writing proofs in university discrete mathematics courses taught by the first author and others at North Carolina State University since 2002. In the Proofs Tutorial, students are assigned a set of 10 problems that range from simpler logical equivalence applications to more complex inference proofs. (The tutorial can check arbitrary proofs, but we use it for a standard set of exercises). In the tutorial, students type in consecutive lines of a proof, which consist of 4 parts: the statement, reference lines, the axiom used, and the substitutions which allow the axiom to be applied. After the student enters these 4 parts to a line, the statement, reference lines, axiom, and substitutions are verified. If any of these conditions does not hold, a warning messageis shown, and the line is deleted (but saved for later analysis).

In this work, we examine student solutions to Proof 1. Table 1 lists an example student solution. Figure 2 is a graphical representation of this proof, with givens as white circles, errors as orange circles, and premises as rectangles.

Table 1: Sample Proof 1 Solution (red lines are errors)

	Statement	Line	Reason
1.	a → b		Given
2.	c → d		Given
3.	¬ (a → d)		Given
	¬ a v d	**3**	**rule IM (error)**
4.	a ^ ¬ d	3	rule IM implication
5.	a	4	rule S simplification
	b	**4**	**rule MP (error)**
	b	**1**	**rule MP (error)**
6.	b	1,5	rule MP modus ponens
7.	¬ d	4	rule S simplification
8.	¬c	2,7	rule MT modus tollens
9.	b ^ ¬c	6,8	rule CJ conjunction

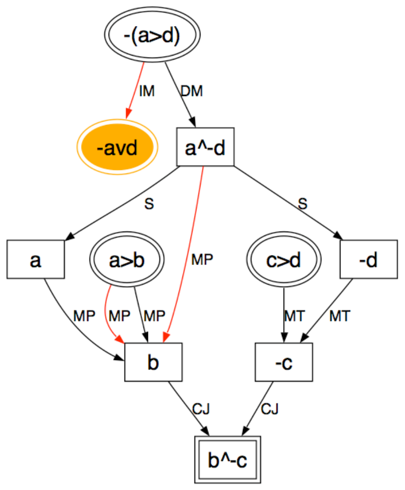


Figure 2: Graph of Proof 1 Solution (Red lines are errors)

3. Markov Decision Processes and ACT-R

A Markov decision process (MDP) is defined by its state set S, action set A, transition probabilities P, and a reward function R [12]. On executing action a in state s the probability of transitioning to state s´ is denoted P(s´ | s, a) and the expected reward associated with that transition is denoted R(s´|s, a). For a particular point in a student's proof, our method takes the current premises and the conclusion as the state, and the student's input as the action. Therefore, each proof attempt can be seen as a graph with a sequence of states (each describing the solution up to the current point), connected by actions. We combine all student solution graphs into a single graph, by taking the union of all states and actions, and mapping identical states to one another. Once this graph is constructed, it represents all of the paths students have taken in working a proof. Typically, at this step reinforcement learning is used to find an optimal solution to the MDP. We propose instead, to create multiple functions to deliver different types of feedback, such as functions that could:

1. Find expert-like paths. To derive this policy, we would assign a high reward to the goal state, negative rewards to error states and smaller negative rewards for taking an action. This function would return an optimal (short and correct) choice, and hence expert feedback, at every point in the MDP.
2. Find a typical path to the goal state. We could derive this policy by assigning high rewards to successful paths that many students have taken. Vygotsky's theory of the zone of proximal development [13] states that students are able to learn new things that are closest to what they already know. Presumably, frequent actions could be those that more students feel fluent using. Therefore, paths based on typical student behavior may be more helpful than optimal solutions, which may be above a student's current ability to understand.
3. Find paths with low probabilities of errors. It is possible that some approaches could be much less error-prone than others. We could this policy by assigning large penalties to error states. Students often need to learn a simple method that is easily understood, rather than an elegant but complex solution.

These MDPs are also intended to serve as the basis for learning a production rule system that will perform model tracing, as in a cognitive tutor, while a student is solving a problem. ACT-R Theory [1] is a cognitive architecture that has been successfully applied in the creation of cognitive tutors, and consists of declarative and procedural components. The procedural module contains a production rules system, and creating its production rules is the most time consuming part of developing an ITS based on ACT-R. A production rule system consists of the current problem state (called working memory in ACT-R), a set of production rules, and a rule interpreter. A production rule can be seen as a condition-action pair such as "if A then C" with associated probability x. An example production rule for solving proofs is "if the goal is to prove b^-c, then set as subgoal to prove b". The rule interpreter performs model tracing to find a sequence of productions that produce the actions a student has taken. This allows for individualized help, tracks student mastery (using the correctness of the productions being applied), and can provide feedback on recognized errors.

Production rules in ACT-R are of a general nature to allow them to apply to multiple problems. However, they could be written very specifically to apply to a particular problem. An MDP extracted for a particular problem could be seen as a set of production rules that describe the actions a student might take from each problem state. In this case, MDP state-action pairs are production rules, where the full current problem state is the condition and the action is the applying a particular axiom.

By building an MDP using data from several semesters, we generate a significant number of rules and record probabilities that reflect different student approaches. We can use this MDP as it is to perform model tracing as a student is working a proof. If a student is working the problem in a way that has already been encountered, we can use the MDP to provide feedback. If he or she asks for help, we can direct that student to optimal, frequent, or simply away from error-prone paths, based on that particular student and/or path. Similarly to constraint-based tutors [4], if a student is solving a proof in a way that is not already present in our MDP, we simply add their steps to our model. If such a student does commit an error, only default feedback will be given.

As a first step, MDPs created from student data can be used to add intelligent feedback to every problem. This would require storing the MDP, adding a process to the end of the statement checking to match the student's state to the MDP, and if it is found, adding a hint option that would reveal the optimal next choice in the MDP. If a student's state is not found in the MDP, we can add it (storing its correctness), and periodically run reinforcement learning to update the reward function values.

Our next step will be to apply machine learning to the MDP to learn more general rules for building a true production rule system. The list of axioms itself can be used as general production rules, and represent most valid student actions. For example, Modus Ponens can be expressed as the production rule: "If A and A→B, then apply Modus Ponens to get B". However, there are no priorities assigned to the axioms themselves. We can cycle through the states of the MDP, examining all cases where the premises hold. We can use the MDP reward functions or the frequency of student use, or a combination of these, to set priorities for each axiom. Then when more than one axiom applies, we know which one was most frequently used with success (and therefore which one should fire).

Alternatively, we could apply machine learning techniques to our MDPs to find common subgraphs in student approached, or build hierarchical MDPs to group substrategies in proof solutions. We believe this approach could be done in a general way so that it could be applied to other domains. Toward that end, we have generated visualizations of our MDPs to investigate the structure of student solutions, what errors they commit, and where they occur most frequently. This process can provide insight into where machine learning could provide the most benefit. In addition, this analysis can help us discover areas for improvement in the ITS.

4. Method

This experiment uses data from the four fall semesters of 2003-2006, where an average of 120 students take the discrete math course at NC State University each fall. Students in this course are typically engineering and computer science students in their second or third year of college, but most have not been exposed to a course in logic. Students attend several lectures on propositional logic and complete an online homework where students complete truth tables and fill in the blanks in partially-completed proofs. Students then use the Proofs Tutorial to solve 10 proofs as homework, directly or using proof by contradiction. The majority of students used direct proof to solve proof 1. We extracted 429 of students' first attempts at direct solutions to proof 1 from the Proofs Tutorial. We then removed invalid data (such as proofs with only one step to reach the conclusion), resulting in 416 student proofs. Of these, 283 (70%) were complete and 133 (30%) were partial proofs. Due to storage limitations, a few (6) of these proofs may have been completed by students but not fully recorded. The average lengths were 13 and 10 lines, respectively, for completed and partial proofs. This indicates that students did attempt to complete the proof.

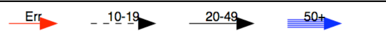
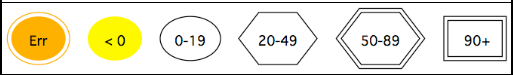
After cleaning the data, we load the proofs into a database and build an MDP for the data. We then set a large reward for the goal state (100) and penalties for incorrect states (10) and a cost for taking each action (1). Setting a non-zero cost on actions causes the MDP to penalize longer solutions (but we set this at 1/10 the cost of taking an incorrect step). These values may need to be adjusted for different sizes of MDPs. We apply the value iteration Bellman backup reinforcement learning technique to assign reward values to all states in the MDP. The equation for calculating values V(s) for each state s, where R(s) is the reward for the state, γ is the discount factor (set to 1), and $P_a(s,s')$ is the probability that action a will take state s to state s':

$$V(s) := R(s) + \gamma \max_a \sum_{s'} P_a(s,s')\, V(s')$$

For value iteration, V is calculated for each state until there is little change in the value function over the entire state space. Once this is complete, the optimal solution in the MDP corresponds to taking a greedy traversal approach in the MDP [12]. The rewards for each state then indicate how close to the goal a state is, while probabilities of each transition reveal the frequency of taking a certain action in a certain state. For the purposes of this paper, just one step of value iteration was performed to cascade goal values through the MDP.

We then ran the MDP for each semester of data, and for the aggregate. This resulted in a set of states, reward values, and actions for each MDP. On average, the four semesters yielded 158 states (ranging from 95-226 states in a semester). The most frequent approaches to problems were very similar across semesters, and the most frequent errors were also repeated. Therefore, in the remainder of this paper, we examine the aggregate MDP. Using Excel®, we assigned labels to each state in the MDP (just using the latest premise added), colors for errors, state values, and action frequencies, and prepared the data for display. We used GraphViz to display the output and convert into pictures. Table 2 shows the legend for nodes and edges. After graphing each MDP, we continually refined the data being displayed to explore questions about the student data. We present our findings in the following section.

Table 2: Legend for MDP edges and nodes

Edges (Values=Frequency)	Nodes (Values=Rewards)
	

5. Results

The aggregate MDP run on all four semesters of data has a total of 547 states (each semester alone generated between 95-226 unique states). Since it is hard to view statically, we omit it here. From interactive exploration, we found that 90% of all student errors related to explaining their actions, and a great majority of these were on the simplification rule. We plan to improve the interface for this explanation. We also found that students commit a great number of errors in the first step but less as they progress. To assist in visualizing student behavior, we plan to build a tool for teacher visualization that will allow for pruning nodes below a certain reward or frequency, and also for highlighting all the correct or incorrect applications of a particular action. We demonstrate some of these views in Figures 3-4.

Figure 3 shows the aggregate MDP restricted to only valid states and frequent state-action pairs. The graph shows only one frequent successful path – indicating an optimal solution that students can perform. This path also corresponds to an expert solution. On this path, it seems that errors are occurring in going from state (a^-d) to (a), demonstrating our observation that applying simplification is difficult for students. We note many errors at the start, indicating that students have trouble getting started. Following the shaded path, second from the lowest in Figure 4, we observe that several (20-49) students apply IM in various ways; this path is *very* error-prone and does not (frequently) lead to a solution. This indicates a need to help students plan proof strategies. We can detect this type of path in the MDP and offer hints to help avoid it.

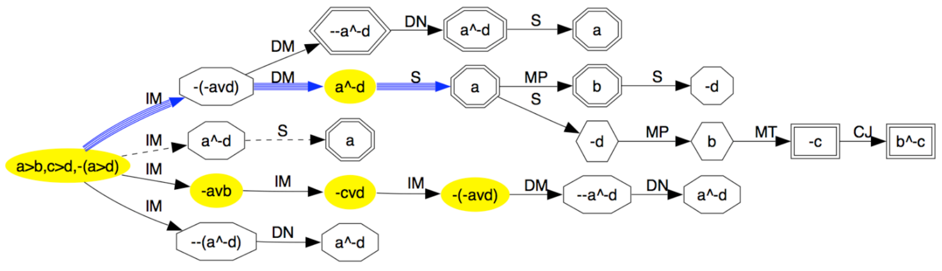


Figure 3: View of MDP restricted to valid states and frequent actions

To further examine student approaches, we expand this graph to include error states in Figure 4. In most errors, students find the correct premise (e.g. –d, which is correct) but have trouble explaining how the rule was applied to get the premise. For example, the rule for S (simplification) states (p^q)→p, so to obtain (a^-d)→-d the student must show that we substitute p=-d and q=a. We conclude that we need a better interface to help students show how a rule applies. For example, in Deep Thought, students are not required to perform substitutions if the program can detect them itself, and for simplification, the program asks: Right or Left? Anecdotally and intuitively, students seem to have less trouble with this approach. All but two of the remaining error states (indicated with darker shading and a double border) are due to substitution errors. (Start) → (–(a^-d)) is an incorrect application of IM (it's missing a not), while Contrapositive (CP) was frequently applied to obtain –c from c>d and –d (which needs Modus Tollens, MT). These findings indicate that more practice might be needed with these rules in the context of negated variables.

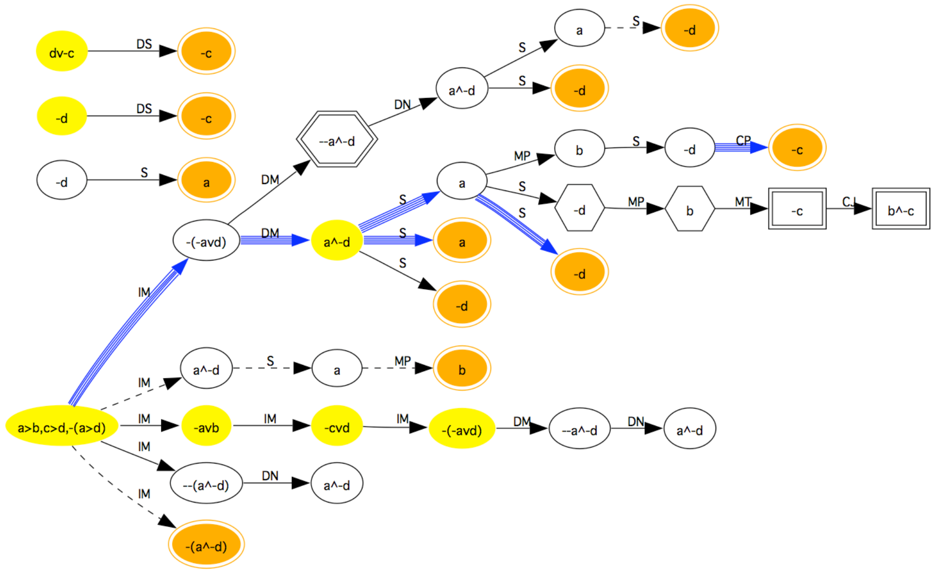


Figure 4: View of MDP restricted to frequent states and actions

These visualizations are useful in predicting what the MDP will find when generating feedback based on frequent responses. However, this does not yield insight into a more general application of MDPs to proofs and what types of production rules might be generated for general problems. To make more general production rules for proof problems, we will take MDPs from several problems and attempt to learn general rules. For instance, a production rule often applied by experts is, "if (p → q) and (p) are premises then apply MP to obtain the new premise (q)". To learn this, we need to break MDP states down into their constituent premises.

We created Figure 5, (which is a graph, not an MDP), by mapping all correct states with a common last premise into a single node, which corresponds to grouping unique action/premise pairs. We then eliminated all low-frequency actions and the simplification rule (since we plan to change the interface to improve usage of this rule). This new visualization of the paths of student solutions allows us to track frequent solution attempts regardless of order. In other words, our previous MDP views correspond to unique paths, while this graph shows us relationships between consecutive steps in the proof regardless of what was done before. In Figure 5, there are some dead-end paths, meaning that several students started proofs in these directions but few students were able to reach the solution this way. These dead-end paths can be used to derive feedback to students that their approach may not be productive. We can also use Figure 5 to derive most frequent orderings of student solutions. To do so, we start at the (top) start node and choose a frequent (wide) edge, and repeat without visiting nodes twice, until we reach the solution, as in Figure 6.

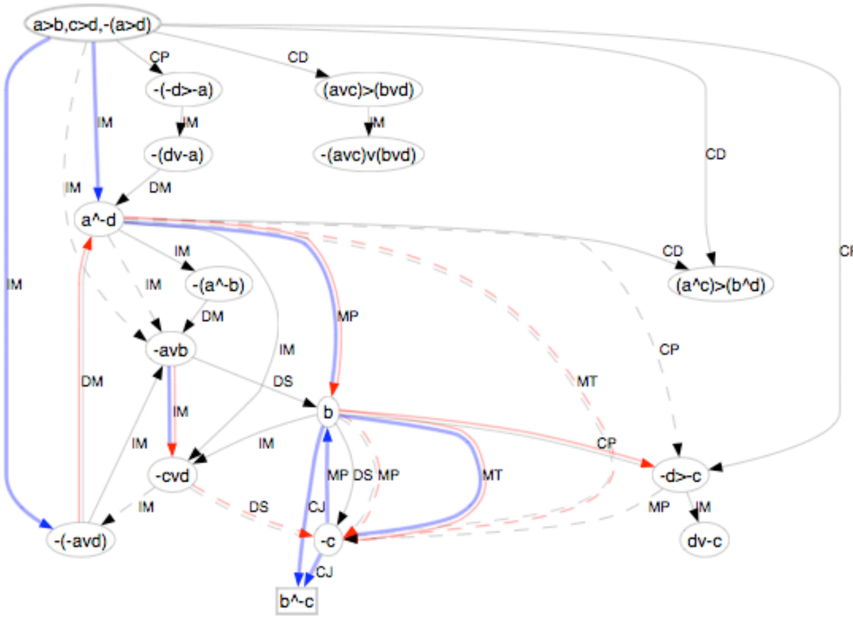


Figure 5: Graph showing inferences, unique premises and frequent (>9) actions

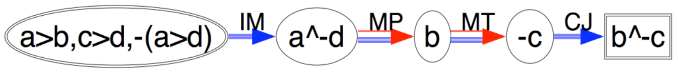


Figure 6: Most frequent proof solution sequences, derived from Figure 5

Some secondary approaches are shown in Figure 7, demonstrating how a teacher could use Figure 5 by following particular paths but not repeating any nodes visited, to find unique solutions to the proof. These approaches demonstrate students' frequent preference to use Disjunctive Syllogism (DS), even though these solutions are longer.

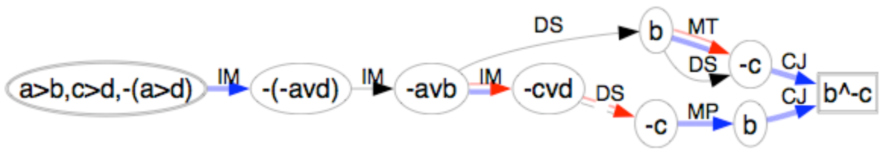


Figure 7: Secondary proof approaches derived from Figure 5

In both Figures 6 and 7, we see that there are errors (on the double edges) leading to and from states containing negated variables. This observation reflects our experience that students need more practice in using rules when variables are of negated, and in applying rules in sequence to achieve a goal.

6. Conclusions and Future Work

We have proposed a general approach to mining Markov decision processes from student work to automatically generate production rules and discussed how this approach can be applied to a particular CBT. This approach differs from prior work in authoring tutoring systems by mining actual student data, rather than relying on teachers to add examples the system can learn from. We have explored visualizations of the Markov decision processes extracted from solutions to a formal logic proof to determine how to improve the tutor and how we might proceed in building useful production rules and feedback. We believe that the process we have applied to creating problem visualizations can be useful in learning about other problem-solving processes from student data, instead of creating expert systems by hand. From these visualizations, we have learned:

1. Although we hypothesized that students need help in planning, this did not seem to be the case. Instead, students needed help on getting started.
2. As we expected, students need more practice with negation.
3. The overwhelming majority of student errors were in explaining rule applications. We plan to add a better interface for this step.

We have also concluded that the extracted MDPs will be useful in generating student feedback. The extracted MDP does contain a frequent expert-like path and contains a significant number of correct paths and student errors. Our tutor can already classify many errors students make. Adding the MDP to this tutor will enable it to model student mastery, provide hints, and feedback on errors. This MDP can constantly learn from new student data. We note that on cold start for a new problem that has no student data, the system will still act as a problem-solving environment, but after even one semester of data is collected, a limited amount of feedback can be generated. As more data are added, more automated feedback can be generated. In our future work we plan to implement this system in our tutor and in Deep Thought, a logic tutor created by Marvin Croy [7]. Once implemented, we will test the feedback generated based on the MDP, and we will continue to explore ways to learn general rules to build production rule systems with greater coverage and robustness.

A novel contribution of this work is the idea of providing feedback on the most frequent approach to a problem. It is often the case that students are not ready to apply optimal solutions to a problem. However, detecting this readiness is a challenging user modeling problem. It may be possible that student feedback based on frequency rather than optimality can provide most students with the help they need. This type of feedback may also encourage student reflection, when the tutor is no longer all-knowing, but has a memory that students tap into instead. In our future work we plan to investigate the impact of this type of feedback on student learning.

7. References

[1] Anderson, J., Gluck, K. 2001. What role do cognitive architectures play in intelligent tutoring systems? In D. Klahr & S. Carver (Eds.) *Cognition & Instruction: 25 years of progress*, 227-262. Erlbaum.

[2] Murray, Tom. 1999. Authoring intelligent tutoring systems: An analysis of the state of the art. *Intl. J. Artificial Intelligence in Education*, 10: 98-129.

[3] Baffes, P. & Mooney, R.J. 1996. A novel application of theory refinement to student modeling. *Proc. AAAI-96*, pp. 403-408, Portland, OR, August 1996.

[4] Mitrovic, A., Koedinger, K. & Martin, B. 2003. A comparative analysis of cognitive tutoring and constraint-based modeling. *User Modeling 2003*: 313-322.

[5] Koedinger, K. R., Aleven, V., Heffernan. T., McLaren, B. & Hockenberry, M. 2004. Opening the door to non-programmers: Authoring intelligent tutor behavior by demonstration. *Proc. 7th Intelligent Tutoring Systems Conference,* Maceio, Brazil, pp. 162-173.

[6] McLaren, B., Koedinger, K., Schneider, M., Harrer, A., & Bollen, L. 2004. Bootstrapping Novice Data: Semi-automated tutor authoring using student log files, In *Proc. Workshop on Analyzing Student-Tutor Interaction Logs to Improve Educational Outcomes, 7th Intl. Conf. Intelligent Tutoring Systems (ITS-2004),* Maceió, Brazil, August 30, 2004.

[7] Croy, M. 2000. "Problem Solving, Working Backwards, and Graphic Proof Representation," *Teaching Philosophy*, 23(2), 169-187.

[8] Scheines, R., & Sieg, W. 1994. Computer environments for proof construction. *Interactive Learning Environments*, 4(2), 159-169.

[9] Merceron, A., & Yacef, K. 2005. Educational data mining: a case study. In: C.-K. Looi, G. McCalla, B. Bredeweg & J. Breuker (eds.), (*Proc. 12th Intl. Conf. Artificial Intelligence*), pp. 467–474. Amsterdam: IOS Press.

[10] Barnes, T. 2006. Evaluation of the q-matrix method in understanding student logic proofs. *Proc. 19th Intl. Florida AI Research Society Conference* (*FLAIRS 2006)*, Melbourne Beach, FL, May 11-13, 2006.

[11] Stamper. J. 2006. Automating the Generation of Production Rules for Intelligent Tutoring Systems. *Proc. Intl. Conf. Computer-Aided Learning (ICL2006)*. Austria, Sep. 27-29, 2006.

[12] Richard S. Sutton and Andrew G. Barto. 1998. *Reinforcement Learning: An Introduction*. The MIT Press, Cambridge, MA.

[13] Vygotsky, L. (1986). *Thought and language*. Cambridge, MA: MIT Press.

Difficulties in inferring student knowledge from observations (and why you should care)

Joseph E. Beck
joseph.beck@EducationalDataMining.org
Machine Learning Department
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA USA

Abstract. Student modeling has a long history in the field of intelligent educational software and is the basis for many tutorial decisions. Furthermore, the task of assessing a student's level of knowledge is a basic building block in the educational data mining process. If we cannot estimate what students know, it is difficult to perform fine-grained analyses to see if a system's teaching actions are having a positive effect. In this paper, we demonstrate that there are several unaddressed problems with student model construction that negatively affect the inferences we can make. We present two partial solutions to these problems, using Expectation Maximization to estimate parameters and using Dirichlet priors to bias the model fit procedure. Aside from reliably improving model fit in predictive accuracy, these approaches might result in model parameters that are more plausible. Although parameter plausibility is difficult to quantify, we discuss some guidelines and propose a derived measure of predicted number of trials until mastery as a method for evaluating model parameters.

Keywords: student modeling, mastery learning, parameter estimation, Bayesian networks, Dirichlet priors

1 Introduction

The focus of this paper is on the difficulties in mapping observable student performance to estimate his level of knowledge about underlying skills. This task, better known as student modeling, has been around for at least three decades [1]. Given the long history, it is fair to ask whether there are large, fundamental problems to be overcome. We discuss several problems resulting from the existence of local-maxima as well as multiple global-maxima in the parameter search space for a common student modeling approach. These problems may seem esoteric, but they result in seemingly similar models that make very different claims about the student's knowledge. There are two direct impacts of the results presented in this paper on Educational Data Mining (EDM). First, since estimating student knowledge is relatively well understood, it should be one of the easiest tasks for us to analyze. Given the surprising complexity and difficulty in evaluating models, we should be cautious about our analyses of more exotic phenomena. Second, fine-grained estimates of student knowledge are a useful lever in attacking other data mining tasks. For example, it is difficult to "measure the effect of individual interventions" (from the workshop's call for papers) if we cannot track student knowledge.

The goal of student modeling is to take observations of a student's performance and use those to estimate the student's knowledge, goals, preferences, and other latent characteristics. The key aspect of the problem is that the characteristics of interest are not directly observable. Thus, some means of mapping the observations to characteristics is required. Knowledge tracing [2] is an example of such a technique, and is the one we will focus on in the explorations in this paper. Knowledge tracing, shown in Figure 1, is an approach for taking student observations and using those to estimate the student's level of knowledge. It assumes that students either know a skill or do not; similarly they either respond correctly to an item or not. There are two parameters, *slip* and *guess*, that mediate student knowledge and student performance. These two parameters are called the performance parameters in the model. An assumption of the model is that even if a student knows a skill, there is a chance he might still respond incorrectly to a question that utilizes that skill. This probability is the slip parameter. There are a variety of reasons for an incorrect response, the student could have made a simple typo (e.g. typed '12' instead of '21' for "7 x 3"), he could have been frustrated with the system and behaving randomly, or he may not have felt like solving the problem and would rather have the tutor tell him how to solve it.

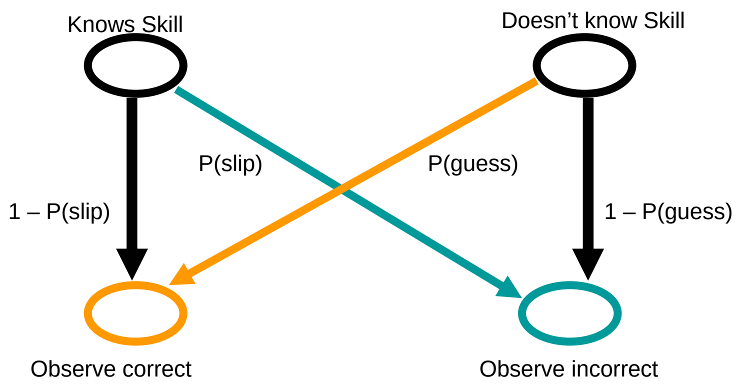


Figure 1. Diagram of knowledge tracing

Conversely, a student who does not know the skill might still be able to generate a correct response. This probability is referred to as the guess parameter. A guess could occur either through blind chance (e.g. in a 4-choice multiple choice test there is a ¼ chance of getting a question right even if one does not understand it), the student being able to utilize a weaker version of the correct rule that only applies in certain circumstances, or through errors in the tutor's scoring mechanism. For example, experiments using knowledge tracing with speech recognition had a very high guess parameter because the speech recognizer had a hard time noticing when students made mistakes [3].

In addition to the two performance parameters, there are two learning parameters. The first is K0, the likelihood the student knows the skill when he first uses the tutor. Initial knowledge can come from a variety of places including material from a prior course or declarative instruction delivered by the tutor before procedural lessons (e.g. [4]). The second learning parameter is t, the probability a student will acquire a skill as a result of an opportunity to practice it. Every skill to be tracked has these four parameters, slip, guess, K0, and t, associated with it.

To train a knowledge tracing model, historical data of student performance is provided to model fitting software. There exist a variety of software packages to perform the optimization such as the curve fitting approach used by some of the cognitive tutors[1] and the Bayes Net Toolkit for Student Modeling (BNT-SM, [5]). To use a knowledge tracing model, when a student begins using an ITS his initial knowledge estimates are set to the K0 estimates for each skill. When a student responds to an item, Bayesian inference is used to update his internal knowledge on the basis of whether he responded correctly or not, and on the skill's slip and guess parameters. The student is then credited with an opportunity to learn the skill via the t parameter.

This work also assumes the tutor's designers already know which skill will be modeled. Although the construction and refinement of domain models from data is an interesting topic (e.g. [6-8]), it is beyond the scope of this paper. This paper uses the knowledge tracing model as a platform for experimentation. We are not asserting knowledge tracing is correct, and in fact would argue the opposite; for one thing, it does not acknowledge that students can forget what they have learned. It is simply an approach that is the closest the AIED community has to a standard. Furthermore, it is the simplest modeling approach that has the promise to be analytically interesting. If we are able to uncover and study basic problems with a model as simple as knowledge tracing, it is probable that more complex modeling schemes will suffer similar drawbacks.

2 Difficulties

Knowledge tracing has two main flaws. The first is that the parameter estimation task is subject to local maxima. That means that given a set of observations, the parameters returned by model fitting software might not the best possible fit to the data. The second problem, less well known, is that there are multiple local maxima. That is, there can be more than one set of parameters that fit the data equally well.

[1] Source code is courtesy of Albert Corbett and Ryan Baker and is available at http://www.cs.cmu.edu/~rsbaker/curvefit.tar.gz

2.1 Addressing local maxima

The first problem, that of local maxima, is an important one to address. The parameters associated with the knowledge tracing model control how it updates its estimates on the basis of student performance. Obviously, we would like estimates that provide the most accurate reflection of the student's knowledge. Since we are unable to observe the student's knowledge directly, we instead settle for preferring parameter estimates that enable the student model to predict student performance.

This study compares two approaches for estimating knowledge tracing parameters. The first approach, curve fitting, uses conjugate gradient descent to find best fitting parameter estimates, and has been used in prior studies with the cognitive tutors. The second approach makes use of the equivalence between knowledge tracing and a simple dynamic Bayesian network (DBN) with two nodes [9]. By casting knowledge tracing as a simple DBN, it is possible to use Expectation Maximization (EM) to perform the parameter estimation. We have constructed a tool, the Bayesian Net Toolkit for Student Modeling (BNT-SM) that takes as input a file of observations and an XML file describing the network structure and outputs the best-fitting parameter estimates [5]. This toolkit is built on top of the Bayes Net Toolkit [10].

Our testbed for this study is data from a 1997 study using the Cognitive Geometry Tutor [11]. These data consist of 24 students with 4101 opportunities to practice the 15 geometry skills that make up the domain. We use these data with each toolkit's implementation of knowledge tracing to obtain best-fitting parameter estimates. It is important to note that in both cases we are using the exact same data and exact same statistical model form. The only aspect that varies is the optimization technique to fit the parameters.

After training each model on the data to obtain the knowledge tracing parameter estimates, we then used them to trace student knowledge on the basis of the performance data. For every student action in the data set, the model compared its estimate of the student's knowledge with his performance. To compare the two modeling approaches, we used the AUC (Area Under Curve) metric for the ROC curve described by the estimated student knowledge and observed performance. The curve fitting approach had an AUC of 0.652 ± 0.01 while the Bayes Net Toolkit had an AUC of 0.696 ± 0.01. Thus, EM appears to provide a somewhat, and statistically reliably, more predictive set of parameters for this study. In a prior study comparing curve fitting with EM using data from Project LISTEN's Reading Tutor, for curve fitting we had an AUC of 0.568, while EM provided a reliably better ($p<0.01$) predictions with an AUC of 0.61. Therefore, this new experimental result provides converging evidence that EM outperforms curve fitting in different domains.

2.2 Addressing multiple global maxima

Multiple global maxima mean there is more than one combination of parameters that minimizes the amount of error. At first consideration, having multiple global maxima does not seem like much of a problem. If there are several best-fitting solutions in the parameter space, that should make the model fitting task easier since there is more than one "right" answer. To understand why multiple global maxima are a problem, consider the three sets of hypothetical knowledge tracing parameters shown in Table 1. The *knowledge* model reflects a set of model parameters where students rarely guess, the *guess* model assumes that 30% of correct responses are due to randomness. This limit of 30% is the maximum allowed in the Cognitive Tutors [12]. The third model is similar to those used by Project Listen's Reading Tutor [13] for performing knowledge tracing on speech input [14]. This model's *guess* parameter is very high because of inaccuracies in the speech recognition signal.

We compute the learning and performance curves for each model by using the knowledge tracing equations and the parameter values from Table 1. Specifically, we initialize P(know) to be K0. After each practice opportunity, we update P(know) = P(know) + (1 – P(know)) * T. For example, at the second practice opportunity the *knowledge* model would have a P(know) of 0.56 + (1 – 0.56) * 0.1 = 0.604. To compute P(correct), we use the knowledge tracing formula to combine the estimated knowledge with the slip and guess parameters: P(correct) = P(know) * (1-slip) + (1 – P(know)) * guess. As seen in Figure 2, the three models have identical student performance[2]—somewhat surprising given that the models appear so different. The much larger surprise is that in spite of having identical performance, their estimates of student knowledge (right graph in Figure 2) are very different.

[2] Technically the three curves are not perfectly identical, however they are equivalent under finite precision arithmetic.

Given the same set of performance data, we have presented three knowledge tracing models that fit the data equally well. Even if the *guess* parameter is capped at 0.3, there are still two competing models that perform equally well. It is not feasible to attack this problem statistically, as all three solutions are best fitting models. If all we have available is the performance data of students attempting to solve a skill, even given arbitrarily large quantities of such data, we are stuck with no way to decide which model we should prefer.

Table 1. Parameters for three hypothetical knowledge tracing models

Parameter	Model		
	Knowledge	Guess	Reading Tutor
K0	0.56	0.36	0.01
T	0.1	0.1	0.1
Guess	0.00	0.30	0.53
Slip	0.05	0.05	0.05

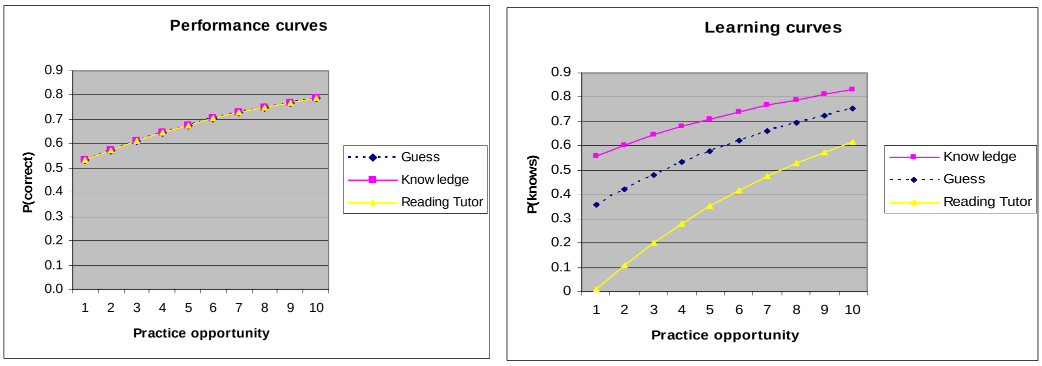


Figure 2. Three hypothetical knowledge tracing models illustrating the identifiability problem

Conceptually, model fitting is finding a set of model parameters, mp, that maximize P(data | mp). That is, the goal is to find a set of model parameters that make the data observed the least surprising. The problem with multiple global maxima is that there are several possible values of mp that maximize this likelihood. One possible workaround is to consider whether some parameter values are more likely than others. For example, few skills are mastered after only one practice opportunity. Therefore, given a choice between two competing models, we would prefer the one that required more than one practice opportunity to master a skill. Mathematically, we instead try to maximize P(data | mp) * P(mp). However, that leaves open the task of specifying the function P(mp).

Bayesian networks provide a natural mechanism, Dirichlet priors [15], for simultaneously maximizing the likelihood of the data and of the parameters. Dirichlet distributions are specified by a pair of numbers (α,β). Figure 3 shows an example (the dashed line) of the Dirichlet distribution for (9,6). This distribution is the one we used for prior experiments in biasing the model fitting process for Project LISTEN's Reading Tutor [16] to help guide the estimation of the K0 parameters. This distribution suggests that few skills have particularly high or low knowledge, and we expect students to have a moderate probability of mastering most skills. Conceptually, one can think of the conditional probability table of the graphical model being seeded with 9 instances of the student knowing the skill initially and 6 instances of him not. If there is substantial training data, the parameter estimation procedure is willing to move away from an estimate of 0.6. If there are few observations, the priors dominate the process. The distribution has a mean of $\alpha/(\alpha+\beta)$. Note that if both α and β increase, as in the solid curve in Figure 3, the mean of the distribution is unchanged but the variance is reduced. If you flip a coin and get 5 heads and 5 tails, you have moderate belief that P(heads) is around 0.5. If you flip the coin 100 times and get 50 heads and 50 tails, your average belief hasn't changed, but you are much less willing to believe that you have observed a biased sample and P(heads) is really 0.7. Similarly, Dirichlets enable researchers to not only specify the most likely value for a parameter but the confidence in the estimate.

We have explained a strategy, optimizing P(D|mp) * P(mp), for determining which model to prefer when several fit the data equally well, and have outlined a computational approach using Dirichlet priors. The problem is now how to specify the Dirichlet distribution. There are several sources of power we can use. One example is expert knowledge of the domain. If someone knows how quickly students tend to master a skill or the likelihood of knowing a skill, that knowledge can be used to set the priors. One complaint is that such an approach is not necessarily replicable as different people will use different experts.

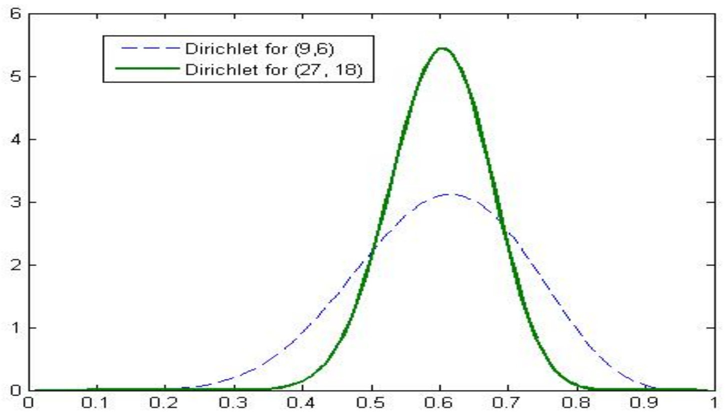


Figure 3. Sample Dirichlet distributions demonstrating decreasing variance

This paper proposes an alternate strategy for determining Dirichlet priors to guide model fitting. If all you have is the performance data for an individual skill, it is not possible to choose which best-fitting set of parameters to use. However, what if you have performance data for multiple skills? Consider a hypothetical example where 14 skills in the domain are estimated to have a guess parameter in the range of 0.1 to 0.2. Imagine if for the 15th skill, there are two equally good models, one has a guess parameter of 0.15 and the other has a guess parameter of 0.5. One heuristic is to prefer models that have parameters that are more similar to the other skills in the domain and therefore select the set of parameter estimates that have a guess value of 0.15.

Based on the idea that it is better to have skills with similar model parameters, we use the following approach to estimate the Dirichlet priors:

1. Use the BNT-SM to estimate the model parameters for the skills in the domain
2. For all four parameters (guess, slip, K0, t)
   - Compute the mean (μ) and variance (σ^2) of the parameter estimates
   - Select α and β to generate a Dirichlet with the same mean and variance as the estimates Specifically, solve for α and β such that:
     - $\mu = \alpha/(\alpha+\beta)$ (mean of the Dirichlet distribution)
     - $\sigma^2 = \alpha\beta / ((\alpha+\beta)^2 (\alpha+\beta+1))$ (variance of the Dirichlet distribution)
3. We now have one Dirichlet distribution described by (α,β) for each of the four parameters
4. Reestimate the value of all four model parameters using their associated Dirichlet priors

This procedure provides a bias for all of the parameters to move towards the mean. If most parameter values are tightly clustered except for a few outliers, the variance will be low, and α and β will be higher. The result of a higher α and β is to provide more weight for moving estimates towards the mean. If the distribution is naturally spread out, α and β will be lower, and there will be little tendency towards correcting extreme parameter estimates. The assumption in step #2 is that a good set of priors for what the parameter estimate should look like is how the parameter is actually distributed as a result of the model fitting process. Consider a skill whose guess parameter, as computed by step #1, is very high. After constructing the Dirichlet distribution for guess, in step #4 there will be a bias against assigning such a high value for guess since it is unlikely. Thus the value will be adjusted downwards towards the mean. The amount of bias towards the mean is proportional to how large α and β are, which is inversely related to the population variance. That is, if the population has a high variance then there is a small bias. Conversely, if a parameter value is tightly clustered, there will be a strong bias towards the mean.

To determine whether our approach of providing Dirichlet priors is effective, we used the same geometry testbed described previously. We split our dataset into training and testing sets, ensuring that we split the data at the level of users (so no user was in both the training and testing set). Even though the no priors and Dirichlet priors models are of equal complexity, it is necessary to split the data into training and testing sets. The process of using Dirichlet priors biases the parameter fitting process away from modeling the training data. After step #1, the parameters are in a local (or perhaps) global maxima. The purpose of step #4 is to move the parameters away from that location to be closer to the average. We should not be surprised if this process causes us to fit the training data less well. Hopefully it should result in improved test set performance.

Using the same evaluation metric as before, AUC, we find that on the training set the no priors approach (step #1) achieves an AUC of 0.707 while the Dirichlet approach achieves 0.705, a slight but not statistically reliable decrease. On the testing data, the Dirichlet approach did slightly but not reliably better with an AUC of 0.692 vs. 0.687 for the no priors approach. Prior experiments using Dirichlet priors resulted in statistically reliable improvements [16]. It is unclear whether we failed to achieve that threshold because of the small data set or because our approach of generating Dirichlet priors from data was not as effective as using human knowledge.

One question is whether it makes sense to iterate the above procedure. That is, once you have estimated the parameters using Dirichlet priors in step #4, loop back to step #2 and take the mean and variance of that distribution, compute α and β, and reestimate. We don't have a theoretic or conceptual model of what would happen, but empirically, model fit statistics dropped on successive iterations. Thus, there seems no point in applying the procedure repeatedly.

3 Evaluating the parameters

In addition to using predictive accuracy of the model, it is also possible to evaluate the parameters of the model directly. However, the question of what constitutes good parameter values is a tricky one. For prior work with the Reading Tutor, we were able to relate the learning parameters to known facts about the domain [16]. For example, the K0 parameter, initial knowledge, should be correlated with a word's frequency in text. Unfortunately, such an approach is difficult to generalize across domains since students are not typically exposed to domain skills in everyday life. For example, how would one count how many times a student would be expected to have an opportunity to learn Newton's Second Law?

However, it is possible to discover trends by visually inspecting the parameter estimates. Table 2 shows the parameter estimates for five random skills in the geometry curriculum for each of the three optimization approaches. Curve fit refers to the classic conjugate gradient descent approach; BNT-SM is the result of fitting a knowledge tracing model with the Bayes Net Toolkit for Student Modeling, and Dirichlets is using the BNT-SM with the four-step procedure described in the previous section. One item is quickly apparent: the curve fitting approach seems to prefer extreme parameter values with it selecting 0.0 (the minimum) or 1.0 (the maximum) for seven parameter estimates. In comparison, the two BNT-SM approaches combined have only one parameter estimate outside the range of [0.05, 0.95]. If we believe that very few skills are known by no one (or everyone), or can be mastered after only 1 exposure, such extreme values should give us pause.

A complementary approach to inspecting the parameter values directly is to see what they imply for the number of trials required to master each skill in the domain. We define mastery the same way as was done for the mastery learning criterion in the LISP tutor [4]: students have mastered a skill if their estimated knowledge is greater than 0.95. We compute the expected number of trials to master a skill by applying the P(know) formula, used to generate the left graph in Figure 2, to each model's K0 and t parameters. We calculated how many practice opportunities would be required until the predicted value of P(know) exceeds 0.95. These values are shown in Table 3. In addition to comparing the three knowledge tracing models, we also compare the parameter estimates generated by LFA (Learning Factors Analysis [7, 17]) for this data set [18]. Focusing first on the knowledge tracing models, it is interesting to note that the BNT estimates tend to believe less practice is needed than the curve fit estimates. Similarly, using Dirichlet priors also reduces the predicted number of trials until mastery.

Which set of numbers to believe is secondary. That the numbers differ so markedly is key. The difference in predictive accuracy between the models is not that great, particularly between the two BNT models. However, the pedagogical implications are great. Some tutors cannot use pure mastery learning and instead advance the entire class at the same rate. Such an approach is needed in many cases to be accepted in actual classrooms where having students spread throughout the curriculum would be unacceptable. When designing the tutor, how much practice should designers assume students will need? Curve fit suggests 160 problems suffice, BNT-SM 136, and BNT-SM with Dirichlet priors says 92. There is nearly a factor of two difference between the high and low estimates. Keep in mind that in many ways this study is a best case scenario for getting similar estimates: the same researcher is conducting all of the analyses, the exact same set of data is being used to train the models, and the exact same statistical model is being trained. The only aspect that varies is the optimization software. Furthermore, relative to problems such as student emotional state, engagement, learning style, etc., the problem of modeling the student's knowledge of a skill is well studied and understood. The moral is that researchers should not be so quick to accept model fit as an arbiter between competing models. There can still be large differences between models that are not easy to resolve quantitatively.

Table 2. Parameter values for 5 random geometry skills, and summary statistics for all 15 skills

Skill	model	K0	guess	slip	T
circle-area	curve fit	0	0.53	0.06	0.15
	BNT-SM	0.22	0.45	0.09	0.18
	Dirichlets	0.25	0.42	0.10	0.23
Circle-radius	curve fit	1	0.68	0.32	0
	BNT-SM	0.83	0.09	0.27	0.22
	Dirichlets	0.67	0.27	0.25	0.36
parallelogram-area	curve fit	0.56	0.99	0.07	1
	BNT-SM	0.90	0.93	0.06	0.49
	Dirichlets	0.81	0.71	0.06	0.69
pentagon-side	curve fit	0	0.44	0	0.1
	BNT-SM	0.12	0.41	0.03	0.11
	Dirichlets	0.15	0.39	0.06	0.14
trapezoid-height	curve fit	0	0.26	0.03	0.24
	BNT-SM	0.17	0.14	0.22	0.35
	Dirichlets	0.18	0.15	0.21	0.39
Mean (variance) for all 15 skills	curve fit	0.2 (0.11)	0.5 (0.09)	0.12 (0.01)	0.33 (.09)
	BNT-SM	0.41 (0.1)	0.43 (0.08)	0.14 (0.01)	0.31 (0.04)
	Dirichlets	0.4 (0.07)	0.41 (0.04)	0.15 (0.005)	0.45 (0.03)

Table 3. Number of predicted trials to master geometry skills

Skill	Curvefit	BNT-SM	BNT-SM+Dirichlets	LFA
circle-area	18	14	11	21
Circle-circumference	16	7	5	16
Circle-diameter	16	9	3	20
circle-radius	0	6	5	32
compose-by-add	21	17	2	n/a
compose-by-mult	6	5	5	13
parallelogram-area	1	2	2	n/a
parallelogram-side	11	1	2	4
pentagon-area	8	5	5	10
pentagon-side	28	25	20	21
trapezoid-area	9	11	8	13
trapezoid-base	4	8	7	15
trapezoid-height	10	7	6	15
triangle-area	11	13	8	61
triangle-side	1	7	3	8
Total practice to master curriculum	160	136	92	249+?

As an example of relaxing our best case scenario of the same data and statistical model, we also include results for LFA. LFA is a slightly different than knowledge tracing, and does not include performance parameters to account for randomness in student behavior. We used the LFA parameters for this data set generated by [18] and again computed the expected number of trials to master a skill. There were two skills that LFA did not believe students could master due to having negative learning rates[3], so we do not know how much practice it thinks students require for the entire curriculum, only that it is at least 249 practice opportunities. This amount of practice is nearly triple that required by the BNT-SM model that uses Dirichlet

[3] It is possible to constrain the LFA search to require non-negative learning rates to avoid this problem.

priors. To be clear, we are not asserting that this difference implies the Dirichlet model is "three times better" than LFA—for all we know it could be three times worse!

In the absence of a controlled field study, it is difficult to think of a way to resolve which model better models student growth. It may be instructive to look at skills where the various approaches diverge. For example, the skill circle-radius is believed to be known by students at the start in the curvefit model, require 5 or 6 practices in the BNT models, and 32 practices in the LFA model. Perhaps focused testing in areas of maximum divergence among the modeling approaches would be a way to resolve the issue?

One possible approach that is both data-driven and domain independent is to find general guidelines for the number of practice opportunities that are required to master a skill. For reading, a domain expert we collaborate with, Dr. Rollanda O'Connor, said that between 5 and 20 practice opportunities should suffice for learning a word. Although the exact numbers 5 and 20 might not be quite right for other domains, perhaps there are general guidelines that apply to many domains. It is likely that there are few skills that require only 1 or 2 practice opportunities to master, similarly numbers beyond 30 are also implausible.

4 Contributions, Future work, and Conclusions

This paper makes contributions in two areas: methodology for constructing student models and methods for evaluating competing models.

From a methodology standpoint, this paper replicates prior work with using Dirichlet priors [16] to bias student model parameter estimation. It extends our prior results by applying Dirichlet priors to a new domain, geometry. This domain is very different from reading both in the number of skills and in not relying on noisy speech recognition to score student performance. Furthermore, the approach presented here does not rely on beliefs about the true value of the parameters, and instead estimates the parameters to instantiate the Dirichlet distribution directly from the data.

This paper also extends methods for evaluating student models, going beyond the commonly used metric of predictive accuracy (e.g. [19, 20]). We have shown that looking at the model parameters themselves, as well as derived measures such as the expected number of trials to master a skill, are both useful supplements. We do not have a quantitative, domain-independent approach for evaluating these supplementary approaches, but have shown that models with comparable predictive accuracy can differ considerably on those measures. Furthermore, these measures are closer to what we care about, both as system designers and as educational data miners.

Although this paper provides new model construction and evaluation approaches, it raises more issues about how little we know. First, the model parameters estimated by EM appear more sensible than conjugate gradient descent. However, we do not know why. Is there some inductive bias between the two approaches that causes them to prefer different parts of the parameter space? Second, although Dirichlets appear a promising way of biasing the model fitting process, the problem of how to best select the α and β parameters is still unsolved. We have provided a data-driven domain-independent approach that seems plausible, but we have no theoretic reason to believe it is the best we can do. Our approach of using Dirichlets is somewhat restricted by only trying to model the domain. Another way of thinking of the priors is a way of biasing the parameter search. Any source of information that one wishes to use, statistics about the parameter distributions, personal beliefs about the domain, selecting a bias that better corresponds to external test scores, etc. are all permissible. In this paper, for purposes of generality, we restrict ourselves to objective, domain-independent sources of information.

Determining a method for evaluating model parameters directly is also a clear need. Are there recurring standard distributions of parameter estimates across different ITS and domains that we should expect to see? This question is a hard one to address, as typically researchers do not publish their model parameters. Perhaps with additional sharing of parameters, or at least summary statistics, some general trends can be derived.

An examination of the parameters underlying knowledge tracing suggests some room for improvement in how different types of interactions are handled. For example, the four model parameters are defined for each skill regardless of how they are presented. If, for a particular skill, some questions the tutor asks are multiple choice while others are free response, one would expect the slip and guess parameters would be different. However, standard knowledge tracing assumes a fixed slip and guess across question types. Similarly, if certain types of interaction are more conducive to learning than others, the probability of learning the skill (*T* parameter) should vary. For example, within the domain of reading we have shown that additional practice opportunities for a word on the same day are much less effective than the first practice opportunity [21]. Acknowledging these differences would be a sensible extension to knowledge tracing. As an example of such an extension, we have used the BNT-SM to discover that the receipt of help influences the probability a student acquires a skill [22].

In conclusion, researchers interested in student modeling should consider using the BNT-SM, available at www.educationaldatamining.org under "Resources." It appears to produce parameter estimates with better predictive accuracy than known alternatives, and a manual inspection of the parameter estimates suggests they are more plausible. Furthermore, the BNT-SM is flexible across types of graphical models it can accommodate. Knowledge tracing is a simple example, but the toolkit is capable of handling more complex models (e.g. [22]). Finally, the toolkit is not restricted to just modeling student knowledge estimates; it is capable of representing other latent variables.

The other broad conclusion is that knowledge estimation is a basic building block of both ITS construction and EDM. For ITS, accurate estimates of the student knowledge can be used for mastery learning [4], to determine whether to provide declarative instruction, and to determine on which problem-solving step help should be provided [23]. For EDM, student modeling is useful to measure the effects of interventions at a fine grain size [22] and serve as a feature to predict affective states about the student such as gaming behavior [24]. Furthermore, estimating student knowledge can be thought of as a base case for inferring other latent traits about the student. Although far from solved, understanding how to map student actions to knowledge estimates is better understood than many tasks. So potential difficulties uncovered for student modeling should apply as strongly to other EDM applications.

Acknowledgements

We acknowledge the help of Hao Cen helping us get set up with the geometry tutor data set, and for graciously sharing his model parameter estimates with us. This work was supported by the National Science Foundation, ITR/IERI Grant No. REC-0326153. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Carr, B. and I. Goldstein, *Overlays: a Theory of Modeling for Computer-aided Instruction,*. 1977, MIT: Technical Report, AI Lab Memo 406.
2. Corbett, A. and J. Anderson, *Knowledge tracing: Modeling the acquisition of procedural knowledge*. User modeling and user-adapted interaction, 1995. **4**: p. 253-278.
3. Beck, J.E. and J. Sison, *Using knowledge tracing in a noisy environment to measure student reading proficiencies*. International Journal of Artificial Intelligence in Education (Special Issue "Best of ITS 2004"), 2006. **16**: p. 129-143.
4. Corbett, A.T. *Cognitive computer tutors: Solving the two-sigma problem*. in *International Conference on User Modeling*. 2001. p. 137-147.
5. Chang, K.-m., J. Beck, J. Mostow, and A. Corbett. *A Bayes Net Toolkit for Student Modeling in Intelligent Tutoring Systems*. in *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*. 2006. p. 104-113 Jhongli, Taiwan.
6. Barnes, T., J. Stamper, and T. Madhyastha. *Comparative Analysis of Concept Derivation Using the Q-matrix Method and Facets*. in *Workshop at Educational Data Mining at AAAI2006*. 2006. p.
7. Cen, H., K. Koedinger, and B. Junker. *Automating Cognitive Model Improvement by A*Search and Logistic Regression*. in *Educational data mining workshop at National Conference on Artificial Intelligence*. 2005. p.
8. Winters, T., C. Shelton, T. Payne, and G. Mei. *Topic Extraction from Item-Level Grades*. in *Educational Data Mining workshop at AAAI2005*. 2005. p.
9. Reye, J., *Student Modelling based on Belief Networks*. International Journal of Artificial Intelligence in Education, 2004. **14**: p. 1-33.
10. Murphy, K., *Brief Introduction to Graphical Models and Bayesian Networks*. 1998: http://www.cs.ubc.ca/murphyk/Bayes/bnintro.html.

11. Koedinger, K.R. and J.R. Anderson, *Reifying implicit planning in geometry: Guidelines for model-based intelligent tutoring system design*, in *Computers as cognitive tools*, S.P. Lajoie and S.J. Derry, Editors. 1993, Erlbaum: Hillsdale, NJ. p. 15-45.
12. Anderson, J.R., A.T. Corbett, K.R. Koedinger, and R. Pelletier, *Cognitive tutors:Lessons learned.* The Journal of the Learning Sciences, 1995. **4**: p. 167-207.
13. Mostow, J. and G. Aist, *Evaluating tutors that listen: An overview of Project LISTEN*, in *Smart Machines in Education*, K. Forbus and P. Feltovich, Editors. 2001, MIT/AAAI Press: Menlo Park, CA. p. 169-234.
14. Beck, J.E. and J. Sison, *Using knowledge tracing in a noisy environment to measure student reading proficiencies.* International Journal of Artificial Intelligence in Education, 2006. **16**: p. 129-143.
15. Heckerman, D., *A Tutorial on Learning With Bayesian Networks*. 1995, Microsoft Research Technical Report (MSR-TR-95-06).
16. Beck, J.E. and K.-m. Chang. *Identifiability: A Fundamental Problem of Student Modeling*. in *International Conference on User Modeling*. 2007. p. Corfu, Greece.
17. Cen, H., K. Koedinger, and B. Junker. *Learning Factors Analysis - A General Method for Cognitive Model Evaluation and Improvement.* in *Intelligent Tutoring Systems*. 2006. p. 164-175 Jhongli, Taiwan: Springer.
18. Cen, H., K. Koedinger, and B. Junker. *Is More Practice Necessary? - Improving Learning Efficiency with the Cognitive Tutor through Educational Data Mining*. in *International Conference on Artificial Intelligence in Education*. 2007. p.
19. Beck, J.E., P. Jia, J. Sison, and J. Mostow. *Predicting student help-request behavior in an intelligent tutor for reading*. in *Ninth International Conference on User Modeling*. 2003. p. 303-312 Johnstown, PA.
20. Corbett, A.T. and A. Bhatnagar. *Student Modeling in the ACT Programming Tutor: Adjusting a Procedural Learning Model With Declarative Knowledge*. in *Sixth International Conference on User Modeling*. 1997. p.: Springer.
21. Beck, J. *Using learning decomposition to analyze student fluency development*. in *ITS2006 Educational Data Mining Workshop*. 2006. p. Jhongli, Taiwan.
22. Chang, K.-m., J.E. Beck, J. Mostow, and A. Corbett. *Does Help Help? A Bayes Net Approach to Modeling Tutor Interventions*. in *AAAI2006 Workshop on Educational Data Mining*. 2006. p. Boston, MA.
23. Gertner, A. and K. VanLehn. *Andes: A Coached Problem Solving Environment for Physics*. in *International Conference on Intelligent Tutoring Systems*. 2000. p. 133-142.
24. Baker, R.S., A.T. Corbett, K.R. Koedinger, and A.Z. Wagner. *Off-Task Behavior in the Cognitive Tutor Classroom: When Students "Game The System."* in *ACM CHI*. 2004. p. 383-390.

What's in a Word? Extending Learning Factors Analysis to Model Reading Transfer

James M. LESZCZENSKI and Joseph E. BECK
School of Computer Science
Carnegie Mellon University
Project LISTEN, Newell Simon Hall, 5000 Forbes Avenue Pittsburgh, PA 15213

Abstract. Learning Factors Analysis (LFA) has been proposed as a generic solution to evaluate and compare cognitive models of learning [1]. By performing a heuristic search over a space of statistical models, the researcher may evaluate different cognitive representations of a set of skills. We introduce a scalable application of this framework in the context of transfer in reading and demonstrate it upon Reading Tutor data. Using an assumption of a word-level model of learning as a baseline, we apply LFA to determine whether a representation with fewer word independencies will produce a better fit for student learning data. Specifically, we show that representing some groups of words as their common root leads to a better fitting model of student knowledge, indicating that this representation offers more information than merely viewing words as independent, atomic skills. In addition, we demonstrate an approximation to LFA which allows it to scale tractably to large datasets. We find that using a word root-based model of learning leads to an improved model fit, suggesting students make use of this information in their representation of words. Additionally, we present evidence based on both model fit and learning rate relationships that low proficiency students tend to exhibit a lesser degree of transfer through the word root representation than higher proficiency students.

1. Introduction

The task of modelling student cognition is at best an attempt to approximate how students mentally represent a given task. On one hand, the researcher can never know exactly how a student internally represents a problem, since often the student might not be exactly aware either. Approaches such as think aloud protocols have been proposed to capture such representations, but are generally too expensive and time-consuming to use in anything but limited studies [2]. However, a reasonably simple assumption can be made in this respect: if we build two models and one of them is a better fit to natural student learning data, we can say that it is a closer approximation to the student's mental representation. The challenge lies in specifying a good model. Even for simple, well-defined tasks, the obvious representation may not always be the right one. Symbolization in word problems, for example, has led to some counter-intuitive results concerning how easy certain mental representations are for students [3]. In fact, prior research has provided evidence that the choice of a model and its basic knowledge components is nontrivial. Determining the best choice of meaningful knowledge components, however, remains beyond the scope of this paper [4].

Learning Factors Analysis (LFA) has been proposed as a generic solution to evaluate and compare many potential cognitive models of learning [1]. As input, LFA takes an initial model of student representation and a set of legal operators to transition from one model to another. By performing a heuristic search over statistical models, the researcher may evaluate different cognitive representations of a set of skills. This framework offers the capability to compare different representations quantitatively, in terms of the statistical model chosen. Through such manipulation, we can check many hypotheses about what skills have the most in common. The advantage to this approach is that it bases all decisions about which model is best on just the data and the transition operators. As was shown in [1], LFA can locate skills that contain such commonalities.

From the perspective of a student's representation, the research task is to approximate how transfer occurs. Given an accurate model of a student's representation, such transfer would be easily extrapolated. To do so, the commonalities between skills (and thus the degree of transfer) would be implicit. The LFA framework attempts to model these relationships between skills under the assumption that if two skills A and B are better modeled by a single skill, then practice on either A or B transfers to the other. This implies that transfer is symmetric within an equivalence class of similar skills. The goal of the search is to identify these equivalence classes.

Above all, LFA has a clear advantage in that the underlying approach is domain-independent. The statistical model has no special knowledge about any particular field; it merely requires that the domain is defined by some set of skills, or knowledge components, which are then manipulated strictly based upon the empirical data. Thus, any advances in the performance, heuristics, or even the statistical model itself can be directly applied to all such domains.

One domain with perennial significance in the field of education is reading, where knowledge transfer has a clearly defined meaning. If a student reads a word, are there other words in the language at which we believe he may have become more proficient? One extreme approach would be the "bag of words" model, where every word is entirely independent. Reading a word in this representation will have no bearing on any other. However, this approach seems lacking, or else teaching students to read would simply consist of memorization of a significant subset of the English language, word by word. At the other extreme, one might suggest that every pair of words has some dependence. This implies that every time a student reads a word, he becomes (marginally) better at every other word in the language. This alternative approach also seems counterintuitive, as the authors feel that a student, both before and after reading the words 'dog' and 'cat' for the first time, will fail equally quickly on a word like 'supercalifragilisticexpialidocious'. However, we could reasonably assume that the student would perform more accurately on 'dog**s**' and 'cat**s**', relative to a similar student with no practice on any of these words.

In this paper, we describe a scalable application of the LFA framework in the context of knowledge transfer in reading. In this paper, we refer to 'words' and 'skills' interchangeably, as words are the initial building blocks of our cognitive models. Our intent is to develop a model that can identify students' representations of reading from data, as opposed to a theory-based assumption that students are all the same. We scale an existing educational data mining method to a level feasible for extensive reading data, and observe the resulting representations.

Given this architecture for reading analysis, what's in a word? Reading is certainly a task, and a task can theoretically be separated into knowledge components. Identifying these components is the complication, which an LFA-based approach can decide based solely on the data. This paper explores how this framework can extend to a corpus of reading data and makes observations about the implications of the resulting models with respect to students' mental representations.

2. Application to Reading Tutor

In order to provide a corpus of reading data for the LFA framework, we selected data from the Project LISTEN Reading Tutor in the 2003-2004 school year. The Reading Tutor [5] is a type of intelligent tutoring system (ITS), which assists elementary school kids in learning how to read. Using speech recognition technology, the Tutor listens to and records student utterances with which it compiles a database of information relating to student performance [6]. Additionally, it attempts to help students based on its recognition of these utterances.

Users of the Tutor were elementary school students in grades 1-6, mainly from the Pittsburgh, PA area. Paper tests were given both at the beginning and end of the year to assess individual reading level and improvement over the course of the school year. In 2003-2004 alone, the Reading Tutor collected approximately 6.9 million attempts by 650 elementary school students to read words. It is worth noting that the correctness of an attempt is defined as whether the automated speech recognizer (ASR) decided whether the student spoke the correct word. Analyses show that ASR correctly flags around one quarter of misread words, with a false positive rate of around 4% [7]. Among the additional data collected by the tutor are a timestamp to maintain a temporal ordering among a student's interactions with the Tutor, information about help asked for by the student, and latencies between utterances of words.

In an attempt to minimize the noise in our results, we chose to pre-process the database and screen out instances that seemed to poorly represent the trends present in the data. To this end, we first chose to only consider instances where a word was being read by a student for the first time in a day. This decision is supported by the observations in [8], where it was noted that "in general, massed practice is not helpful to learning." Additionally, we chose to screen out a list of 36 common stop words to avoid placing a great deal of weight on words that were already mastered by most students, and for which it would likely be difficult to discern a learning curve. To reduce the dimensionality of the data and avoid problems with sparsely estimated parameters, we screened out all attempts where there were not at least five students who had encountered the word at least five times (subject to the above constraints). Finally, since little gain in word decoding skill is to be expected after the word is read many times, we only considered the first 50 exposures to each word by a student. This screening process resulted in a set of 651,301 attempts by 469 students to read 1011 distinct words.

The advantages to using the Reading Tutor as a data source are numerous. Above all it is ecologically valid, in the sense that all data collected are from a natural learning environment, unhindered by confounding effects that might be present in a laboratory setting. The autonomous nature of data collection avoids both grader bias and inconsistency. Additionally, it allows for longitudinal, fine-grained, and comprehensive data to be collected with minimal effort, as opposed to a more typical and controlled psychological study where data generally must be aggregated by hand. The obvious trade-off is the inaccuracy associated with ASR, but this is quickly outweighed by the ease with which 6.9 million natural data points can be collected.

3. Background Theory and General Approach

Learning Factors Analysis (LFA) has been proposed as an interface by which to "combine statistics, human expertise and combinatorial search to evaluate and improve a cognitive model" [1]. This framework represents students' learning in terms of a logistic model, and performs a heuristic search over the space of all such models to locate the best of them. A discussion of model scoring heuristics will be saved for later in the section. The models themselves are each based upon a set of data including the students, skills performed, and the outcome of each trial. Models are transformed via 'split', 'add', and 'merge' operators which act on a single skill at a time, making incremental transitions between similar models. In essence, the operators are re-labelling the instances of data to form a new dataset, derived from the original. Over these potential states we perform a best-first search, where we further explore the states which seem to be the most beneficial to us. It is worth noting that while we direct our search in this fashion, countless other methods could be chosen to direct the heuristic search, such as the association rules described in [9].

Before describing a representation or approach, though, it is important to have a direction in which to apply this model. For the purposes of this paper, we decided to focus our efforts on analyzing a word root-based hypothesis of transfer. In terms of the original LFA operations, words are themselves atomic skills. We allow merges of words on only those skills which have the same word root, according to the Porter stemming algorithm [10]. For example, consider Figure 1, which depicts merging 'cat' and 'cats'.

Student	Skill	Skill Opportunities
A	cat	1
A	dog	1
A	cats	1
A	cat	2
B	platypus	1
B	cats	1
B	cat	1

Student	Skill	Skill Opportunities
A	cat*cats	1
A	dog	1
A	cat*cats	2
A	cat*cats	3
B	platypus	1
B	cat*cats	1
B	cat*cats	2

Figure 1. Left: Original Data, Right: Data after Merging 'cat' and 'cats'

Note that 'Skill Opportunities' denotes the number of times that that particular student has seen the given word at that point in time, as the entries are sorted by student, then by timestamp. From the learning curves estimated from each of these tables, a better fit on the second model would suggest that there is evidence of transfer between 'cat' and 'cats'. We can make this observation because modeling both of them as the same word made our model of student performance, and presumably our model of the students' representations, better. It is worth noting, from an implementation standpoint, that the data occasionally had instances where the Reading Tutor read the word to the student before he had a chance to read it. Since it was in fact an opportunity for the student to see the word, we made the decision to count it towards the Skill Opportunities. However, we declined to consider it a success or failure, since such instances were not a true test of the student's performance.

Our goal is to determine whether we can achieve a reliable improvement in the cognitive model by performing some subset of the allowable merges. If so, then we would have evidence of a correlation between the word root representation of words, and the student representation. The philosophical implication of the final set of skills chosen is that this model best represents how a student represents the task at hand. If the behavior of the student is better fit by the final model than before the operators were applied, we assume this is consistent with having discovered a better approximation to their mental representation.

4. Model and Implementation

Now that we have defined a dataset and a goal, the next crucial step is to decide how to represent it in terms of the logistic model in LFA. The original logistic model [1] is as follows:

$$\ln\left(\frac{p}{1-p}\right) = \sum \alpha_i X_i + \sum \beta_j X_j + \sum \gamma_j Y_j T_{ijt}$$

Equation 1

p = the probability to get an item right
X = the covariates for students
Y = the covariates for skills
T = the covariates for the number of opportunities practiced on the skills
Y T = the covariates for interaction between skill and the number of practice opportunities for that skill
α = the coefficient for each student, i.e. the student's "smarts"
β = the coefficient for each rule, i.e. the skill's difficulty
γ = the coefficient for the interaction between a skill and its opportunities, i.e. the learning rate

Note that there is one parameter for each student, as well as two parameters for each skill. For the dataset listed above this model would result in well over 1,000 parameters, which is bordering on infeasible already, before accounting for the vast number of instances to be handled. While we could theoretically compute model fit statistics for 650,000 instances of input as in the original LFA code, time and memory restrictions quickly inhibit the performance of even the most advanced of statistical software packages. While initial attempts were made by the authors to render this problem tractable, it became apparent that developing an alternative approximate solution might yield reasonable results. The first aspect of the model we focused on was the number of student parameters. These student variables serve as indicator variables in the original regression, identifying exactly which instances belong to each student. Rather than treating student identity as a separate factor, which would result in a number of parameters equal to the number of students, we instead computed the proportion of words accepted as correct in our dataset, for each student. This value is then used as a covariate in our model, requiring only a single parameter to represent an arbitrary number of students. While this simplification loses the representational power that the original model had regarding the students, it makes for a much more computationally feasible regression.

Having reduced the number of parameters significantly from the student perspective, we then focused on developing an approximation for the other part of the model; namely, those parameters relating to the skills. The immediate problem at hand is that of tractability; LFA recalculates the entire model after each application of an operator. As mentioned earlier, we are focusing on analyzing the word root model of decoding representation, which means that each transition involves a merge of a pair of related words. However, this focus implies that (now that the specific student parameters are gone) much of the data has no impact on a merge. For example, in the merge detailed in Figure 1, only those instances with 'cat' or 'cats' would need to be considered in the regression. A considerable savings in both time and memory required follows from the observation that only a small portion of the data will involve the two words to be merged. These simplifications also allow us to consider all merges to be independent, excepting merges with the same word root. From an implementation standpoint, proper caching of each independent merge in the best-first search tree can lead to greatly improved run times.

These smaller models introduce an element of approximation, especially due to the student smarts parameter, as seen in Equation 1. However, the skill coefficients obtained through logistic regression for each model can still be interpreted similarly to the original LFA ones, as can the model fit. Additional empirical validation came from a comparison of the decisions made by our regression model on the Geometry Cognitive Tutor data used in [1], where we saw similar trends to the published results.

The last topic related to defining our approach to LFA is to actually decide on some heuristic by which to score an individual model, in order to allow comparison between models. Numerous scoring metrics have already been proposed for LFA, including the Akaike Information Criterion (AIC), the Bayesian Information Criterion (BIC), log likelihood of the data, as well as the coefficient of determination (R-squared).

An in-depth discussion of the pros and cons of each of the criteria, as well as their definitions, is beyond the scope of this paper, for which the authors refer you to [11]. However, for the purposes of this paper we chose to direct our searches using AIC [12]. Since we are working with a large dataset, we wanted to minimize the effect of numerous instances upon the score, unlike BIC, which tries to factor in this effect. Additionally, the coefficient of determination was ruled out due to the lack of agreement by experts as to its usefulness in the logistic model [13]. A 'good' merge (i.e. one that should be allowed) is defined to be one where the AIC score of the model improved when the words are merged.

5. Experiments and Results

Our goal for this framework is to observe what sort of relationship exists between the word root model and student mental representations. The first question we must ask is: "Does the word root model actually provide us with a better model of the mental representation we are trying to examine?" In order to answer this question we examined each pair of skills in our model, to determine how many benefit from merging, and to what extent. The results can be seen in Figure 2. Points above the 0 line indicate that there was a positive benefit from performing the merge on the corresponding skill. We immediately recognize that there seem to be some skills that benefit greatly from merging, whereas others actually made the model score worse. One notable fact from this curve is that approximately 113 out of 177 of the skills would be merged (64%), if we were to choose the best-fit model for the population. Since each pair of skills is independent, we can also compute the net effect of these merges. Overall, there was an improvement of 581 points to the AIC score for the entire model, as compared to the total AIC score of 93,697, summed over the skills which benefit from merging.

In order to observe relative improvements to the students' representation due to the use of the word root model, we grouped students into subsets which we hypothesized would lead to different degrees of transfer. We felt that the word identification component of the Woodcock Reading Mastery Test [14], designed to assess a student's ability to read words of varying difficulty, would be a productive way to separate students. We divided students in two ways. First, we separated students into three equal-size groups based on their pre-test performance. Second, we created three equal-size groups based on the student gains from pre- to post-test. The first split reflects the impact of student knowledge on mental representation; the second split reflects whether the students' rates of learning and mental representations are intertwined. Perhaps students learn more quickly because they have a more efficient representation of the domain?

With these two divisions in mind, our purpose is to determine whether allowing merges of words with the same root would provide a better model fit. In order to measure this effect, we focus on two related experiments. Given the models for each pair of words and their resulting merge, we first examine the relationships between model score (AIC) and the skills being merged, over each of the divisions. These relationships provide us with two interesting pieces of information. First, within each of the three individual groups, the corresponding relationship provides some intuition as to what effect the merge has on the model representation itself. The perspective of such a result tells us whether there is any benefit to considering the word root model as a potential improvement to students' mental representations. Second, the relationships between each of the three curves provide information concerning the students themselves. For instance, we can ask ourselves "Who do we think is more likely to be using a word root representation of reading: a student who performs poorly on the pre-test, or one who did well?"

The second of our experiments involves looking directly at the learning rates for each of the skills. In LFA, the coefficients involving the learning rates and difficulty for skills are crucial for understanding how students represent words. In our pair-wise model, we produce similar parameters, and would like the ability to make statements concerning their interpretation. To this end, we look directly at the slope parameters for each pair of skills before and after merging, and attempt to discern whether there is any noticeable relationship between their initial values and the merged one. We can then examine this relationship to make claims about the transfer effect between words with the same root.

5.1. Results for Experiment 1: Model Scoring Comparisons

As previously described, we measured the model score value (AIC) for each pair of words with the same word root. For each pair, LFA resulted with two values: one score for the pair prior to merging, and one after having performed the merge. Figures 2 and 3 demonstrate the relationships between the three groups in each division, where the dependent variable on the y-axis is the improvement in model score for each pair of words merged.

For the pre-test divisions shown in Figure 2, we note that there seems to be a reliable difference between the high performing students and the low performing ones. Namely, each group has some words which, upon merging, improve the model score. However, the high performance students have the greatest proportion of these. From another perspective, there also seems to be an opposite trend with the words which do poorly when merged. The degree to which merging hurts the score is much less on the high proficiency students. In contrast, there is no such discernable pattern in the student gain graph in Figure 3. To better visualize how the groups compared, Table 1 contains the percentage of words that resulting in an improvement in AIC.

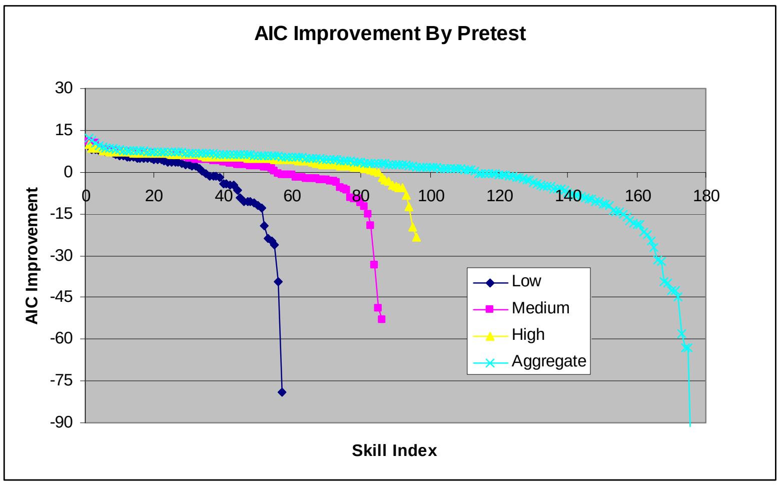


Figure 2. Impact of Merging for Pre-test (with overall aggregate)

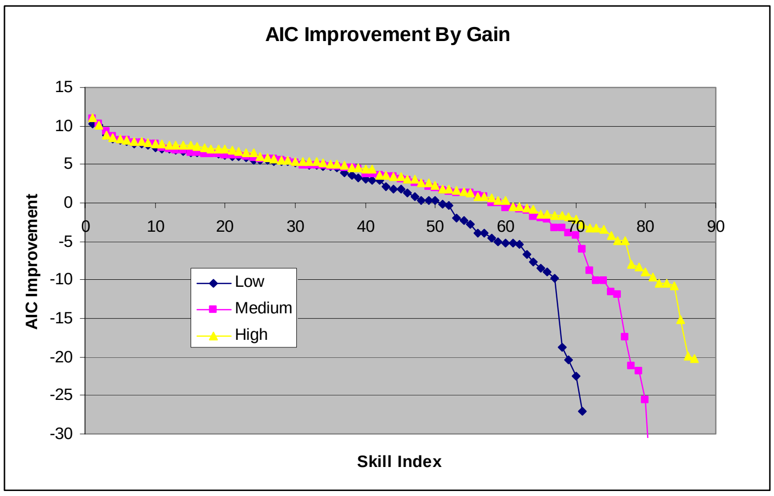


Figure 3. Impact of Merging for Gain

Upon performing a χ^2 test of reliability on the results in Table 1, we found that the only two reliable relationships present in the graph were in the pre-test group between the high proficiency students and each of the other two groups. High proficiency students showed signs of transfer for 89% of the words, while low and medium proficiency groups only showed signs of transfer for 59% and 64%, respectively. In order to consider these results at a finer grain, we performed additional reliability testing in the form of statistical bootstrapping. Bootstrapping is a resampling approach made popular by Efron in 1979 [15]. Among the approaches several desirable traits is a complete independence from assumptions concerning the distribution of the population, which is unknown for our application. In this case, our population is the set of differences in AIC score for each pair of student divisions. We bootstrapped 20,000 examples, from which we determined that there was only a reliable difference (with 95% confidence) between the high proficiency pre-test students and the other two pre-test groups. The resulting p-values are summarized in Table 2. This reaffirms the initial results from the χ^2 test.

From these observations we can conclude that there exists a reliable difference between these student divisions, using pre-test scores as an indicator. Specifically, high proficiency students exhibit more transfer at the word root level. However, using student gains over the year is not useful as a predictor.

Table 1. Percent of possible merges that improve AIC score

Student Group	Test Score Measure	
	Pretest	Gains
Low	59%	70%
Medium	64%	71%
High	89%	69%

Table 2. P-values for bootstrapping comparisons between divisions

Student Group	Test Score Measure	
	Pretest	Gains
Low vs. Medium	.18	.52
Medium vs. High	.01	.28
Low vs. High	<.002	.68

5.2) Results for Experiment 2: Learning Rate Relationships

An alternate way of exploring the models developed by LFA is to focus on the learning rate for each of the skills. Specifically, prior to merging two skills such as 'cat' and 'cats', the regression model has two different parameters, corresponding to the last summation in Equation 1, which represent the learning rates for each skill. We can similarly analyze the learning rate obtained from the merged skill. We take the average of the initial learning rates for each pair of skills and regress the learning rate of the final merged skill against it. In theory, the more positive the resulting slope is, the more transfer is occurring. More importantly, we can compare these values within both sets of divisions, to determine what types of students seem to get the most transfer from the word root model. In order to maintain our goal of computing interpretable results, we decided to remove any data points which had values greater than five times the value of any other data points for a particular regression. This definition of outlier resulted in the removal of one data point from the high proficiency pre-test division, and two data points from the high proficiency gains division. Our results are summarized in Table 3. In general, this ratio represents the fact that if a skill has very little associated transfer, we can expect that the merged skill will have a lower learning rate than if the word had exhibited a high degree of transfer.

Table 3. Degree of transfer from words to word roots

Student Group	Test Score Measure	
	Pretest	Gains
Low	.30	.28
Medium	.26	.43
High	.40	.41

Table 4. P-values for bootstrapping comparisions between regression slopes

Student Group	Test Score Measure	
	Pretest	Gains
Low vs. Medium	.78	.11
Medium vs. High	.17	.97
Low vs. High	.55	.27

For the pre-test divisions, we first note that the degree of transfer doesn't necessarily increase steadily as our estimate of initial student knowledge increases. However, the high proficiency pre-test students seem to have the strongest positive relationship, which implies that we observed the greatest degree of transfer from their data.

When examining the gains divisions, we have the opposite problem with the middle group, as they seem to be exhibiting approximately as much transfer as the high gain students. Nevertheless, it appears as if the low gains students are exhibiting the least degree of transfer, which supports our overall conjecture that stronger students will have a stronger degree of word root transfer than weaker students. We note that these results paint a slightly different picture than the previous experiment, which could make more substantial claims about the pre-test results while providing little information about the student gains. However, one consistent result is that high proficiency students exhibit the greatest degree of transfer.

Similarly to our previous experiment, we performed statistical bootstrapping to determine to what degree these results are reliable. Our resulting p-values, summarized in Table 4, indicate that we cannot reliably depend on the noted trends, perhaps due to a great deal of variation in the slope data.

It is worth noting that these overall results are generally harmonious with those in [16] on the same dataset, especially with respect to higher proficiency students demonstrating a greater degree of transfer between words with the same root. It remains a matter of contention concerning the students in the medium proficiency group, since no results seem to indicate anything with certainty. However, all results seem to indicate a difference between the degrees of transfer in low and high proficiency students.

6. Contributions, Future Work, and Conclusions

In this paper, we have introduced the LFA framework to a new domain: student modelling in reading. Our pair-wise approximation to the original model for LFA allows for tractable computation on datasets orders of magnitude greater than those used with the original code base[1]. We have used these new capabilities of LFA to demonstrate that the use of word root representation improves our model of student knowledge. Furthermore, the experiments in this paper imply that there is a difference in students' mental representations based on two alternate metrics: first, their initial knowledge at the beginning of the school year, and second, their improvement over the course of the year. We have found a positive correlation between pre-test scores and the proportion of word merges that are predicted to improve the fit of our model. Additionally, we have explored the relationships between learning rates in each division of students. From these relationships we demonstrated that students who perform well on the pre-test seem to exhibit a greater degree of transfer than the other groups. On the other hand, students who are deemed to have gained little over the course of the year (unreliably) exhibit a rather low degree of transfer. Furthermore, we have introduced the application of two metrics for evaluating the models in LFA; namely, by examining the percentage of merges that occur, as well as examining the relationship between learning rates to approximate the amount of transfer.

While our model has provided useful preliminary results, many directions of exploration still remain. To begin with, an analysis or classification of which words were beneficial to merge, both within and across groups, could lead to interesting observations about transfer. Also, word root is only one of many possible linguistic models of word comparison. For instance, past studies have been done with student models involving grapheme to phoneme mappings [17]. Other potential models include examining the onset or rime of each word. Even these ideas only scratch the surface of possible models of reading representation; it is much more intuitive that each student's representation is comprised of an amalgamation of these models. This will also lead to more complex applications where the factors are potentially dependent on each other, making the composition of operations a nontrivial problem.

From an implementation standpoint, there exist a number of optimizations that could be considered for future iterations. Among these include an improvement to the underlying model search, where each state is represented not by an additional copy of the data, but by a set of operators performed to reach the state. This optimization would drastically improve memory management for LFA when used on large datasets, and would require a relatively small amount of computation to apply all of the operators to generate the data when necessary, even if the A* search were very deep. Also, preliminary research was done into locating a software package that could handle a larger dataset using the full LFA model [18]. While performing the entire search using this model might still be intractable, it seems reasonable to believe that a hybrid approach might be taken which combines the faster approximation presented here, and the extensive evaluation offered by full LFA.

This research represents a significant step forward in developing the Learning Factors Analysis framework. By demonstrating that the approach can be approximated in the domain of reading, we have developed new intuitions concerning student mental representations as well as the effects of student proficiency. Indeed, if dividing empirical data yields different models for each group of students, perhaps maintaining a single canonical domain model is not necessarily an appropriate practice for intelligent tutoring systems. These results impact the educational data mining community by showing that LFA is a tool that can be feasibly modified to model extensive and complex corpora of data. In addition, it is capable of capturing the effects of transfer in reading, as demonstrated by the two experiments in this paper.

[1] Source code for our implementation of LFA is available online at http://www.educationaldatamining.org under "Resources."

Acknowledgements

This work was supported by the National Science Foundation, ITR/IERI Grant No. REC-0326153. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation. Additionally, we thank Hao Cen, Brian Junker, and Becky Kennedy for their help and advice in the development of this approach.

References

1. Cen, H., K. Koedinger, and B. Junker, *Learning Factors Analysis - A General Method for Cognitive Model Evaluation and Improvement.* Proceedings of Intelligent Tutoring Systems, 2006.
2. Newell, A. and H.A. Simon, *Human Problem Solving*. 1972, Englewood Cliffs, NJ: Prentice-Hall.
3. Heffernan, N.T. and K. Koedinger, *A developmental model for algebra symbolization: The results of a difficulty factors assessment.*, in *Proceedings of the Twentieth Annual Conference of the Cognitive Science Society*. 1998, Lawrence Erlbaum Associates, Inc: Mahweh, NJ. p. 484-489.
4. Croteau, E.A., N.T. Heffernan, and K. Koedinger. *Why Are Algebra Word Problems Difficult? Using Tutorial Log Files and the Power Law of Learning to Select the Best Fitting Cognitive Model.* in *7th International Conference on Intelligent Tutoring Systems*. 2004. p. 240-250: Springer Berlin.
5. Mostow, J. and G. Aist, *Evaluating tutors that listen: An overview of Project LISTEN*, in *Smart Machines in Education*, P. Feltovich, Editor. 2001, MIT/AAAI Press: Menlo Park, CA. p. 169-234.
6. Mostow, J., J. Beck, R. Chalasani, A. Cuneo, and P. Jia. *Viewing and Analyzing Multimodal Human-computer Tutorial Dialogue: A Database Approach*. in *Fourth IEEE International Conference on Multimodal Interfaces*. 2002. p. Pittsburgh, PA.
7. Banerjee, S., J. Mostow, J. Beck, and W. Tam. *Improving Language Models by Learning from Speech Recognition Errors in a Reading Tutor that Listens*. in *Second International Conference on Applied Artificial Intelligence*. 2003. p.
8. Beck, J. *Using learning decomposition to analyze student fluency development.* in *ITS 2006 Educational Data Mining Workshop*. 2006. p. Jhongli, Taiwan.
9. Freyberger, J., N.T. Heffernan, and C. Ruiz, *Using Association Rules to Guide a Search for Best Fitting Transfer Models of Student Learning*. 2004, Worcester Polytechnic Institute: Worcester, MA. p. 10.
10. van Rijsbergen, C.J., S.E. Robertson, and M.F. Porter, *New models in probabilistic information retrieval*. 1980, British Library: London.
11. Cen, H., K. Koedinger, and B. Junker, *Automating Cognitive Model Improvement by A* Search and Logistic Regression.* Proceedings of AAAI 2005 Educational Data Mining Workshop, 2005.
12. Akaike, H., *A new look at the statistical model identification.* IEEE Transactions on Automatic Control, 1974. **19**(6): p. 716-723.
13. Meynard, J., *Coefficients of determination for multiple logistic regression analysis.* American Statistician, 2000. **54**: p. 17-24.
14. Woodcock, R.W., *Woodcock Reading Mastery Tests- Revised (WRMT-R/NU).* 1998, Circle Pines, Minnesota: American Guidance Service.
15. Efron, B., *Bootstrap Methods: Another Look At The Jackknife.* The Annals of Statistics, 1979. **7**(1): p. 1-26.
16. Zhang, X., J. Mostow, and J. Beck, *All in the (word) family: Estimating transfer from reading similar words.* Educational Data Mining Workshop, (under review).
17. Chang, K., J. Beck, J. Mostow, and A.T. Corbett. *Using speech recognition to evaluate two student models for a reading tutor.* in *12th International Conference on Artificial Intelligence in Education*. 2005. p. 12-21 Amsterdam.
18. *SPLUS 7.0 Enterprise Developer*. 2005, Insightful Corp.: Seattle, WA.

Predicting Student Engagement in Intelligent Tutoring Systems Using Teacher Expert Knowledge

Nicholas M. LLOYD [a], Neil T. HEFFERNAN [a] and Carolina RUIZ [a]
[a] *Department of Computer Science, Worcester Polytechnic Institute, Worcester, MA USA*

Abstract. Detection and prevention of off-task student behavior in an Intelligent Tutoring System (ITS) has gained a significant amount of attention in recent years. Previous work in these areas have shown some success and improvement. However, the research has largely ignored the incorporation of the expert on student behavior in the classroom: the teacher. The main goal of this project was to predict student engagement, both positive behavior and negative (gaming) behavior, within the *Assistments* system using teacher observations of student behavior in the tutoring classroom. Using a dataset incorporating attributes associated with previous findings in gaming detection research, we developed two logistic regression models using stepwise regression to predict positive engagement as well as gaming behavior. Gaming detection models proved unsuccessful, however positive engagement prediction shows promising results with prediction accuracy at 71.1%. To our knowledge this is the first investigation that has shown that we can detect high levels of student engagement, thus paving the way for more accurate ways of providing positive feedback to students.

Introduction

The effectiveness of an Intelligent Tutoring System (ITS) can be undermined by students who are not engaged in the learning activity. Recent research into disengaged behavior, most notably gaming behavior where a student is exploiting the available help and feedback provided by an ITS, has shown that there is a correlation between such behavior and reduced learning [6]. Developed methods of detecting gaming have shown some success [4,5] along with studies directed towards classifying, measuring, and modeling a wider range of student engagement and disengagement as well as emotional states and attitudes with an ITS or other computer learning environment [2,11,8,9,10]. Detecting gaming behavior within the *Assistments* system, an Intelligent Tutoring System that has been developed jointly between Worcester Polytechnic Institute (WPI) and Carnegie Mellon University (CMU) [16], has proved to be less successful [19]. However, all of this research has largely ignored the expert of student behavior in the classroom: the teacher. Teachers have long been seen as the most knowledgeable of their students' behaviors in the classroom, and this has been acknowledged by some in the ITS community [9,18]. Teachers also have a direct influence on their students' engagement patterns within the classroom [17,12]. Additionally, work to determine why students game came to the conclusion that

Table 1. Student Focus Grades

1 = Very focused, model student
2 = Focused, appears to be working
3 = Unsure
4 = Unfocused, I don't think they are making an effort
5 = Very unfocused, they are messing around/not paying attention

students who are not good at math and get frustrated are typically the students who game [7]. However that study left open the question of why some students show very high persistence.

Following previous work on detecting gaming within the *Assistments* system [19], the goal of this research was to explore the use of teacher observations of student engagement within the *Assistments* system in developing predictive models of student engagement. Instead of asking the teachers to provide a simple rating of gaming or not-gaming, teachers were provided with a simple 5 value grading scheme as a code for identifying highly engaged students to moderately engaged students to highly disengaged/gaming students. This method was chosen in contrast to prior studies that focus on either gaming or not-gaming [5,4,19] in order to provide the possibility of detecting good behavior as well as bad behavior. . Positive feedback is important to student learning and detecting only the presence or absence of gaming behavior does not provide an effective means for rewarding and encouraging good engagement behavior.

Methods

Classroom Observation

Teachers being the primary source of data, it was important to be as unobtrusive as possible as well as to be as straightforward as possible in order to acquire data that accurately represents how a teacher typically monitors the behavior of their students during tutoring sessions. These issues required a different approach to the data collection process than has previously been employed [6,19].

At the beginning of every tutoring session with a new teacher, the observer was instructed to briefly explain the purpose of the study and what would be required of the teacher. The actual observations consisted of the observer shadowing the teacher for the span of the tutoring session as the teacher went about their way as they would during any tutoring session. As the teacher monitored the behavior of his/her students, they were instructed to grade the current activity of the students they were watching using a coding scheme for student engagement. Table 1 describes this coding scheme. Note that the terms "focus" and "engagement" may be used interchangeably throughout this article, however the meaning is equivalent in this domain[1].

As the teacher provided the grades for the different students, the observer would record the grade on a table associating the grade with the prerecorded student's user

[1]The term "focus" was chosen in contrast to "engagement" or "effort" in order to emphasize a more positive intention behind the teacher-given grades. Stating that a student is less focused is not as demoralizing as saying a student is putting forth less effort. This term posed no difficulty in interpretation for the teachers observed.

name as well as the minute in time of the observation. The data recording tables were constructed such that the starting hour of the period is recorded in an area at the top of the document and the column labels represent the minutes following this start period.

The observation period for this research spanned the month of March, 2007. During this period a total of 7 classes of approximately 20 students per class. Over all of these classes 3 different teachers were observed. In any given class a teacher typically yielded 2–3 observations per student over an hour long period with grade differences between observations rarely exceeding one. The final dataset yielded 265 teacher given grades for the analysis[2]. Workload for the observers was found to be approximately equivalent to that of previous studies [6,19].

Dataset Creation

The *Assistments* system maintains a record of a running dialog between the student and the tutor detailing every student action and tutor response[3]. The recorded observation times from the data collection phase of this research were used to determine what problem or problems the student was working on in the minute long time window when the observation was made. This tactic was chosen so as to characterize the relationship of engagement behavior with not only student behavior but student behavior with particular problems.

The data collected for the dataset focuses on several aspects of student activity within an ITS that have been identified as important in identifying and measuring student engagement. These aspects can be broken down into data relating to the student and data relating to the problem the student is currently working on. A student's prior knowledge of the given material is strongly correlated with their engagement patterns and, in particular, whether or not they will engage in gaming behavior [17,6,9,10,19]. However, performance alone is not an effective predictor of engagement since not all students who engage in gaming behavior are hurt by it, as demonstrated by high performing students who are among the gaming students [4]. Additionally, how quickly and in what ways a student interacts with the system are important predictors of student engagement [6,10,19]. As for problem related data, the measure of a problem's difficulty has been noted to correlate with engagement patterns [10,18] as well as the type of problem being presented, in other words multiple choice problem or short answer problem [6,19].

The dataset attributes are clustered into two distinct groups: 1) Observation period - indicating that they are directly related to the observation period, and 2) General - indicating that they are attributes that are not directly related to the observation period. Descriptions of these attributes are as follows:

Focus grade - The teacher-given grade for the student at a certain time, which will constitute the dependent variable for the analysis.

Observation period

First action - This value indicates what action the student performed after being shown the problem associated by time with the focus grade. This value consists of three possible values: 1) attempt - indicating that the student made an attempt to answer the problem 2) hint - indicating that the student requested a hint and 3) bottomHint -

[2]For more information regarding the observation process of this study see citation [13].

[3]At this time the system does not record as detailed information as mouse movement in the tutor environment.

indicating that the student requested a bottom out hint (in other words the answer to the problem). It is important to note that students do not know if the next hint they receive will be a bottom out hint; however, some students who are disengaged from the learning activity will purposefully seek bottom out hints. Additionally, although this is not common, some problems only contain bottom out hints, thus indicating why this is presented as a possible value for the first action variable. For the analysis these string values are encoded into the integer values 1, 2, and 3 respectively.

First action time - A millisecond value representing how long the student took to respond to the problem with the first action.

Second action - Similar to the first action except this value has the additional possibility of being empty if the first action performed was a correct answer. Empty values are encoded in the analysis as a 0 integer.

Second action time - A millisecond value representing how long the student took to make their second response to the problem, provided that they did not answer the problem correctly on the first attempt.

Student attempts this problem - A count of the number of attempts the student has made to answer this problem at this particular observation time.

Attempts this problem z-score - A standardized measure of the number of attempts the student has made on this problem based upon how students typically respond to this problem. The z-score[4] is calculated by taking the mean and standard deviation of the number of attempts made to answer this particular problem over all available data in the database. The mean is subtracted from the number of attempts the student has made on this problem at this time and the result is divided by the standard deviation.

Student hints this problem - A count of the number of hints requested by the student for this problem.

Hints this problem z-score - Same procedure as the “attempts this problem z-score” value except that the number of hints requested for the problem is under consideration.

Student bottom hint this problem - A “count” of the number of bottom hints requested by the student for this problem. This is a value of 0 or 1 since there is only ever one bottom out hint for any given problem.

General

Poor man’s prior knowledge - A measure of the student’s preceding performance. Although there is current research in the *Assistments* system that uses a student’s performance as related to skills associated with different problems at varying levels of granularity [15], this does not happen live and not all problems in the *Assistments* system are associated with distinct math skills. As a result, the student’s performance before each observation was estimated in a similar manner to the work by Walonoski and Heffernan [19], where the percent correct of the student’s previous work is calculated.

Problem type - It is important to note that a student’s interaction with the system is largely dependent upon the type of input that is required of the student to answer the problem. This input is broken into two separate categories: multiple-choice and short answer. An example of how a student will interact differently lies in the observation that a student may simply “guess and check” their way through a multiple choice problem since there are a limited number of answer options presented, whereas a short answer problem is less conducive to this behavior. In contrast to this a student presented with a short answer problem is arguably more likely to follow “help abuse” gaming patterns.

[4]Z-score is a statistical method of standardizing an observation value with respect to the properties of the population [http://en.wikipedia.org/wiki/Z-score].

Problem difficulty - This variable is a simple measure of a given problem's difficulty based upon all data available in the *Assistments* system database before the observation period. This data goes back to the year 2004. This value is a percentage of the number of times this problem was answered *incorrectly* in a problem set.

Analysis and Results

The first step in our analysis was to evaluate the distribution of focus grades in order to look for any potential bias from the teachers towards a particular focus grade. Figure 1 shows the overall distribution of grades in the dataset with the notably strong trend towards the more positive focus grades (see figure 1). The particular trend pattern displayed in this figure is equivalent for each of the three teachers with the key difference being a stronger trend towards more of the positive focus grades and less of the negative focus grades for the advanced level students [5]. This indicates that teachers tend to rate their students as being highly engaged, in this case 52.8% of the dataset, whereas negative behavior such as gaming constitutes only 6% of the data (focus grade 5). It should be noted that gaming behavior has been observed to be just as infrequent in previous studies of such activity in the *Assistments* system [19].

For the development of predictive models we chose stepwise binary logistic regression using the Likelihood Ratio Forward Selection procedure as available in the SPSS Statistical Software Suite which was used for this analysis[6]. Initial analyses of the data using linear and multinomial regression techniques provided disappointing results [13]. However, these previous analyses were focused on developing models for the entire range of focus grades rather than attempting to predict focus grades of particular importance. Following this, two binary logistic regression models were developed to predict the presence or absence of the two extremes of the focus grade range: highly engaged level 1 students and highly disengaged level 5 students. Considering the variety of categorical and

[5]These particular students were members of an advanced level mathematics curriculum. The students had to meet certain academic standards and formally apply to get into this particular class.

[6]The Likelihood Ratio forward selection method is a stepwise selection method where forward entry of variables into the model is based upon the significance of the score statistic while removal is tested using likelihood-ratio statistic and maximum partial likelihood estimates. For more information see the SPSS Logistic Regression documentation.

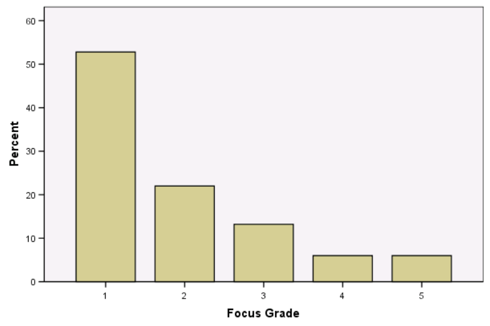


Figure 1. Distribution of Focus Grades in the Dataset

numeric data logistic regression was deemed most appropriate for this analysis. Following this two new dependent variables were added to the dataset, one representing “Great Effort” and one representing “Gamer.” Both variables are binary representations of the focus grades with “Great Effort” having a value of 1 if the associated focus grade is 1 and 0 if otherwise. The “Gamer” value is just the opposite with 1 representing a focus grade of 5 and 0 indicating all other focus grades.

The first model produced was to predict “Great Effort” for a student based upon their activity on a given problem. Table 2 shows the attributes determined to be statistically significant to the model along with their coefficients. The attributes, and their coefficients, determined to be most significant to the model were the “first action type” and “pmp” (Poor man’s prior knowledge) values. The “first action type” value was translated from its original form as being either an “attempt” or a “hint” to a value of either 1 or 0 respectively. Both have a positive correlation indicating that if the first action performed was an attempt and the student has a higher performance, then there is a greater likelihood that they are putting forth “Great Effort.”

Classification tests on the dataset, where the original dataset is classified by the model, were run to determine the accuracy of the predictions. Table 3 shows the test results for the “Great Effort” model. The overall percent correct shows that 71.1% of the time the predictions are correct. This is significantly better than simply guessing which, considering the quantity of focus grade 1, would have a 50% probability of success.

The second model produced was to predict whether or not a student was a “Gamer” based upon the teacher given grades. Table 4 shows the attribute and coefficient for the model. It is surprising that there is only one attribute and coefficient that have been selected for this model, “first action type,” however this does coincide with the “Great Effort” model. Additionally, the coefficient is almost the exact reverse of the coefficient for the “Great Effort” model for the same attribute. This makes sense considering that we are trying to predict what essentially is the polar opposite from the other model, that being gaming behavior.

Table 2. Selected Attributes for the “Great Effort” Model

Attribute	**Coefficient**	**S.E.**
Poor man’s prior knowledge	2.704	0.547
First action type (attempt)	1.932	0.429

Table 3. Great Effort Model Classification Test Results

Great Effort	**Percent Correct**
False	63.2%
True	78.0%
Overall	71.1%

Table 4. Selected Attribute for the “Gamer” Model

Attribute	**Coefficient**	**S.E.**
First action type (attempt)	-1.416	0.545

Table 5. Gamer Model Classification Test Results

Gamer	Percent Correct
False	100.0%
True	0.0%
Overall	94.0%

The results from the classification test on the "Gamer" model were disappointing. Table 5 shows that although the overall percent correct was a strong 94%, the low percentage of focus grade 5 rows in the dataset (6%) proved to be an insurmountable challenge for the model which was unable to accurately predict any of the focus grade 5 values correctly. The negative values for the constant and the coefficient in this model indicate that no matter what this model will predict that a student is not gaming.

Considering the significance of the poor man's prior knowledge attribute in predicting "Great Effort," it was important to re-analyze the data from the perspective of different student performance groups. The reasons for this are: 1) High performing students, though less likely to game, still do game [4] and 2) Identifying students who put forth great effort based in part on their performance does not account for students who may have low prior knowledge but are still putting forth a significant amount of effort. The dataset was subdivided into three groups based upon distinct ranges of the prior knowledge attribute. These ranges were: low performing students with prior knowledge <= 0.33, mid-range performing students with prior knowledge > 0.33 and <= 0.66, and high performing students with prior knowledge > 0.66. The distributions of these groups in the dataset are 29.2%, 41.2%, and 29.6%.

Figure 2 displays the percentage of correctly predicted Great Effort values using the Great Effort Model on each of the prior knowledge subgroups. The x-axis shows numeric labels for each subgroup with 0 representing the low performers, 1 representing the mid-range performers, and 2 representing the high performers. Though there is a notable increase in the accuracy of predicting great or not great effort in the high performance group, all groups still have a prediction accuracy level around 70%. The consistence of the prediction accuracy across these groups shows that the model does not exclusively consider all high performers to have excellent effort and at the same time low performers are not all identified as having poor effort.

Conclusion and Discussion

The results from the evaluations of the produced models indicate that the data collected on teacher grades can to a certain degree of accuracy predict whether or not a student is strongly engaged in the learning activity with the *Assistments* tutor. Although 71.1% is still not an ideal accuracy level, it provides a starting point for future work using teacher data to develop models of student engagement. In addition, since this work shows that there is potential for predicting positive engagement, there is by extension the potential for generating reward systems to go along with the gaming prevention systems that have been developed in past research [14,3,20]. However, the prediction accuracy across low, medium, and high performing students is consistent for the model suggesting that even though prior performance is a factor the model does not distinguish high performing students as being exlusively the positively engaged students. This is important considering that as much as low performing students are most likely to be frustrated and, therefore, to game [7], the performance level of a student does not preclude them from being persistent in their learning effort. What is important for a model of this nature is to identify when a student is putting forth great effort regardless of their prior performance in order to encourage the students who are doing well, as well as the students who are having the most difficulty with the subject material.

Unfortunately the prediction of gaming behavior by our model was ineffective. However, this is hardly surprising considering the low percentage of gaming instances in our dataset (6%). This difficulty has been noted in a previous study on gaming behavior in the *Assistments* system [19]. In that study the detection of non-gaming instances by their model was 98% accurate, however it is important to note that this does not indicate positive engagement behavior in the student. Evaluation of other statistical models of the data provided less effective results, however the correlation of teacher given grades with a graphical reporting tool on student engagement provided positive results [13]. Other regression models of the data splitting the focus grade groups in different ways, 1-2 vs 3-5 for instance, were ineffective.

Since 26.67% of students who were identified as gaming were mislabeled by the model as having positive engagement, this model is not accurate enough to be implemented in the *Assistments* system. However, the determined accuracy is still better than guessing and the prediction accuracy is consistent across all performance levels. This shows that it is possible to predict positive engagement in an Intelligent Tutoring System, and that teacher data can be used to develop these predictive models. While models designed to detect gaming behavior can institute corrective action, either by active [14] or passive [3,20] techniques[7], more accurate models predicting positive engagement would allow for a reward system for the student potentially bringing positive reinforcement of desired learning behaviors.

[7]Active gaming prevention techniques directly alter the learning environment in order to prevent gaming while passive techniques offer unobtrusive yet highly visible feedback on student behavior.

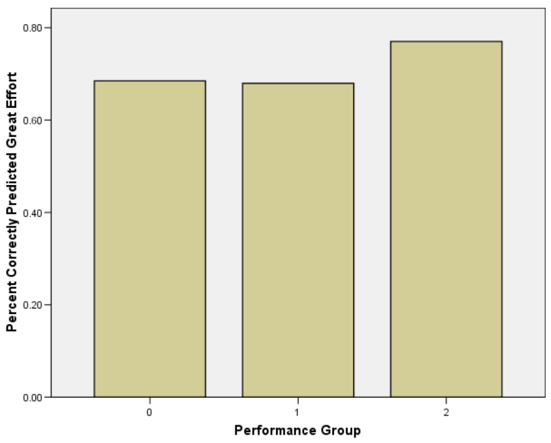


Figure 2. Great Effort Model Prediction Accuracy for Low (0), Mid-Range (1), and High (2) Performing Students

Future Work

The results in this study provide a benchmark for future research into the detection of positive engagement in an Intelligent Tutoring System as well as the incorporation of teacher expert knowledge in the development of such models. This study was instituted as an exploratory analysis in using teacher data in gaming and engagement detection. It is clear from these initial results that there is potential for models developed from teacher given data for predicting student positive engagement. More data from a larger group of teachers could provide better models for these predictions. Additionally, a larger attribute set incorporating a wider time window to evaluate not only the problem they were working on as in this study but also the subsequent problems and possibly an average between them could improve the results of this study.

While we would like to have messages be displayed to the student along the lines of “You seem to be really focused on the assignment! Good job!” models that more accurately predict when students are positively engaged are necessary. It is important to note that misidentification of students who are engaged as being not engaged is not as problematic as rewarding students who are not engaged, or even gaming, for having positive engagement. These false positives would have the opposite of the desired effect and could convince the gaming students that their actions will go undetected.

Another question that has yet to be answered is which source of data is better at predicting student behavior patterns: teachers or outside observers. This study relied on data exclusively from the teachers, however a comparison of separate models produced by the two data sources could provide a clue as to where better data can be gathered for the development of behavior detection models in Intelligent Tutoring Systems.

Acknowledgements

This research was made possible by a large number of people listed at www.ASSISTment.org. This research was also made possible by the US Dept of Education, Institute of Education Science, "Effective Mathematics Education Research" program grant #R305K03140, the Office of Naval Research grant #N00014-03-1-0221, NSF CAREER award (REC-0448319) to Neil Heffernan, and the Spencer Foundation. All of the opinions in this article are those of the authors, and not necessarily of the funders.

References

[1] V. Aleven and K.R. Koedinger, Limitations of student control: Do students know when they need help?, In *Proceedings of the 5th International Conference on Intelligent Tutoring Systems*, Montreal, Canada, (2000), 292–303.

[2] I. Arroyo, T. Murray, and B.P. Woolf, Inferring unobservable learning variables from studentsâĂŹ help seeking behavior, In *Proceedings of the Workshop on Analyzing Student-Tutor Interaction Logs to Improve Educational Outcomes, at the 7th International Conference on Intelligent Tutoring Systems*, MaceiÃş, Alagoas, Brazil, (2004), 29–38.

[3] R.S. Baker, A.T. Corbett, K.R. Koedinger, S. Evenson, I. Roll, A.Z. Wagner, M. Naim, J. Raspat, D.J. Baker, and J.E. Beck, Adapting to When Students Game an Intelligent Tutoring System, In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, Jhongli, Taiwan, (2006), 392–401.

[4] R.S. Baker, A.T. Corbett, and K.R. Koedinger, Detecting Student Misuse of Intelligent Tutoring Systems, In *Proceedings of the 7th International Conference on Intelligent Tutoring Systems*, MaceiÃş, Alagoas, Brazil, (2004), 531–540.

[5] R.S. Baker, A.T. Corbett, K.R. Koedinger, and I. Roll, Generalizing Detection of Gaming the System Across a Tutoring Curriculum, In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, Jhongli, Taiwan, (2006), 402–411.

[6] R.S. Baker, A.T. Corbett, K.R. Koedinger, and A.Z. Wagner, Off-Task Behavior in the Cognitive Tutor Classroom: When Students "Game the System", In *Proceedings of ACM CHI 2004: Computer-Human Interaction*, Vienna, Austria, (2004), 383–390.

[7] R.S. Baker, J.A. Walonoski, N.T. Heffernan, I. Roll, A.T. Corbett, and K.R. Koedinger (In press), Why Students Engage in "Gaming the System" Behavior in Interactive Learning Environments, *Journal of Interactive Learning Research (JILR)*, Chesapeake, VA: AACE.

[8] R.L. Bangert-Drowns and C. Pyke, A Taxonomy of Student Engagement with Educational Software: An Exploration of Literate Thinking with Electronic Text, *Journal of Educational Computing Research* **24**(3) (2001), 213–234.

[9] C.R. Beal, L. Qu, and H. Lee, Classifying Learner Engagement Through Integration of Multiple Data Sources, In *Proceedings of the 21st National Conference on Artificial Intelligence*, Menlo Park, California (2006), 2–8, AAAI Press.

[10] J.E. Beck, Engagement tracing: using response times to model student disengagement, In *Proceedings of the 12th International Conference on Artificial Intelligence in Education (AIED)*, Amsterdam, The Netherlands, (2005), 88–95.

[11] C. Conati, Probabilistic assessment of user's emotions in educational games, *Journal of Applied Artificial Intelligence* **16** (2002), 555–575.

[12] R.D. Goddard, W.K. Hoy, and A.W. Hoy, Collective Teacher Efficacy: Its Meaning, Measure, and Impact on Student Achievement, *American Educational Research Journal* **37**: 479–507, 2000.

[13] N.M. Lloyd, Measuring Student Engagement in an Intelligent Tutoring System, Worcester Polytechnic Institute, Technical Report WPI-CS-TR-07-04 (2007).

[14] R.C. Murray and K. VanLehn, Effects of Dissuading Unnecessary Help Requests While Providing Proactive Help, *Artificial Intelligence in Education* (2005), 887–889.

[15] Z.A. Pardos, N.T. Heffernan, B. Anderson, and C.L. Heffernan, Using Fine-Grained Skill Models to Fit Student Performance with Bayesian Networks, *Workshop in Educational Data Mining held at the 8th International Conference on Intelligent Tutoring Systems* (2006).

[16] L. Razzaq, M. Feng, G. Nuzzo-Jones, N.T. Heffernan, K. Koedinger, B. Junker, S. Ritter, A. Knight, E. Mercado, T.E. Turner, R. Upalekar, J.A. Walonoski, M.A. Macasek, C. Aniszczyk, S. Choksey, T. Livak, and K. Rasmussen, The Assistment Project: Blending Assessment and Assisting, In *Proceedings of the 12th International Conference on Artificial Intelligence in Education*, Amsterdam, The Netherlands, (2005), 555–562.

[17] J.C. Turner, C. Midgley, K.K. Meyer, M. Green, E.M. Anderman, Y. Kang, and H. Patrick, The Classroom Environment and Students' Reports of Avoidance Strategies in Mathematics: A Multimethod Study, *Journal of Educational Psychology* **94**: 88–106, 2002.

[18] A. de Vicente and H. Pain, Informing the Detection of the Students' Motivational State: An Empirical Study, In *Proceedings of the 6th International Conference on Intelligent Tutoring Systems*, Biarritz, France, (2002), 933-943.

[19] J.A. Walonoski and N.T. Heffernan, Detection and Analysis of Off-Task Gaming Behavior in Intelligent Tutoring Systems, In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, Jhongli, Taiwan, (2006), 382–391.

[20] J.A. Walonoski and N.T. Heffernan, Prevention of Off-Task Gaming Behavior in Intelligent Tutoring Systems, In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*, Jhongli, Taiwan, (2006), 722–724.

Analyzing Fine-Grained Skill Models Using Bayesian and Mixed Effects Methods

Zachary A. PARDOS, Mingyu FENG, Neil T. HEFFERNAN, Cristina LINDQUIST-HEFFERNAN & Carolina RUIZ
Worcester Polytechnic Institute
{zpardos, mfeng, nth, ruiz}@cs.wpi.edu

Abstract. Two modeling methods were employed to answer the same research question of how accurate the various grained models with 1, 5, 39 and 106 skills are at assessing student knowledge in the ASSISTment online tutoring system and predicting their performance on the 2005 state MCAS test. One method, used by the second author, is mixed-effects statistical modeling. The first author evaluated the problem with a Bayesian networks machine learning approach. We compare the two results to identify benefits and drawbacks of either method and to find out if the two results agree. We report that both methods showed compelling similarity in results especially with regard to residuals on the test. Our analysis of these residuals and our online skills allows us to better understand our model and conclude with recommendations for improving the tutoring system, as well as implications for state testing programs.

1. Introduction

Intelligent Tutoring Systems (ITS) rely on models that associate the various skills students are learning with different questions or actions. We created 3 different models with different grain sizes. One model has 5 skills, another 39 and our finest grain model has 106 skills. A model with a single skill was used to represent unidimentional assessment. We found that working with the 20 teachers that use our system, many appreciate the reports made possible by the fine grain sized models that tell them which specific skills a student is doing poorly on. But are these finer grained models at least as accurate as more traditional, coarse grained models [6] in prediction performance?

To our knowledge, no one else has specifically investigated this question. However some have investigated the results of skill hierarchies using simulated users [2, 3]. This paper attempts to investigate fine grain skill modeling using; Bayesian networks [10] popular in Artificial Intelligence research and mixed-effects modeling [12] popular in statistical analysis. We investigate if both modeling methodologies yield similar results to the question of "Are finer grain sized skill models more accurate at test prediction." We will be able to gain confidence in both types of modeling if one method corroborates the other's results.

2. The Massachusetts Comprehensive Assessment System (MCAS)

The MCAS is a Massachusetts state administered standardized test that produces tests for English, math, science and social studies for grades 3rd through 10th. We are focused on only 8th grade mathematics. Our work relates to the MCAS in two ways. First we have built our content based upon the ~300 publicly released items from previous MCAS math tests. Secondly, we will be evaluating our models by predicting the 8th grade 2005 MCAS test which was taken by students after the online data being used was collected.

3. Background on the ASSISTment Project

The ASSISTment system is an e-learning and e-assessing system [5]. In the 2004-2005 school year, 600+ students used the system about once every two weeks. Eight math teachers from two schools would bring their students to the computer lab, at which time students would be presented with randomly selected MCAS test items. Each tutoring item, which we call an ASSISTment, is based upon a publicly released MCAS item which we have added "tutoring", also known as "scaffolding", to. If students get the item correct they are advanced to the next question. If they answer incorrectly, they are provided with a small "tutoring" session where they are asked to answer a few questions that break the problem down into steps. The first scaffolding question appears only if the student gets the item wrong. We believe that the ASSISTment system has a better chance of showing the utility of fine-grained skill modeling due to the fact that we can ask scaffolding questions that break the problem down in to parts and attempt to identify which skills were to blame. Most MCAS questions that were presented as multiple-choice were converted into text-input questions on the tutor system to reduce the chance of guess. As a matter of logging, the student is only marked as getting the item correct if they answer the question correctly on the first attempt and do not ask for hints.

4. Creation of the Fine-Grained Skill Model

In April of 2005, we staged a 7 hour long "coding session" at Worcester Polytechnic Institute (WPI), where our subject-matter expert, Lindquist-Heffernan, with the assistance of the 3rd author, set out to make up skills and tag all of the existing 8th grade MCAS items with these skills. There were about 300 released test items for us to code. Because we wanted to be able to track learning between items, we wanted to come up with a number of skills that were somewhat fine-grained but not too fine-grained such that each item had a different skill. We therefore imposed upon our subject-matter expert that no one item would be tagged with more than 3 skills. She gave the skills names, but the real essence of a skill is what items it was tagged to. To create the coarse-grained models we used the fine-grained model to guide us. For the WPI-5 model we started off knowing that we would have the 5 categories; 1) Algebra, 2) Geometry, 3) Data Analysis & Probability, 4) Number Sense and 5) Measurement. Both the National Council of Teachers of Mathematics and the Massachusetts Department of Education use these broad classifications as well as a 39 skill classification. After our 600 students had taken the 2005 state test, the state released the items from test and we had our subject matter expert tag up those test items.

The WPI-1, WPI-5 and WPI-39 models are derived from the WPI-106 model by nesting a group of fine-grained skills into a single category. This mapping is an aggregate or "is a part of" type of hierarchy as opposed to a prerequisite hierarchy [1]. Figure 1 shows the hierarchal nature of the relationship between WPI-106, WPI39, WPI-5 and WPI-1.

WPI-106	WPI-39	WPI-5	WPI-1
Inequality-solving	setting-up-and-solving-equations	Patterns-Relations-Algebra	The skill of "math"
Equation-Solving			
Equation-concept			
Plot Graph	modeling-covariation		
X-Y-Graph	understanding-line-slope-concept		
Slope			

Figure 1. Skill model hierarchy sample

5. The Dataset

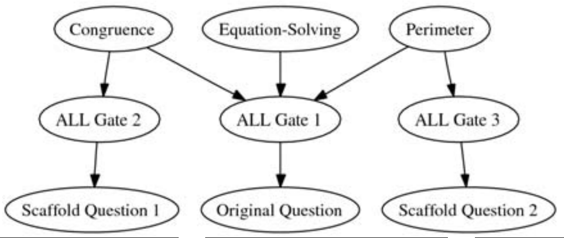
Both methods used the same data to evaluate users. Both methods used the same skill tagging as prescribed by the various grained skill models. The mixed-effects execution was memory limited and thus only a subset of 447, out of the 600 total users could be run. The Bayesian method used these same 447 users. This is the single point of difference between a previous Bayesian result [8] which ran 600 users. The current mixed-effects result also differs from a previous result [4] in the number of users run as well as the inclusion of the WPI-39 in this paper.

6. Bayesian Methodology

P(Congruence)
0.50

P(Equation-Solving)
0.50

P(Perimeter)
0.50



Gate	P(Question)
True	0.95
False	0.10

Gate	P(Question)
True	0.95
False	0.10

G	P(Question)
True	0.95
False	0.10

Figure 2. Bayesian Belief Network

The Bayesian topology for the various grain sized skill networks consist of skill nodes, 'ALL' nodes (which can be though of as 'AND' gates) and question nodes (Figure 2). Each question node represents an original item or scaffold in the ASSISTment system and has its own 'ALL' node which has mapped to it the skills associated with that question according to the skill model. The reason for the 'ALL' node is to simplify the parameters for each question to only a guess and slip value (no matter the number of skills tagged) which are set *ad hoc* to 0.10 and 0.05 respectively. These values are what might be expected for a text-entry question. The parameters have not been optimized or allowed to vary per question or per model. The 'ALL' node also imposes that all the parent skills must be known in order for the questions to be answered correctly. The background probability of knowing a given skill is set to 0.50.

The prediction processes is run for one user at a time. The user's data responses on the ASSISTment system are organized and entered into the Bayes net as evidence. The knowledge probabilities of the skills are then inferred. Now that we have predicted the user's skills we can predict the test items. This is done by entering the inferred skill probabilities as "soft evidence" into the MCAS test network. Soft evidence is probabilistic as opposed to observed evidence. Now that the MCAS test network has skill values, we can infer the likelihood that a given question on the test will be answered correct. If the probability of correct for a question is inferred to be 0.70, then 0.70 points are added to the total score. Once all questions have been evaluated, a total projected MCAS score is left as a sum of the predicted question probabilities. This sum is compared with the user's actual score for accuracy measures.

7. Mixed Effects method

For dichotomous (binary in our case) response data, several approaches adopting either a logistic or probit regression model and various methods for incorporating and estimating the influence of the random effects have been developed. Snijders & Bosker [13] provide a practical summary of the mixed-effects (fixed effect plus random effect) logistic regression model and various procedures for estimating its parameters. Hedeker & Gibbons [7] describes mixed-effects models for binary data that accommodate multiple random effects. As these sources indicate, the mixed-effects logistic regression model is a very popular and widely accepted choice for analysis of dichotomous data. It describes the relationship between a binary or dichotomous outcome and a set of explanatory variables. In this work, we adopted this model and fitted on our longitudinal, binary response data. When fitting the model, two sub-models will be simultaneously built, in which level-1 sub-model fits within-person change and describes how individuals change over time and level-2 sub-model tracks between-person change and describes how these changes vary across individuals. The 2-level representation of the model in terms of *logit* can be written as

Level-1 (or within-person) model: $\log[\frac{p_{ij}}{1-p_{ij}}] = b_{0i} + b_{1i} * TIME_{ij}$

Level-2 (or between-person) model: $b_{0i} = \beta_0 + v_{0i}$
$b_{1i} = \beta_1 + v_{1i}$

Where p_{ij} is the probability of a positive response for student *i* at time *j*.
b_{0i}, b_{1i} denote the two learning parameters for student *i*. b_{0i} represents the "intercept" and tells how good is the student's initial knowledge; b_{1i} represents the "slope" and tells what's the change (i.e. learning) rate of student i. β_0, β_1 are the fixed-effects and represent the "intercept" and "slope" of the whole population average change trajectory. v_{0i}, v_{1i} are the random effects and represent the student-specific variance from the population mean.

Such a model is often referred to as "longitudinal model" [12] since time is usually introduced as a predictor of the response variable, which allows us to investigate change over time. After the model was constructed, the fixed-effects for the whole group (i.e. β_0, β_1 in the above 2-level model) and the random effects for each student (i.e. v_{0i}, v_{1i}) were extracted and then the two learning parameters "intercept" and "slope" (i.e. b_{0i} and b_{1i} in the model above) was calculated for each individual student (and for each skill if skill was introduced as factor into the model). Given this, we thus can apply the model on the items in the state test to estimate students' response to each of them.

8. Comparing Test Results

For consistency, all models' logged results were arranged in standard format and evaluated by the same process to produce the average MAD and Error numbers in Table 1. The MAD score is the mean absolute difference which is the average difference between actual and predicted score across all users for a given model. MAD Error is the MAD score divided by the total number of questions on the test (MAD / 34). The under/over prediction is our predicted average score minus the actual average score on the test. The actual average score will be the same for all models. The centering is a result of offsetting every user's predicted score by the average under/over prediction amount for that model and recalculating MAD and error percentage. This table of results appears in a poster paper at the AIED07 main conference [9].

Table 1. Bayesian and Mixed Effects Test Prediction Results

Model		MAD Error	MAD Score	Under/Over Prediction	Error After Centering	Centered MAD Score
WPI-106	Bayes	13.75%	4.19	1.0	13.60%	4.10
	Mixed-effects	12.10%	4.12	0.6	12.04%	4.10
WPI-39	Bayes	12.05%	4.10	1.0	11.82%	4.02
	Mixed-effects	12.40%	4.22	1.2	12.07%	4.11
WPI-5	Bayes	18.70%	5.42	3.5	16.20%	4.70
	Mixed-effects	12.84%	4.37	1.8	12.13%	4.12
WPI-1	Bayes	25.46%	7.39	4.6	21.63%	6.27
	Mixed-effects	13.00%	4.42	2.1	12.06%	4.10
Best Bayes Error: 12.05% (WPI-39)				**Best Mixed-effects Error:** 12.10% (WPI-106)		

We can see that the mixed-effects models outperform all but the Bayesian WPI-39 which stands as the highest performing model. A sizeable decrease in accuracy can be observed between the mixed-effects WPI-5 and WPI-1 and also in the Bayesian WPI-5 and WPI-1. This result suggests that the current Bayesian prediction performs best with the finer grained models. The top performing models for both methods are the fine-grained WPI-39 and WPI-106. The relative performance of the models are the same between the two methods with the exception of the WPI-39 and WPI-106. The paired T-test values of 0.8282 (Bayesian) and 0.2146 (mixed-effects) for the WPI-106 vs. the WPI-39 explains that the difference between the two models is not statistically significant and thus the two models are susceptible to variability amongst each other. All other models compared to one another are statistically significantly different in both methods ($p < 0.05$).

The internal fit of the models has also been evaluated using Bayesian methods [8]. The result was that the number of skills in the model was proportionate to the accuracy of prediction. The WPI-106 did best at predicting the online student answers with 5.50% error while WPI-1 did the worst with 23.77% error. The methodology for predicting internal fit with the Bayes method is similar to the methodology described above for predicting the test. The difference in the training phase is that for each question predicted, the Bayes model was retrained using all the student data except the question being predicted. This was done for every question with every student. The error is the average percent difference between predicted total correct answers and actual correct answers. Internal fit was also evaluated with the Mixed-effects method. Though the evaluation methodology was different than the Bayes method, the relative ranking of model accuracy was the same: the more skills in the model, the better the internal fit. In addition, the result of the paired t-test showed that when the mixed-effects method was used, finer grain-sized models always made statistically significantly better prediction of students' responses to the online data.

9. Analysis of Residuals

To identify where the models struggle, we first look at the Bayesian test item residuals to find which questions have the lowest prediction accuracy. Figure 3 shows the Bayesian residuals for each model grouped by test question. Figure 4 shows the mixed-effects models' residuals. The test questions correspond to the question numbers of the 2005 MCAS 8th grade math test. Some numbers, such as 9, are skipped because they are short essay questions which we choose not to evaluate since there are no questions of that type in the tutor. A positive residual means that the item was underpredicted on average, a negative residual means the item was overpredicted. The residual is the average of the students' actual responses minus our predicted probabilities of correct for that question.

One observation that is immediately apparent in both the Bayes and mixed-effects models is that the residuals do not differ greatly per question from one skill model to the next. It is also apparent that the Bayes and mixed-effects residuals are quite similar. This similarity raises our confidence in the execution of the methodology used with the two models. There are a few differences that we can point out. The WPI-106 model has the largest positive residual spikes in the mixed-effects model despite being the best performer; however, the Bayesian WPI-106 does not have these spikes. Also, question number 24 is an item of contention between the two methods with all the mixed-effects models overpredicting and all the Bayesian models underpredicting answers to the question. However, the two figures show decideably similar traits best emphasized by questions 18 and 11 which are the worse predicted questions in both the Bayes and mixed-effects models. Question 18 is tagged with Venn-Diagram while question 11 is tagged with Point-Plotting. In the next section will investigate our online system to see if poor assessment of these skills is to blame.

10. Analysis of Online Skills: A case study of Venn-Diagram

A few items on the test stick out as being predicted very poorly. The reason for this could be a discrepancy between the knowledge displayed on the test for the skills relating to those questions and the assessed probability of knowledge for those skills after training on the online data. It could also be a poor tagging of skills to the questions.

We will perform a case study of the skill of Venn-Diagram tagged to item number 18 on the MCAS (Picture 1) which was the item with the highest average residual value among all the models. This item was consistently underpredicted by all models 40% of the time. Our system believes this question should be a lot harder than it was, to answer why we gather information about that skill on our system (shown in Table 2).

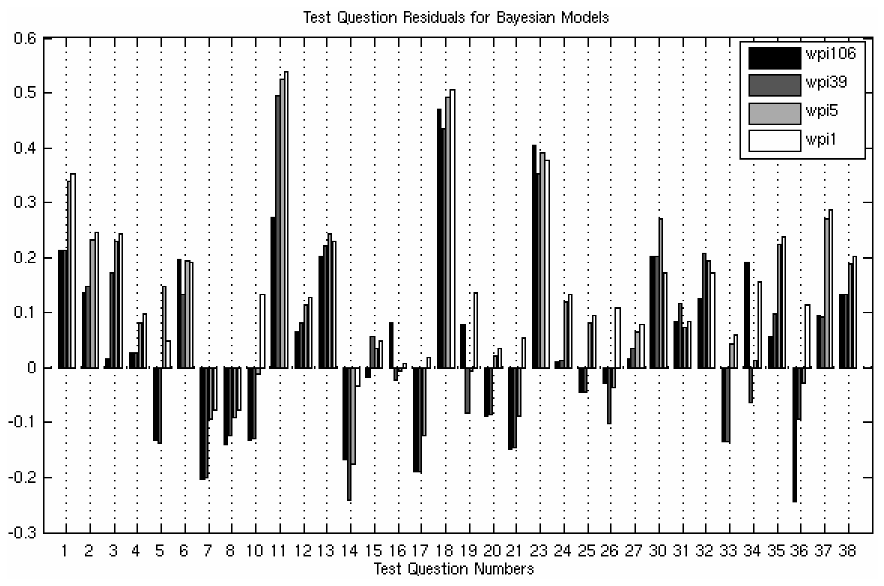


Figure 3. Bayesian residuals

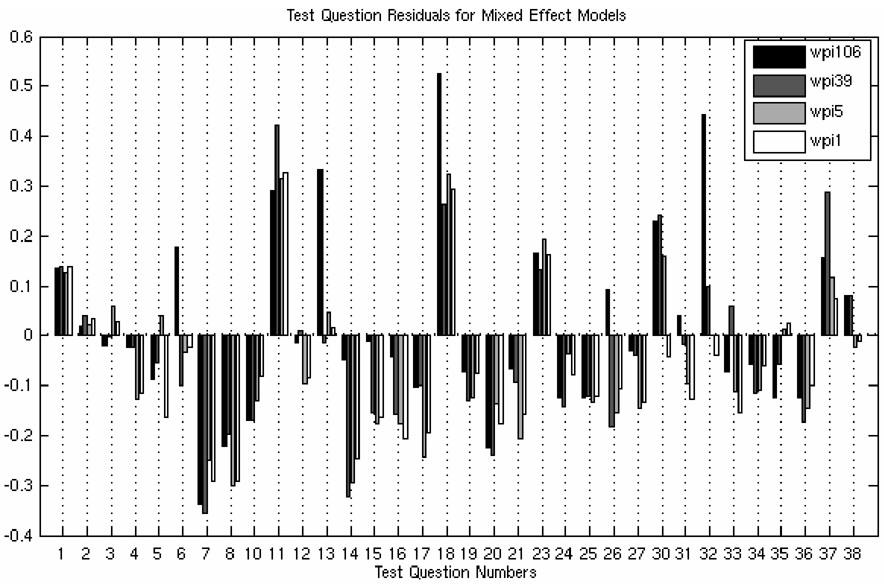


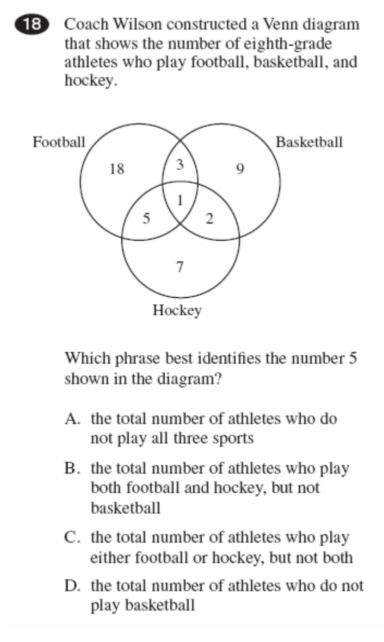
Figure 4. Mixed-effects residuals

Table 2. Online skill information for Venn-Diagram

Percent-Correct:	18.2%
Bayes Assesment Avg:	22.9%
Original Items:	2
Scaffold Items:	4
Data Points:	2,106

The percent correct of questions tagged with the skill Venn-Diagram is 18.2%. This is a dramatic difference from the percent correct of 87.3% on item 18 of the MCAS tagged with the same skill. This certainly suggests that the Venn-Diagram questions in our online system are much harder than the question that appears on the test.

A significant difference between the MCAS and online questions is that the online questions are text entry where as the question on the test is multiple-choice. This difference in answer type creates a dramatic disparity in difficulty of the problem. We take a quick look at common answers for this online item in order to emphasize the relative ease of multiple-choice questions on the MCAS test compared with their text entry counter parts in the ASSISTment system. The online problem was taken from the 2002 MCAS test (Item 30) and converted to an ASSISTment item by removing the multiple-choices and adding scaffolding (See Figure 6). There were ~1,200 responses to this question and the most common answer, with 455 responses, was the incorrect answer of 126 compared to the correct answer of 130 which received only 319 responses. The relevance to difficulty is that the common wrong answer of "126" was not an option among the 4 choices for the problem on the 2002 MCAS test which resulted in a 70% correct rate for that question on the 2002 test with an IRT difficulty value of -1.4 (MCAS technical report[1]). The other Venn-Diagram item in the tutor had an even lower percent correct of %6. We can conclude that the relative ease of the multiple-choice question was a significant factor in the poor predictive performance of that item. A more in depth analysis of buggy answers using this same item as an example can be found in a poster submission [11].


18 Coach Wilson constructed a Venn diagram that shows the number of eighth-grade athletes who play football, basketball, and hockey.

Football
Basketball
Hockey
18
3
9
1
5
2
7

Which phrase best identifies the number 5 shown in the diagram?

A. the total number of athletes who do not play all three sports

B. the total number of athletes who play both football and hockey, but not basketball

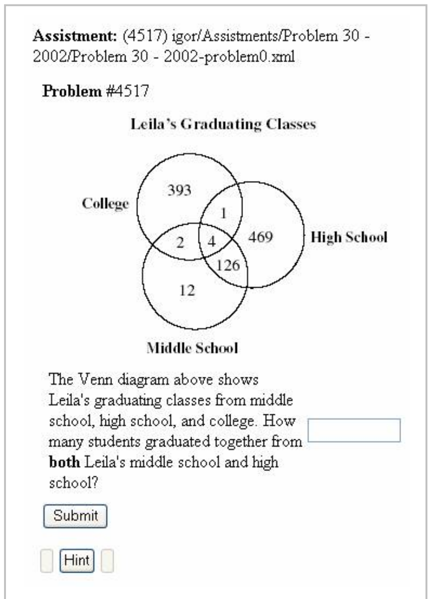
C. the total number of athletes who play either football or hockey, but not both

D. the total number of athletes who do not play basketball

Figure 5. MCAS test item 18, 2005

How can we correct for this disparity in difficulty between our online content and the test? One approach is to optimize the parameters of the Bayes net. Right now the Bayesian online network does not make a distinction with regard to question type. This means that the same guess and slip parameters are used for multiple-choice and text entry

[1] http://www.doe.mass.edu/mcas/2003/news/02techrpt.pdf


Assistment: (4517) igor/Assistments/Problem 30 - 2002/Problem 30 - 2002-problem0.xml

Problem #4517

The Venn diagram above shows Leila's graduating classes from middle school, high school, and college. How many students graduated together from **both** Leila's middle school and high school?

Figure 6. An ASSISTment for MCAS item 30, 2002

questions. To address this, two separate guess and slip parameters were created to represent each question type. This was done using equivalence classes in BNT. Our current guess and slips values were set *ad hoc*, an improvement could be made if these values were learned from the data. To learn the guess and slip values for multiple-choice and text entry questions we use the Expectation Maximization algorithm. Both skill priors and guess/slip parameters were allowed to be optimized. An increase in accuracy of 0.77% or 0.22 MAD was attained using the optimized guess and slip parameters. This result was statistically significantly different than the result with unoptimized parameters. To further increase the performance of the network each question's guess and slip parameter could be learned and hold out data could be used to train the parameters for the MCAS test network. Additionally, parameters for scaffold and original questions could be learned separately. There is currently no distinction made between the two.

11. Conclusions

We have seen how the methods of Bayesian networks and mixed-effects modeling produce very similar results. This gives us confidence in the execution of the two methods and provides a rare doubly reinforced result arrived at from two different angles. We have also answered our research question about the utility of finer-grained models and can report that the fine-grained WPI-39 and WPI-106 models provide the most accurate test prediction as confirmed by both Bayesian and mixed-effects methods. Teachers will be happy to know that there is some validity to the fine-grained models. We also have shown how inspection of the models' residuals can lead to a better understanding of transfer, content and the tutoring system as a whole.

There is one implication we would like to discuss. In the United States many schools and teachers are being encouraged to use frequent (i.e., monthly) testing to be "data-driven". The problem is that many tests used are unidimensional and do not provide cognitive reporting to the teachers. They also take up valuable class time. There seems to be a tension between tests that are fast and tests that are cognitively diagnostic. One implication of the success of fine grained model assessment is that it might be possible for states to use these models to develop a system of their own similar to ASSISTment that does all three of these things; 1) accurately assesses students, 2) gives finer grained feedback that tends to be more cognitively diagnostic and 3) saves time by assessing students while they are getting "tutoring" or taking a test.

12. Future work

Our static analysis in this paper has shown encouraging results. In addition to grouped parameter learning we are also working on adding time to our Bayesian Network model. A preliminary evaluation with our temporal WPI-5 has shown a prediction improvement of 3.6% over the non temporal WPI-5. We also plan to evaluate other years' datasets with both Bayesian and Mixed-effects methods. The results of those evaluations will tell if our results generalize beyond this particular year's dataset.

Acknowledgements

We would like to thank all the people associated with creating the Assistment system listed at www.assistment.org including the investigators Kenneth Koedinger, and Brian Junker at Carnegie Mellon. We would also like to acknowledge funding from the US Department of Education, the National Science Foundation, the Office of Naval Research and the Spencer Foundation. All of the opinions expressed in this paper are those solely of the authors and not those of our funders.

References

[1] Carmona1, C., Millán, E., Pérez-de-la-Cruz, J.L., Trella1, M. & Conejo, R. (2005) Introducing Prerequisite Relations in a Multi-layered Bayesian Student Model. In Ardissono, Brna & Mitroivc (Eds) User Modeling 2005; 10th Internaton Confrence. Springer. 347-356

[2] J. Collins, J. Greer, and S. Huang. Adaptive assessment of using granularity hierarchies and Bayesien nets. In Proceedings of Intelligent Tutoring Systems, pages 569--577, 1996.

[3] Fang Wei, Glenn D. Blank: Student Modeling with Atomic Bayesian Networks. Intelligent Tutoring Systems 2006: 491-502

[4] Feng, M., Heffernan, N. T., Mani, M., & Heffernan, C. (2006). 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.

[5] Feng, M., Heffernan, N.T., & Koedinger, K.R. (2006b). Predicting state test scores better with intelligent tutoring systems: developing metrics to measure assistance required. In Ikeda, Ashley & Chan (Eds.). Proceedings of the 8th International Conference on Intelligent Tutoring Systems. Springer-Verlag: Berlin. pp. 31-40. 2006.

[6] Hambleton, R.K., & W. J. van der Linden. (1997). Handbook of modern item response theory. New York, NY: Springer-Verlag.

[7] Hedeker, D. & Gibbons, Robert. D. (in progress). "Longitudinal Data Analysis": "Mixed-Effects Regression Models for Binary Outcomes" (chapter 9).

[8] Pardos, Z. A., Heffernan, N. T., & Anderson, B., Heffernan, C. L. The Effect of Model Granularity On Student Performance Prediction Using Bayesian Networks. 11th Internation Conference on User Modeling, 2007. Greece.

[9] Pardos, Z. A., Feng, M. & Heffernan, N. T. & Heffernan-Lindquist, C. Analyzing fine-grained skill models using Bayesian and mixed effect methods. In Luckin & Koedinger (Eds) Proceedings of the 13th Conference on Artificial Intelligence in Education. IOS Press.

[10] Reye, J. (2004). Student modelling based on belief networks. International Journal of Artificial Intelligence in Education: Vol. 14, 63-96.

[11] Rob Weitz, Neil Heffernan, Viswanathan Kodaganallur, David Rosenthal (submitted). The Distribution of Student Errors Across Schools: An Initial Study. AIED 2007

[12] Singer, J. D. & Willett, J. B. (2003). Applied Longitudinal Data Analysis: Modeling Change and Occurrence. Oxford University Press, New York.

[13] Snijders, Tom A. B., and Bosker, Roel J. (1999). Multilevel Analysis: An Introduction to Basic and Advanced Multilevel Modeling, London etc.: Sage Publishers, 1999.

Mining learners' traces from an online collaboration tool

Dilhan Perera, Judy Kay, Kalina Yacef, Irena Koprinska
School of Information Technologies
University of Sydney, Australia
{dper6077, judy, kalina, irena}@it.usyd.edu.au

Abstract: As university courses increasingly require students to use online tools in their studies, the opportunity arises to mine the resulting large amounts of student learning data for hidden useful information. In this paper we study the application of data mining to data collected from third year software development group projects using Trac, an online collaboration tool. We applied two very distinctive techniques, clustering and sequential pattern mining. The results point to the importance of leadership and group interaction, and give promising indications of whether effective leadership is occurring in a group. In addition, patterns were found which appear to indicate good individual practices. The results have considerable promise for advising groups at the start of their work and in early identification of both effective and poor patterns, in time for remediation.

1. Introduction

Educational institutions are increasingly using online computer-based tools to support both face-to-face and distance teaching. Because use of such tools generates a large amount of data about learners, there is the potential that mining that data can disclose useful information that can be used to improve learning and teaching. In this paper, we explore the use of data mining techniques on data collected from students using an online collaboration tool. In particular, our goal was to evaluate the effectiveness of data mining for: 1) gaining knowledge about the behaviour of students (and learners in general) working in groups, and 2) identifying which behavioural patterns are associated with positive and negative outcomes. This knowledge can then be used to inform students during the semester if their current behaviour was associated with positive and negative outcomes in the past, and also to provide them with advice on how to rectify problems. The data mining results could also be regularly presented to students in a comprehensible form, and thus facilitate reflection and self-regulation during the semester. Such 'mirroring' of student behaviour has shown promise with the use of suitable visualisation techniques [1]. This information can also play a very important role in discussions between teachers and learners.

Section 2 describes the background of the study: the groups and the online tool used. Section 3 presents the main data exploration, and the application of two data mining techniques: clustering and sequential pattern mining. We discuss the results, problems encountered and possible solutions. Section 4 summarises our conclusions.

2. Learners, learning environment and data

The learners were completing a senior software development project course. Over 12 weeks, and working in groups of 5-7 students, they were required to develop a software solution for a client. These projects varied from creating a computer-based driving ability test to developing an object tracking system for an art installation. The groups were required to use Extreme Programming (XP) [2], including use of user stories, small releases, and collective code ownership. Three semesters of data was collected for cohorts in 2005 and 2006, the last being the focus of this paper.

The learning environment used is an online professional software development tool, Trac [3]. It supports collaboration by integrating three tools: group Wiki for shared web pages, a task management system (also known as a 'ticketing' system) and a Subversion (SVN) browsing interface for source code version control. We have enhanced this professional tool with artefacts which extract information from learners' data (in the form of student models) (1) for students to peruse and reflect on and (2) for teachers to have a bird's eye view of what students are doing and where to focus their teaching efforts [4].

Data was captured whenever a learner performed an action in the Trac system, e.g. created or modified a Wiki page, ticket, or file in the SVN repository. The addition and reorganisation of directories in the repository was also captured. Information on each of these events was stored, including the time of the event, the author(s) and resources involved. Each event also included information specific to the event type, such as priorities for ticket events and a count of lines added and deleted for Wiki events. More details about the data and Trac can be found in [5].

In addition to these electronic traces, we also had the progressive and final marks, together with a very good understanding of the quality of each group's processes and product throughout the semester. The groups were ranked based on their performance where Group 1 denotes the strongest and Group 7 the weakest group.

3. Data mining

3.1. Simple statistical analysis

Before any data mining was carried out, the data was examined to see whether any simple statistics could distinguish the stronger from the weaker groups.

Firstly, we checked the total number of ticket events for each group. Intuitively we expect a large number to be associated with strong groups as the tickets allow group members to manage their work (e.g. allocate and accept tasks, post comments about tasks). Indeed, the results show that the top group had the highest number of ticket events. However, the performance of the other groups does not seem to correlate with the number of ticket events. For example, Group 2 had the lowest number. Interviews with this group indicated that they were reluctant to use the system as they felt it to be too cumbersome, and hence preferred to communicate their progress by other means.

Secondly, we looked at the distribution of the individual ticketing events (ticket created, accepted, reopened and closed). As tickets must be accepted by the assignee before they are recorded as being assigned, we expect the better groups to have near equal proportion of created and accepted tickets, which was the case. In contrast, some of the poorer groups had a much lower proportion of accepted than created tickets.

Again, this statistic is not very useful on its own: the poorest group displayed similar patterns to the top groups. Notably, we determined that in Group 4 one person logged in as each team member and entered all contributions for the group: this may explain for their seemingly ideal distribution of ticketing events.

Thirdly, we examined the usage span of the Wiki pages, i.e. the time between the first and last event on the page. Group 1 has the lowest number of Wiki pages but they were, on average, active for the longest period of time. This pattern is also evident for the next best group (Group 2), and the opposite pattern is displayed by the two poorest groups. There are several possible interpretations for this result and more work is needed to validate them. It could be that the better groups used the Wiki for more "active" purposes, such as group discussion or a logging of personal progress, while the poorer groups used the Wiki for more "static" purposes such as posting research and guidelines. Considering groups were required to post assessable work (such as reports) on the Wiki, it could also be that the better groups started this work earlier, while the poorer groups worked in a more compressed timeframe. However again, as shown by Group 5, this measure alone was not predictive of the quality of the group.

Lastly, we studied our SVN data and found that it was problematic for two reasons. First, as files were identified by their pathnames, we could not track unique files as they were often moved to different locations within the group repositories. Second, differences between SVN clients meant that data which was recorded on the number of lines added and deleted to committed files was not reliable. Thus, the only reliable SVN data was the time each commit took place. We use it to count the number of days on which SVN activity occurred for a group. The top group again was ranked highest on this measure: however, there was no obvious pattern in this statistic for other groups.

3.2. Clustering

As shown above, simple statistical exploration of the data together with a ranking of the groups was quite limited. The results suggested the need to consider multiple data attributes simultaneously, as well as removing the crudeness of relying on a single ranking system for the groups. Clustering allows us to use multiple attributes to identify similar groups in a unsupervised fashion, without the need to label the groups. In addition, it provides the opportunity to mine the data at the level of individual learners and in this way to examine the composition of each group.

Clustering can have useful applications in educational setting [6, 7]. For instance in [6] students using an intelligent tutoring system were clustered according to the types of mistakes made. The authors suggested that through the use of clustering, teachers could identify different types of learners and apply different remedial methods. A similar goal can be transferred to the current context, with clustering possibly identifying different styles of groups which may benefit from different styles of intervention. However, it must be noted that with a small number of groups, this could be performed by the teachers alone, without the aid of clustering results. Therefore, our primary goal was simply to assess whether our data contained features which could be translated through clustering into meaningful information about groups and individual learners.

3.2.1. Clustering groups

The most important problem was attribute selection. The performance of the algorithms is very sensitive to the quality of the attributes. Initially we chose a set of 8 numeric attributes representing ticketing behaviour such as the number of tickets and ticket events; the number of days on which tickets were opened, closed, or a ticket event occurred; and the ticket usage span (number of days between first and last event). We selected the classical K-means clustering algorithm and set the number of clusters to 3. The results are shown in Table 1 and reveal useful characteristics of the groups.

Table 1. Clustering ticketing behaviour using K-means (k=3) and 8 attributes

Clustered groups	**Distinguishing characteristics**
Groups 2, 3, 4 & 7	Overall low ticketing activity
Groups 5 & 6	Many tickets, Fewer ticketing events, Greater percentage of trivial and minor ticket priorities, Less accepting events
Group 1	Many tickets and many ticketing events, Lowest percentage of minor ticket priorities, More events where ticket priorities were changed or comments posted

One problem we met was that many attributes were correlated (some as high as 0.918, p=0.004). This motivated the manual creation of composite attributes that seemed to capture essential aspects of team performance. Attributes that measured total activity were excluded in favour of those that gave an indication of *how* Trac was used *when* it was used. Also, pairs of attributes were selected as together, they seemed to reveal information that was not obvious in either alone. Through this process, the 11 attributes listed in Table 2 were selected. Notably, 5 of them (marked with *) were ranked favourably when three of the Weka's [8] supervised attribute selection algorithms were used together (Information Gain, Relief and Support Vector Machines) with two slightly different group performance labelling.

Table 2. The 11 attributes selected for clustering

• Average number of events per ticket • Number of different days ticketing occurred • Average num. of ticket events per active ticketing day * • Percentage of ticket events not involving an 'action' on the ticket (comment added or a priority change) * • Percentage of ticket 'action' events where a ticket was accepted	• Average number of events per Wiki page • Average Wiki page usage span (days between first and last edit) * • Average number of edit days per page * • Average number of lines added per Wiki edit • Average number of lines deleted per Wiki edit * • Number of different an SVN activity occurred

Table 3 shows the K-means results using the above attributes. The same results were obtained with the EM clustering algorithm. Group 1 was again separated from the others.

Table 3. Clustering Trac activity using K-means (k=3) and 11 attributes

Clustered groups	**Distinguishing characteristics**
Groups 2, 3, 4 & 6	Moderate events per ticket, Infrequent Trac activity (tickets and SVN), Moderate % of ticket update events, Moderate number of lines added/deleted per Wiki edit
Groups 5 & 7	Moderately frequent Trac activity (tickets and SVN), High edits per Wiki page, Low number of lines added/deleted per Wiki edit, Low number of events per ticket, Low % of ticket update events
Group 1	Very frequent Trac activity (tickets and SVN), High events per Wiki page and per ticket, High Wiki page usage span, High % of ticket update events, High % of ticket accepting events

3.2.2. Clustering students

We also performed clustering of the individual students, with the hope that the group composition would reveal information that was missed when all individuals in a group were considered together. Table 4 shows the clusters obtained with K-means with the above 11 attributes, along with a label applied to each cluster based on an interpretation of its characteristics. The distribution of students from different clusters is presented in Table 5, with asterisks showing the cluster in which each group's manager was placed.

Table 4. Student clusters obtained using K-means

Cluster size	Distinguishing Characteristics	Cluster label
8 students	High ticketing activity, Involved in many tickets, High Wiki activity, Involved in many Wiki pages, Moderate SVN activity	'Managers'
9 students	Moderately high ticketing activity, Ticketing occurring on many different days, Moderate Wiki activity, Very high SVN activity	'Trac-Oriented Developers'
11 students	Low ticketing activity, Low Wiki activity, Low SVN activity	'Loafers'
15 students	Moderately low ticketing activity, Moderately low Wiki activity, Many Wiki events on days which Wiki events occurred, Many SVN events on days which SVN events occurred	'Majority'

Table 5. Distribution of students from each cluster shown in Table 3

	Managers	Trac-Oriented Developers	Loafers	Majority
Group 1	*1	3	1	1
Group 2	*1	0	1	3
Group 3	0	1	2	**3
Group 4	*1	3	2	0
Group 5	3	*1	0	3
Group 6	*1	1	3	1
Group 7	*1	0	2	4

Some differences between previously coupled groups began to emerge. For example, Groups 2 and 3 differ by Group 3's lack of a 'manager'. This was consistent with our knowledge of the leadership problems this group encountered, with the original manager leaving the course and another group member taking over. The lack of 'Trac-oriented developers' in Group 2 was validated in a group interview where the main developers expressed a reluctance to use Trac. Group 5 is also distinctive in its excess of 'managers', perhaps suggesting too many managerial and organisational processes were occurring at the expense of actual work being done. This is further complicated by their designated manager being placed in the cluster which performed more technical than managerial work. One possibility is that this weak leadership resulted in others reacting to fill the manager's role, with their technical work subsequently being compromised. This may be a pattern to be aware of in future groups.

3.3. Sequential pattern mining

An important aspect of our data which is ignored by mining techniques such as clustering is the timing of events. We believe that certain *sequences of events* distinguish the better groups from the weaker ones. These sequences may represent characteristic team interaction on a specific resource, or group members displaying specific work patterns across the three aspects of Trac. A data mining technique which considers this temporal aspect is sequential pattern mining [9, 10]. It finds sequential patterns that occur in a dataset with at least a minimal level of frequency.

To extract patterns, the data first needs to be transformed into a set of sequences. The first problem is how to break down the long traces of events into meaningful sequences. We considered three possibilities:

- A sequence per resource, where a separate sequence is obtained for the events on each ticket, Wiki page, and SVN file.
- A sequence per group 'session', where sessions are formed by cutting up the group's event list where gaps (of no activity) of a minimum length of time occur. A related sequence formation method is a sequence per author 'session' – before the event list is broken up into sessions, the event list for each group member is extracted. Sessions are then formed from these event lists of individuals.
- A sequence per 'task', defined by a ticket. The task sequence includes all ticket events on that ticket, and SVN and Wiki events referring to it.

The second problem is to define an alphabet to represent the events inside the sequences. In our case this involved removing certain author and resource identification information, as well as collapsing similar consecutive events into single alphabet items. The three alphabets used are summarised in Table 5.

Table 5. Three alphabets used

Alphabet 1: (i,X,j)	**Alphabet 2: (A,i,X)**	**Alphabet 3: (i,X,A)**
The number of consecutive events (i) occurring on a particular Trac medium (X) and the number of individuals (j) involved in the events.	The number of consecutive events (i) performed by a single author on a specific resource of Trac medium type X and the role of the author (A).	The number of consecutive events (i) occurring on a particular Trac medium (X), and the role of the author (A).
A new item is generated when an event appears in the sequence:		
from a different Trac medium	by a different author	from a different Trac medium or by a different author

Alphabet 1 was introduced in [5] and used with the group's event sequence. Alphabet 2 was developed for use with the resource and Alphabet 3 for the task sequences. Alphabets 2 and 3 were specifically developed to provide a tighter integration of the research with psychological theories of group work. One example is the Big Five framework which emphasises the importance of leadership, mutual performance monitoring, backup behaviour, adaptability, and team orientation [11]. Other theories also emphasise the influence of leadership in determining the success of groups. In addition, the role of 'tracker' in XP was thought to correspond to the Big Five function of performance monitoring. For these reasons, author roles in these alphabets were identified as: L (leader), T (tracker), and all other authors identified as a, b, c, etc. in order of appearance on the resource.

Alphabet 2 also allowed us to examine whether earlier authors returned to edit the resource after it was edited by another author, representing a group interaction on the resource. In addition, leaders and trackers, though always displayed with the same symbol, were still placed in the ordered author list, allowing us to identify resources which were created by group members other than the leader or tracker.

3.3.1. Patterns observed in group sessions

As observed in the previous year [5], the better groups had many alternations of SVN and Wiki events, and SVN and Ticket events whereas weaker groups had almost none. We hope these patterns correspond to documentation in the Wiki about the SVN commits and to tickets being updated following SVN commits. Although we now have the ability to store the supporting events for each pattern, it is still arduous to trace events back to actual Trac actions to test such suspicions.

Through an analysis (using sequences per author), we observed that individuals in the top group displayed a higher than average level of alternating SVN and Wiki events. The top group also had the highest proportion of author sessions containing many consecutive ticket events (matching their high use of ticketing) and SVN events (suggesting they committed their work to the group repository more often).

In contrast, the least successful group displayed a high level of alternating Wiki and ticketing events, but also had a distinctive lack of sequences containing SVN events. Although it also showed frequent consecutive Wiki events, which was a characteristic associated with the most successful 2005 group, their lack of SVN events seem to suggest that their Wiki and tickets were not being used in support of software development. This was validated by the course coordinators, who described this group as technically less proficient.

A more detailed analysis of these patterns revealed that the top group used the Ticket more than the Wiki, whereas the weakest group displayed the opposite pattern. The use of the ticketing system may be indicative of actual work being done, as it is more task-oriented than the Wiki. This trend was even stronger when we exclusively considered the sessions of the group leaders. This suggests that the work of the group leaders clearly influences the success of the groups, with the leaders of the top groups using tickets more than the other leaders. Note that this does not just include leaders assigning work to other group members (i.e. tickets being created), but also leaders commenting on tickets and following up assigned work. In addition, the data suggested these group leaders were much less involved in technical work, suggesting work was being delegated properly and the leader was leading rather than simply doing all the work. In contrast, the leaders of the poorer groups either seemed to use the Wiki (a less focused medium) more than the tickets, or be involved in too much technical work.

3.3.2. Patterns observed in task sequences

A task sequence can be more informative than a session sequence because it only contains events that are related, as opposed to events that occurred in the same window of time. Task sequence mining also shows how the different groups used the three elements of Trac in completing project tasks.

We found that the two top groups had by far the greatest percentage support for the pattern (1,t,L)(1,t,b), which were most likely tickets initiated by the leader and accepted by another team member. The fact this occurred more often than (1,t,L)(2,t,b), suggests that the better groups were distinguished by tasks being performed on the Wiki or SVN files before the ticket was closed by the second member. Notably, the weakest group had higher support for this latter pattern than the former. As a validation to this interpretation, we also found that the best group was one of only two to display the patterns (1,t,b)(1,s,b) and (1,s,b)(1,t,b) – the first likely being a ticket being accepted by a team member and then SVN work relating to that task being completed and the second likely being work being done followed by the ticket being closed. The close coupling of task-related SVN and Wiki activity and Ticket events for this group was also shown by relatively high support for the patterns (1,t,b)(1,t,b)(1,t,b), (1,t,b)(1,s,b)(1,t,b) and (1,t,b)(1,w,b)(1,t,b). Interestingly, the poorest group displayed the highest support for the last pattern, but no support for the former, again indicating their lack of SVN use in tasks.

Another series of patterns which characterised the best groups were tickets being initiated by non-leader group members. These tickets were evidently not created just for the sake of the course requirements, as this group also showed high support for wiki and SVN patterns by these team members. An example is (2,t,a)(1,s,a)(1,t,a), which may likely be a ticket being created by a team member for him/herself, the ticket being accepted, work being committed related to the ticket, and the ticket finally being closed.

A pattern which characterised the poorest group was the tracker creating and editing many tickets, for example in the patterns (1,t,T), (1,t,T)(1,t,b) and (2,t,T). As it is the tracker's role to follow up tasks, their general involvement in tickets should not be a matter of concern. However these patterns may have been more common in the poorer groups because of weaker leadership, resulting in trackers performing a share of the leader's role. Conversely, the better groups may have shown less involvement by the tracker because of prominent leaders who were also able to perform tracker duties. An alternative explanation may be that group problems lead to greater tracker activity.

3.3.3. Patterns observed in resource sequences

Apart from good individual practises, such as SVN commits being documented on Wiki pages, another aspect of good group work which we hoped the original sequence generators and alphabets captured was interaction between team members. For example, it was hoped that events such as (3,w,2) would be indicative of 2 group members interacting on the Wiki. However we cannot be certain about this conclusion because the pattern does not tell us that the three events occurred on the same Wiki page. To better capture interactions between group members we decided to examine sequences across specific resources.

By forming new alphabet items when a new author appeared in the resource's event sequence, and by identifying the managers and assigning within-resource roles to other group members, we were better able to track these group interactions. We found that the top group had very high support for patterns where the leader interacted with group members on tickets, such as (L,1,t)(b,1,t)(L,1,t). The poorest group in contrast lacked these interaction patterns, and had more tickets which were created by the Tracker rather than the Leader, suggestive of weaker leadership. The importance of leadership and leadership style has been emphasised by the Big Five theory and other classic psychological studies , and the success of our data mining in detecting differences in leadership is especially promising.

In addition, the best group displayed the highest support for patterns such as (b,3,t) and (b,4,t), suggestive of group members making at least one update on tickets before closing them. In contrast, the weaker groups showed support mainly for the pattern (b,2,t), most likely indicative of group members accepting and closing tickets with no update events in between. These extra events on tickets may be important in allowing the team to monitor each other (one of the other Big Five aspects) and also indicates the presence of frequent task-focused communication in successful groups.

Patterns indicative of interaction on tickets in the best group were not just limited to the group leader and one other member. Significantly, this group also displayed higher than average support for patterns of interaction involving multiple team members, such as (b,1,t)(c,1,t)(L,1,t). This is especially notable on tickets, which usually only directly involve two individuals (the assigner and the assignee). The involvement of a third person may be indicative of a number of desirable group characteristics, such as mutual performance monitoring, team orientation (two elements of the Big Five), or collective code ownership (an Extreme Programming practice). It should also be noted that another top group in 2006 and the top group in 2005 displayed similar patterns of long interactions on Wiki pages rather than on tickets. This may suggest that the interactions themselves are more important than the medium on which they take place.

Another pattern with above average support in the best group was consecutive events on SVN files by an individual author, for example (a,2,s) and (a,3,s). These may have been caused by group members committing to files more frequently, or group members requiring less intervention from others in work being completed by them. Regardless of the interpretation, it is interesting to note that the poorest group, despite lacking these patterns, also lacked the pattern (a,1,s)(b,1,s), where a second team member commits to the file. Instead, we found that in this group it was more common for the group *leader* to be involved in a file after just one commit by the original author. Again this suggests that the leader intervened on technical aspects of the project and may be a sign of group problems. However because of our noted problems with identifying unique files by pathname, it may also simply be that the group leader moved files around in the repository frequently.

3.3.4. Some Limitations of Sequential Pattern Mining

A number of issues emerged during the use of this technique, ranging from limitations in the data to how output was interpreted. Currently our data contains only modification and creation events. The common situation where a team member views another's work but does not feel the need to modify it was thus effectively ignored. This emphasises the need to incorporate data from sources such as weblogs. A problem with our mining program itself is the lack of gap constraints – as noted in [9], a frequent subsequence of (X)(Y) may not be meaningful if many other events occur between X and Y. Another issue already stated in [5] is the need for more automated methods of processing output which go beyond manual sorting techniques. Emergent pattern mining [12] and contrast sets [13] may be possible solutions. Finally, there still remained the need to assign meaning to the patterns in order to learn about group work in general. The importance of finding the right balance between alphabets that are too abstract (limiting interpretation) and those which are too specific cannot be understated.

4. Conclusion

We performed mining of data collected from students working in teams and using an online collaboration tool in a one-semester software development project. Clustering was applied to find both groups of similar teams and individual members, and sequential pattern mining to extract sequences of frequent events. The results revealed interesting patterns characterising the work of stronger and weaker students. Some of them are specific for our context (i.e. the course requirements and tool used). Others are more generic and consistent with psychological theories of group work, e.g. the importance of group interaction and leadership for success.

This knowledge can be used in several valuable ways. Firstly, we already lecture students on various aspects of group work. This work will enable us to give concrete examples of how of the patterns associated with some of the general principles, such as those for leadership and monitoring activity. Just from the clustering, we can point to the data in Tables 4 and 5. Each of these can be illustrated with actual wikis and tickets. We can explain how these patterns were associated with leadership behaviours that were either more or less effective. Secondly, we can automate the identification of the most salient patterns described above and present these to the students for their own group discussions as well as using them in meetings with teachers. This may help to rectify ineffective patterns of group operation and consolidate effective ones. Essentially, this work will enable us to provide regular feedback to students *during* the semester if their current work behaviour is more likely to be associated with positive or negative outcomes and where the problems are. Teachers can also greatly benefit from such feedback. Although more work is needed before more formative and timely feedback can be provided, students and teachers could be made aware of the current findings and limitations, to encourage better group practice and teaching and learning experience.

This work also highlights some of the data mining challenges posed by educational data, e.g. it is temporal, noisy, correlated, incomplete, may lack enough samples for some tasks. We addressed some of them by providing specific solutions for our task. There are many avenues for future work. Both the data mining and the data itself could be extended and enriched. For example, the addition of a chat module can increase the student usage of Trac and generate useful data. Clustering can be improved by the collection of data for more groups and individuals. New alphabets could also be developed for the sequential pattern miner to reveal as-yet hidden work behaviours and group interactions. More work is also needed in assessing the generalization ability of the discovered patterns.

References

[1] Kay, J., Maisonneuve, N., Yacef, K. and Reimann, P., The Big Five and Visualisations of Team Work Activity. In *Intelligent Tutoring Systems*, Taiwan, 2006, Springer-Verlag, 197-206.

[2] Extreme programming. www.extremeprogramming.org, last accessed June 2007.

[3] http://trac.edgewall.org/, last accessed June 2007.

[4] Kay, J., Reimann, P. and Yacef, K., Visualisations for team learning: small teams working on long-term projects. In *International Conference on Computer-Supported Collaborative Learning*, New Brunswick, USA, 2007.

[5] Kay, J., Maisonneuve, N., Yacef, K. and Zaiane, O. Mining patterns of events in students' teamwork data. Heiner, C., Baker, R. and Yacef, K. eds. *Educational Data Mining Workshop*, Taiwan, 2006, 45-52.

[6] Merceron, A. and Yacef, K. Clustering students to help evaluate learning. In Courtiat, J.-P., Davarakis, C. and Villemur, T. eds. *Technology Enhanced Learning*, Springer, 2005, 31-42.

[7] Legaspi, R., Sison, R., Fukui, K.-i. and Numao, M. Cluster-based predictive modeling to improve pedagogic reasoning. *Computers in Human Behavior*, 2007.

[8] www.cs.waikato.ac.nz/ml/weka. www.cs.waikato.ac.nz/ml/weka, last accessed June 2007.

[9] Srikant, R. and Agrawal, R., Mining sequential patterns: Generalizations and performance improvements. In *Fifth Int'l Conference on Extending Database Technology (EDBT)*, Avignon, France, 1996.

[10] Agrawal, R. and Srikant, R., Mining Sequential Patterns. In *International Conference on Data Engineering (ICDE95)*, 1995, 3-14.

[11] Salas, E., Sims, D.E. and Burke, C.S. Is there a "Big Five" in teamwork? *Small Group Research*, *36* (5), 2005, 555-599.

[12] Dong, G. and Li, J., Efficient Mining of Emerging Patterns: Discovering Trends and Differences. In *International Conference on Knowledge Discovery and Data Mining*, 1999, 43-52.

[13] Bay, S. and Pazzani, M. Detecting group differences: mining contrast sets. *Data Mining and Knowledge Discovery*, *5* (3), 2001, 213-246.

Mining On-line Discussions: Assessing Technical Quality for Student Scaffolding and Classifying Messages for Participation Profiling

Sujith RAVI , Jihie KIM and Erin SHAW
University of Southern California/Information Sciences Institute
4676 Admiralty Way, Marina del Rey, CA 90292 USA

Abstract. On-line collaborative discussions play an important role in distance education and web-enhanced courses. Automatic tools for assessing student activities and promoting collaborative problem solving can provide a better learning experience for students and also offer useful assistance to teachers. This paper presents two novel instructional tools that apply data mining and information retrieval techniques. First, we describe an approach that could be used to scaffold undergraduate student discussions by retrieving useful information from past student discussions. The tool exploits both the discussions from the same undergraduate course and the ones from a graduate-level course. The second part of the paper presents an instructional tool that profiles student contributions with respect to student genders and the roles that students play in discussion. We apply speech act classifiers that automatically identify whether the given message contains questions and/or answers, and use the classification results in profiling male and female student contributions. Our initial evaluation of the scaffolding tool shows that discussions from the same course contain more number of similar concepts than the ones from the graduate-level course. However, technical quality of graduate-level discussions is higher. The results from the profiling tool indicate that female participation in undergraduate-level discussions is lower than that in graduate-level discussions, and graduate female students post more questions and answers compared to undergraduate female students.

Keywords: On-line discussion board, Question Answering, discussion assessment

1. Introduction

On-line discussion boards play an important role in distance education and web enhanced courses. Recent studies have pointed to on-line discussion board as a promising strategy for promoting collaborative problem solving courses and discovery-oriented activities [Scardamalia & Bereiter, 1996; Koschmann, 1996]. However, other research indicates that existing systems for on-line discussion may not always be fully effective in promoting learning in undergraduate courses. For example, some analyses of collaborative on-line learning indicate that student participation is low or weak, even when students are encouraged to participate [Palloff & Pratt 1999; Kim & Beal, 2006]. As course enrollments increase, with some introductory courses enrolling several hundred students, the heavier on-line interaction can place a considerable burden on instructors and teaching assistants. We are developing instructional tools that can automatically assess student participation and promote interactions by sending responses to student messages.

In this paper, we present two novel tools that apply data mining and information retrieval techniques. First, we describe an approach that scaffolds undergraduate student discussions by retrieving useful information from past student discussions. We first semi-automatically extract domain terms from textbooks specified for the courses, and use them in modelling individual messages with term vectors. We then apply TF*IDF (term frequency and inverse document frequency) [Salton, 1989] in retrieving useful information from past student discussions. The tool exploits both the discussions from the same undergraduate course and the ones from a graduate-level course that share similar topics. Our hypothesis is that since graduate discussions are full of rich, elucidating dialogues, those conversations can be recommended as interesting references for undergraduates. We also hypothesize that we can scaffold discussion by sending messages from past students who had similar assignments or problems regarding course topics. We analyze the usefulness of the retrieved information with respect to relevance and technical quality.

The second part of the paper presents an instructional tool that profiles student contributions with respect to student genders and the roles that they play in discussions. We apply two speech act (SA): question classifier and answer classifier [Ravi and Kim, 2007]. The question classifier identifies messages that play a role of asking questions and the answer classifier detects messages with answers in response to a previous message. We use the classification results in profiling male and female student contributions.

We performed an initial evaluation of these tools in the context of an undergraduate Operating Systems course offered by the Computer Science department at the University of Southern California. The current results for the scaffolding tool indicate that past discussions from the same course contain many similar concepts that we can use in guiding the discussions. Although graduate level discussions did not contain many similar topics, their technical quality was higher. The initial results from the profiling tool show that female participation in undergraduate-level discussions is lower than that in graduate-level discussions, and graduate female students post more questions and answers compared to undergraduate female students.

2. Developing Scaffolding Capability: Mining Useful Information from Past Student Discussions

This section describes our approach to developing an instructional tool that scaffolds student discussions by retrieving useful information from past student discussions. The new scaffolding capability is built on top of an existing question answering module that sends responses to student queries [Feng, et al., 2006a]. First of all, in order to handle noisy student messages and retrieve more useful information for scaffolding, we extended the answer extraction procedure to focus on technical terms. We also expanded our corpus to include graduate-level courses so that graduate student conversations as well as undergraduate ones can be used. The retrieved information can be sent out as responses when discussion threads are short or stagnant. Figure 1 shows the system components and steps involved in retrieving responses. The following sections explain the steps shown in the figure.

2.1 Modeling Messages with Technical Terms

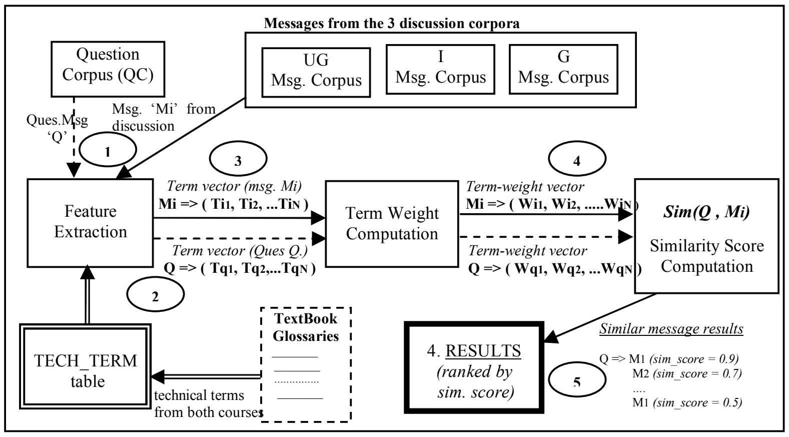


Figure 1. Mining relevant or similar messages from past discussions

We use discussion corpora from two Operating Systems: an undergraduate level course and a graduate-level course. The undergraduate course is held every semester, and students and instructors have access to discussion boards to exchange information regarding various topics covered in the course. We first removed messages from administrative forums that contain non-technical discussions. The undergraduate discussion corpus was created with data from four semesters, and consisted of 3788 messages from students and 531 messages from the instructor. The graduate discussion corpus available to us was comparatively smaller and obtained from two semesters and contained 957 messages. The past discussions from the two courses were divided into three corpora – (1) "UG" - undergraduate student messages, (2) "I" - messages posted by the instructor for the undergraduate course, and (3) "G" - messages from graduate student discussions[1]. Step 1 in Figure 1 represents this process.

We found that discussion data from students, especially undergraduate students, are highly incoherent and noisy. The raw data includes humorous messages and personal announcement as well as technical questions and answers. Student messages are very informal and there are high variances in the way they present similar information. A lot of messages on programming assignments also include programming code. Due to the noise and the highly informal nature of messages posted by students in the discussion forums, in modeling the messages for scaffolding, we use only technical terms present in the messages.

The technical terms that we use were extracted from the glossaries of the undergraduate and graduate text books. The glossaries were automatically scanned and

[1] We applied TextTiling (Hearst,1994), to segment every document (message) into semantically-related segments (tiles) and subsequently process tile units.

processed. We created a "TECH_TERM" table that contains all the technical terms from both courses. This corresponds to step 2 in Figure 1.

Individual messages are then modeled with a term vector of the following form (Ste p 3 in Figure 1):

$$M_i =< T_{i1}, T_{i2}, ..., T_{iN} >$$

where N is the total number of technical terms in the domain and $T_{ij} = 0$ if a term is missing in that message.

*2.2 Retrieving Relevant Messages with TF*IDF*

After term vectors are created for all messages in the corpus, the weights for the terms are computed. The weights are used in calculating similarity scores between messages, such as similarity of a new message and a message in a corpus.

In computing term weights, we use TF*IDF (term frequency * inverse document frequency) [Salton, 1989]. Messages with same technical terms are more likely semantically related. This information is captured in TF (term frequency). TF tends to weight the commonly occurring terms more and give low weights to rare technical terms. IDF (inverse document frequency) fixes it by introducing general importance of the term. Equation 1 describes the method to compute individual TF*IDF weight values for each term.

Equation 1. TF-IDF feature weight computation

$$W_{ik} = TF_{ik} * \log(N / n_k)$$

Tk = term *k* in document (message) *Mi*
TFik = frequency of term *Tk* in document (message) *Mi*
IDFk = inverse document frequency of term *Tk* in *C*
N = total number of documents (messages) in the discussion corpus *C*
nk = the number of messages in *C* that contain *Tk*
IDFk = log(N / nk)
Wik = TF-IDF weight for term *k* in document (message) *Mi*

Equation 2. Cosine similarity computation (between document/message Di and question message Q)

$$sim(Q, D_i) = \frac{\sum_{j=1}^{t} w_{q_j} w_{d_{ij}}}{\sqrt{\sum_{j=1}^{t} (w_{q_j})^2 \sum_{j=1}^{t} (w_{d_{ij}})^2}}$$

D_i = (W*di1*, W*di2*, ... W*dit*)
Feature-weight vector representing document/message *Di* in the discussion corpus.

Q = (W*q1*, W*q2*, ... W*qt*)
Feature-weight vector representing question message *Q*.

W*dit* = TF-IDF weight for feature term *t* in message *Di*.
W*qt* = TF-IDF weight for feature term *t* in question message *Q*.

The term vector for each message is converted into a corresponding term-weight vector, where each term-weight represents the TF*IDF measure for a particular technical term existing in the message (Step 4 in Figure 1).

Given a new message posted on the discussion board, a new term vector is created with term weights in the same way. We use cosine similarity to determine relevance between the new message and a message in the corpus (Equation 2). This measures the cosine of the angle between the two term-weight vectors representing the two messages. Messages are ranked by their scores in order to find the most relevant messages.

We compare the given message with the messages from the three discussion corpora (UG, I, G) and compute their corresponding similarity scores. Once all the similarity scores are computed, we select the messages with the similarity score higher than 0.5. To ensure that there is some amount of technical content in the message, we retrieve the messages with at least three technical terms. The results can be sent as a response (Step 5 in Figure 1).

2.3 Evaluation of System Responses

For evaluating usefulness of retrieved information, we created a new message corpus with a separate discussion data from a recent semester, Fall 2006. We removed administrative forums that contain non technical discussion as we did for existing corpora. We extracted the first messages in all the threads in the new corpus, which result in 207 messages. The messages were used in triggering the above procedure and retrieving relevant messages from the three corpora (UG, I, G). With UG and I, the system found responses for 132 messages and 71 messages respectively. G provided answers for only 8 messages. Since there are many responses for some of the messages, the actual number of messages retrieved was large. We randomly selected 13 messages that have responses from UG and I so that human evaluators could perform the analysis within a day. For the 13 messages there were 123 and 27 responses from UG and I, respectively. For G, we use all 8 messages and the system provided 13 responses for them.

Four human evaluators rated the system responses with respect to 1) technical quality and 2) degree of relevance to the given message. Figure 2 shows the average ratings of technical quality. Messages with high quality would contain information pertaining to the operating systems course and related areas, and would convey some technical knowledge on the subject. Such messages may also present a deeper understanding of the subjects rather than providing shallow answers. Some student messages included humour, and other extra information besides technical discussions. This is especially true for undergraduate student discussions. The technical quality of undergraduate student messages does not seem as good as others. The messages from the instructor for the undergraduate course on the other hand are more specific and directed towards answering some questions/problems posed by students on the discussion board. The messages in this case, contain more technical information and sometimes give detailed explanations for specific concepts or solutions to problems. However, in addition to these, the instructor corpus also contains many short messages with Yes/No answers, suggestions or alternative approaches in response to specific questions posted by students. So, some of the messages were rated lower than others. The number of message retrieved from G is less. However, their technical quality seems a little better than the other two (3.31).

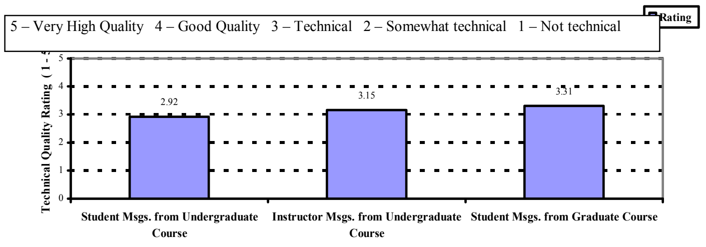


Figure 2. Technical Quality Evaluation : Ratings for messages (based on technical quality)

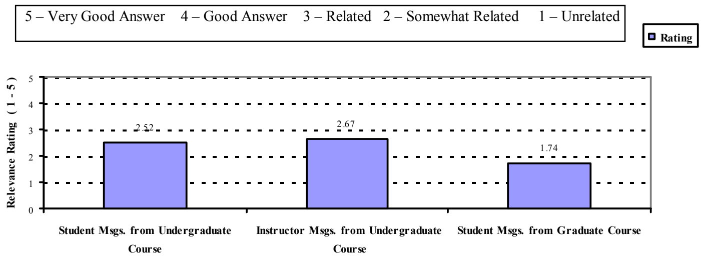


Figure 3. Relevance Evaluation : Ratings for message results (based on relevance)

Figure 3 shows the average ratings of relevance. In this case, the evaluators assessed relevance of system responses to the given message. The evaluators checked "how useful the message is as a response" or "how related the message was to the given message". The UG results returned by the system tend to be less relevant (average rating ~ 2.52) to the given message than the ones from I (average rating ~ 2.67). The average rating for responses from G is the lowest (~1.74). Even though the graduate student messages tend to be more technical than undergraduate student messages, they might not be fully relevant to the given undergraduate student message. Although there are some common topics between the undergraduate and graduate courses, the kinds of problems that they are discussing seem different. In addition, we had a smaller set of data for G, which made it harder for the system to extract relevant results.

3. Profiling student participation with gender data and speech act classifiers

In this section, we describe new Speech Act classifiers based on [Ravi and Kim, 2007] that identify messages containing questions and answers, and use the results of the classification to report the message distribution for male and female students in two different courses.

3.1 Speech Act Classifiers

Within a thread, each message plays a different role. A message may represent a question, an intermediate response, such as a clarification or correction, or a solution, such as a suggestion or answer. We have defined a set of speech acts (SAs) [Austin, 1962; Searle, 1969] that relate pairs of messages in the discussion corpus. For example, a pair of messages in which the first message is an elaboration of the second is given the label 'elaboration'. Other speech act categories include question, answer, correction, acknowledgment, etc.

We grouped the speech acts to create two binary classifiers, one which identifies questions and the other which identifies answers in questions and answer type discussion threads [Ravi & Kim, 2007]. The classifiers use Ngram features and an SVM strategy for machine learning. Using the question classifier, we identified the question messages posted by students from among all the messages in the corpus. The answer classifier was used similarly to identify the messages from the same corpus that contained answers or suggestions. Questions and answers occasionally appear in the same posts so that a particular message may be classified as both a question and an answer, i.e. the identifications are not mutually exclusive.

3.2 Gender Classifier/Distribution

Using the new classifiers, we then analyzed student participation by gender for two discussion board datasets. The undergraduate corpus contained messages from threaded discussions in an undergraduate level Operating Systems course at USC; the graduate corpus contained messages from discussion in a graduate level Operating Systems. We used the question and answer classifiers on each discussion set to identify the messages containing questions and answers respectively within the set. We use the results to compare types of posts (question or answer), classes (undergraduate or graduate) and students (by gender).

The undergraduate corpus contained 699 student posted messages, out of which we identified 370 question messages and 157 answer messages. The graduate corpus contained 154 student posted messages, out of which we identified 85 question messages and 26 answer messages.

In our analysis, we consider only those messages that are classified as either a 'Question' or 'Answer' or 'Both' by the Speech Act classifiers. Table 1 shows the gender distribution of both the total number of students registered in a course and the number of students who contribute to discussions in a course. In the undergraduate course, 89% of students are male and 11% are female; in the graduate course, 86.4% are male and 13.6% are female. Not all of the registered students participate in discussions: In the undergraduate course, 91.3% of the students who participate in discussions are male and 8.7% are female. In the graduate course, the percentage of participating female students increases to 12.5% of participating students. Table 2 gives the gender distribution of the types of messages posted in both courses. In the undergraduate course, most of the questions and answers are posted by male students. In comparison, female students in the graduate course post a higher percentage of messages, especially questions.

3.3 An Application of Gender Classifier/Distribution

This type of classification can help us better analyze how students are using a discussion board. For example, it could help determine if female students prefer text-based discussion as a means of communication/collaboration in highly technical courses.

The question is of interest to the STEM research community, especially those interested in gender issues. We used the classifier to generate the results shown in Figure 4. Results are mixed: In the undergraduate course (left) female students as a group posted fewer questions and answers than male students as a group. However, the reverse was true for female graduate students (right): Though they posted fewer messages overall, a greater percent of females posted more questions and more answers than that of males.

Table 1. Male/Female student distribution (registered / participating) in the two courses

Course	Students	Male s	Female s
Undergraduate	Among total students registered	89%	11%
	Among students posting messages (participating in discussions)	91.3%	8.7%
Graduate	Among total students registered	86.4%	13.6%
	Among students posting messages (participating in discussions)	87.5%	12.5%

Table 2. Male/Female student participation by message type

Course	Message type	Male s	Females
Undergraduate	Question messages	97%	3%
	Answer messages	96%	4%
Graduate	Question messages	78%	22%
	Answer messages	88%	12%

Graduate level female students appear to engage more actively than undergraduate female students in discussions and utilize the discussion board as a collaborative tool to help in their learning. These results may be used for further pedagogical analysis, to develop techniques to help instructors better assess student participation from the discussions, which has the potential to increase and improve participation so as to promote a better understanding of the subject topics.

4. Related Work

Developing automatic natural language based question answering systems has been a field of research for many years [Hermjakob et al., 2000; Pasca et al., 2001], but question answering in an open domain is still an unsolved problem. Much research in the NLP community has focused on developing techniques to understand and answer user queries. Question Answering (QA) systems provide a general framework for parsing questions, searching documents and retrieving relevant answers. Some QA systems employ semantic based and ontology based approaches. Most systems assume that queries are concise, coherent and of a particular form (e.g., a factoid question), and can be answered with short phrases. But queries posted to a discussion board often span multiple sentences, are incoherent (in the computational sense) and include extra (informal) content and lengthy descriptions, especially in technical discussions. Providing question answering support for student discussions still remains a challenge.

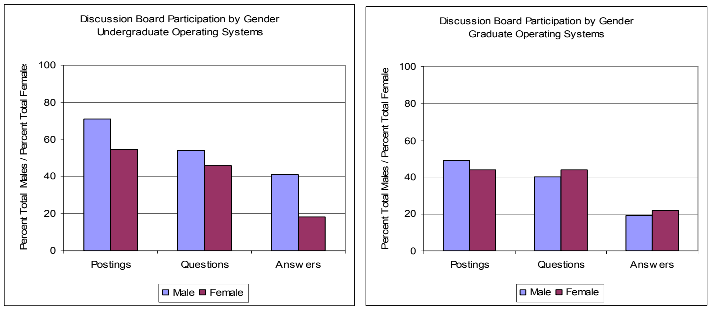


Figure 4. Comparison of participation by gender in undergraduate and graduate courses

The goals of our system include 1) referring students to relevant past discussions from related courses, 2) promoting interactions among students and 3) aiding instructional assessment through automatic tools that assess quality and participation, all of which contribute to a better learning environment for students. Related work on dialogue modeling investigates different ways to manage continuous dialogue for personal assistants [Nguyen and Wobcke, 2005] or scaffold conversations in a virtual meeting space [Isbister et al., 2000]. In contrast, we focus on optimizing a one step question response by mining knowledge from archived materials asynchronously. More similarly, the work in [Marom et al., 2005] generates helpdesk responses using clustering techniques, but their corpus is composed of two party, two turn, conversation pairs, which makes it easier to bypass the complex analysis of discussions among multiple parties.

In the area of online learning, much attention has been paid to the analysis of student learning behaviors in online communications. Various frameworks have been proposed for characterizing and analyzing computer mediated communication in the context of collaborative discussions [Shaw, 2005], email and chat exchanges [Cakir et al., 2005], knowledge sharing [Soller & Lesgold, 2003] and general argumentation [Feng et al., 2006a, 2006b, Painter et al., 2003], but none have been sufficiently relevant or fine-grained to facilitate data mining and answer extraction in threaded discussions.

5. Summary and Discussion

This paper presents some approaches for mining information automatically and building tools to help students participating in on line discussions using various Natural Language Processing and Machine Learning techniques. Our work focuses on technical discussions among students enrolled in operating systems courses both at the undergraduate and graduate levels. We first presented an automated way of helping students asking questions in discussion boards by retrieving useful messages and related discussions from the past both from the same course and a graduate level course. Next, we used automatic speech act classifiers to classify student messages as Question or Answers and used this to analyze the distribution of student participation by gender. We performed the analysis for both undergraduate and graduate level discussions and presented our results.

In order to improve the quality of the results and extract better and more relevant/related answers from past graduate discussions for questions posted by undergraduates, we plan to include more discussion data and additional features, such as topic profiles. We are also looking at other NLP approaches like LSA (Latent Semantic Analysis) to infer and extract semantic information and relations from the messages posted on the discussion board. Another interesting study that we wish to perform in the future is to distinguish the answer messages based on the complexity: short answers vs. complex answers, and perform separate analysis on each type. We also plan to extend participation profiling with topic analysis and contribution changes over time.

References

[1] Austin, J., 1962. How to do things with words. Cambridge, Massachusetts: Harvard Univ. Press.

[2] Cakir, M., Xhafa, F., Zhou, N., and Stahl, G. 2005. Thread-based analysis of patterns of collaborative interaction in chat. *In Proceedings of AIED.*

[3] Feng, D., Shaw, E., Kim, J., and Hovy, E.H. 2006a. An intelligent discussion-bot for answering student queries in threaded discussions. In *Proceedings of Intelligent User Interface (IUI)*, pp. 171-177

[4] Feng, D., Kim, J., Shaw, E., and Hovy E. 2006b. Towards Modeling Threaded Discussions through Ontology-based Analysis, *Proceedings of National Conference on Artificial Intelligence.*

[5] Hearst, M. A. 1994. Multi-paragraph segmentation of expository text. Proceedings of the 32th Annual Meeting of the Association for Computational Linguistics, 9-16, Las Cruces, New Mexico.

[6] Hermjakob, U., Hovy, E. H., and Lin, C. 2000. Knowledge-based question answering. *Proceedings of TREC.*

[7] Hovy, E.H., Gerber, L., Hermjakob, U., Junk, M., and Lin, C. 2000. Question answering in Webclopedia. *Proceedings of TREC.*

[8] Isbister, K. Nakanishi, H., Ishida, T., Nass, C. 2000. Helper agent: Designing an assistant for human-human interaction in a virtual meeting space. *In Proceeding of CHI...*

[9] Kim, J. & Beal, C. 2006. Turning quantity into quality: Supporting automatic assessment of on-line discussion contributions. American Educational Research Association.

[10] Koschmann, T. (Ed.).. 1996. CSCL: Theory and practice of an emerging paradigm. Hillsdale, NJ: Lawrence Erlbaum.

[11] Marom, Y. and Zukerman, I. 2005. Corpus-based Generation of Easy Help-desk Responses. Technical Report, School of Computer Science and Software Engineering, Monash University. Available at: http://www.csse.monash.edu.au/publications/2005/tr-2005-166-full.pdf.

[12] Nguyen, A. and Wobcke, W. 2005. An Agent-Based Approach to Dialogue Management in Personal Assistants. *In Proceedings of IUI..*

[13] Painter, C., Coffin, C., and Hewings, A. 2003. Impacts of Directed Tutorial Activities in Computer Conferencing: A Case Study. *Distance Education*, Vol. 24, No. 2

[14] Pallof , R.M. & Pratt, K. 1999. Building Learning Communities in Cyberspace: Effective Strategies for the Online Classroom.

[15] Pasca, M. and Harabagiu, S. 2001. High Performance Question/Answering. *In Proceedings of SIGIR-2001 pp. 366-374.*

[16] Ravi, S., Kim, J. 2007. Profiling Student Interactions in Threaded Discussions with Speech Act Classifiers, *Proceedings of AI in Education.*

[17] Salton, G. 1989. Automatic Text Processing, The Transformation,Analysis, and Retrieval of Information by Computer. Addison-Wesley, Reading, MA.

[18] Scardamalia, M., & Bereiter, C. 1996. Computer support for knowledge building communities. In T. Koschmann (Ed.), CSCL: Theory and practice of an emerging paradigm. Mahwah NJ: Erlbaum.

[19] Searle, J. 1969. *Speech Acts*. Cambridge: Cambridge Univ. Press.

[20] Shaw, E. 2005. Assessing and Scaffolding Collaborative Learning in Online Discussions. In Proceedings of the 12th International Conference on AI in Education.

[21] Soller, A., & Lesgold, A. 2003. A Computational Approach to Analyzing Online Knowledge Sharing Interaction. In Proceedings of AIED.

All in the (word) family: Using learning decomposition to estimate transfer between skills in a Reading Tutor that listens

Xiaonan ZHANG, Jack MOSTOW, Joseph E. BECK
Project LISTEN, Carnegie Mellon University, Pittsburgh, PA, USA

Abstract. In this paper, we use the method of learning decomposition to study students' mental representations of English words. Specifically, we investigate whether practice on a word transfers to similar words. We focus on the case where similar words share the same root (e.g., "dog" and "dogs"). Our data comes from Project LISTEN's Reading Tutor during the 2003—2004 school year, and includes 6,213,289 words read by 650 students. We analyze the distribution of transfer effects across students, and identify factors that predict the amount of transfer. The results support some of our hypotheses about learning, e.g., the transfer effect from practice on similar words is greater for proficient readers than for poor readers. More significant than these empirical findings, however, is the novel analytic approach to measure transfer effects.

1. Introduction

One key problem in Intelligent Tutoring Systems is student modeling, i.e., inferring an individualized model that represents the student's state of knowledge. In this paper, we focus on one particular aspect of student modeling: estimating the transfer from learning one skill to similar skills. In other words, we investigate the extent to which practice on one skill improves another skill.

The amount of transfer reflects the student's mental representation of the skill, and may grow as the representation deepens. For example, when children first learn to read, they may well just treat each word as an independent entity; as they grow to be more advanced readers, they learn to represent a word as a sequence of phonemes, or as a base form with morphological changes [1]. Gradually in this way various words became representationally connected to each other. Transfer effects also affect the benefits of practice. Again take the example of learning to read. If there's no transfer between words, reading practice on one word is unrelated to any other word, and will contribute solely to learning that word; however, if the child has acquired the alphabetic principle, knowledge of one word should also transfer to words that are alphabetically similar. As a result, the child should make faster progress than he or she did when there was no transfer.

In this paper, we study transfer in the context of reading, and investigate whether the skill gained from exposure to a word transfers to similar words. The study is carried out in the context of Project LISTEN's Reading Tutor, an intelligent tutor that helps students learn to read [2]. It displays stories sentence by sentence on the computer screen, and listens to the student read aloud. The Reading Tutor uses Automatic Speech Recognition (ASR) to analyze the student's reading, and provides help when it detects a mistake or the student gets stuck. The student can also click for help when encountering difficulties. The time-aligned output from the recognizer is logged everyday into Project LISTEN's database. The Reading Tutor also logs its interactions with the student during the entire session, such as the sentence text that the student sees on the screen, or whether the student is helped on a specific word. As a result, the data available is copious and detailed, but noisy.

Our work is related to but different from some prior work. Both Chang [3] and Koedinger et al. [4] try to elucidate students' mental representation of the target skill, but by comparing competing models that differ in skill level and granularity. In contrast, we estimate *to what degree* a skill transfers to related skills, rather than focus on which single model fits best.

The rest of the paper is organized as follows. Section 2 describes our modeling approach in detail. Section 3 presents four analyses based on it, and their results. Section 4 discusses some limitations of our current work, and possible future directions. Section 5 concludes.

2. Approach

Our goal is to measure the amount of transfer from practicing one word to learning similar words. Sections 2.1-2.6 describe successive steps to achieve this goal.

2.1 *Find a way to estimate the influence of various practice*

We approach the problem with a relatively new method—learning decomposition [5]. It extends the classic exponential learning curve (Equation 1) by taking into account the heterogeneity of different learning opportunities for a single skill, as shown in Equation 2:

$$performance = A * e^{-b*t}$$

Equation 1. Exponential model

$$performance = A * e^{-b*(t1+\beta*t2)}$$

Equation 2. Learning decomposition model of practice

In both models, A measures students' performance on the first trial, and b represents the learning rate. The two models differ only in their counting of "trials." The simple learning curve uses a single variable t that pools all learning opportunities together. In contrast, learning decomposition partitions the t trials into $t1$ trials of type 1 and $t2$ trials of type 2, so as to differentiate one type of learning opportunity from the other by keeping count of each separately. The β parameter characterizes the relative effectiveness of type 2 trials compared to type 1 trials. For example, a β value of 2 would mean that practice of type 2 is twice as valuable as practice of type 1. The basic idea of learning decomposition is to find the weight β that renders the best fitting learning curve.

Equation 1 decomposes the learning opportunities into two types. More generally, we distinguish n types of trials by replacing t with $t_1 + \beta_2 t_2 + \ldots + \beta_n t_n$. That is, we just add a counter t_i for each type i, and a free parameter β_i to represent the impact of a type i trial relative to the "baseline" type 1, where $\beta_1 = 1$ by definition. We can number the types however we like, so we can freely choose which one to designate as the baseline by calling it type 1. Any non-linear regression package can fit the model to data; we use SPSS 11.0.

2.2 *Define similarity*

We hypothesize that exposures to a word may transfer to a similar word, but what does "similar" mean here? We operationalize "similar" as "sharing the same word root." By "word root," we mean the reduced form of a word after its morphological and inflectional endings are removed. We use Porter Stemmer [6] to extract word roots. Thus "dog", "dogs", and "dogged" are similar to each other because they all share the root "dog." In effect, we treat each word as a different skill, and focus on transfer between similar words.

It should be pointed out that there are many other ways to define words as "similar" (e.g., having at least 3 letters in common), and correspondingly many other ways to define what kind of transfer to model. However, the learning decomposition method is applicable no matter how similarity is defined, which is a great advantage of this approach.

2.3 *Define the learning outcome*

We need a performance variable that measures the quality of learning. As in the earlier work [5], we measure a student's performance on a word as the time to read it, capped at 3 seconds. The word reading time is computed from the Reading Tutor's log of the time-aligned ASR output. We also assign a reading time of 3 seconds to a word if the ASR rejects the word as misread or the student clicks on it for help.

Another concern is that our performance measure should distinguish short term retention from real learning. For example, suppose that a student sees two sentences with the word "dogs" in them, and then a sentence with "dog" shortly afterwards. We can imagine that the student should read the word "dog" fairly quickly, after having seen "dogs" twice. However, is it due to transfer from practice on similar words, or merely a temporary scaffolding effect? To avoid such ambiguities, we only use students' first encounters of a word root each day to estimate their knowledge, and exclude observations that may be contaminated by recency effects. However, subsequent encounters of the word root on the same day still count as practice opportunities. For clarity, we will use the term "outcomes" to refer to the subset of observations useful in estimating student knowledge, and "exposures" to refer to the entire set of observed practice opportunities. The model predicts only the outcomes, but all the exposures contribute to the model in calculating the amount (and types) of practice preceding each outcome. In the earlier example, the first encounter of "dogs" is an outcome for skill "dogs", while the two subsequent encounters count only as exposures for the skill "dogs" and "dog."

2.4 Partition the practice space

In our model, every word is a distinct skill. For each word, any previous encounter of the same word naturally constitutes a practice opportunity. Since we hypothesize that there is transfer from similar words, previous encounters of all similar words should also count as learning opportunities. Thus for every word, there is a unique practice space made up of two separate parts: exposures to similar words, and exposures to the word itself. However, not all practice opportunities are equal, even within these two parts. As reported in [5], massed practice (reading a word more than once in a day) is generally not as effective as distributed reading (reading a word on different days) for helping students to learn the word. With this in mind, we further distinguish between massed and distributed practice, and split all the practice opportunities for a word *w* into 4 non-overlapping types:

D (Distributed same-word exposure): see word *w* for the first time that day.
M (Massed same-word exposure): see *w* again on the same day.
DS (Distributed similar-word exposure): see its root for the first time that day, in some word similar to *w*.
MS (Massed similar-word exposure): see its root again that day, in some word similar to *w*.

Let's use an example to illustrate this partition. Suppose we are modeling the skill of reading "dog." In Table 1, the leftmost 3 columns list a student's successive encounters over two days of the word "dog" itself, and words similar to "dog" such as "dogs" and "dogged." The following four columns for each type (M, D, MS and DS) show the cumulative number of exposures of that type. The rightmost column shows whether the exposure also counts as an outcome. On Monday, the student sees the word "dog" for the first time, so the exposure is of type D. Then the student sees "dog" again, this time as a massed exposure for word "dog." On the 3rd exposure, the student encounters "dogs", which is similar to "dog." Since the student has already seen a word whose root is "dog" (namely "dog") that day, the counter for type MS is incremented. On the following Tuesday, the student sees "dogged"—another word similar to "dog", which counts as type DS since it's the first encounter that day. Another thing to note here is that all 5 exposures are practice opportunities, but only Exposure 1 and Exposure 4 count as outcomes, for the words "dog" and "dogged" respectively.

Table 1. Example of computing prior exposures to "dog"

Exposure No.	Day	Target Word	D	M	DS	MS	Count as Outcome?
1	Monday	dog	1	0	0	0	Yes
2	Monday	dog	1	1	0	0	No
3	Monday	dogs	1	1	0	1	No
4	Tuesday	dogged	1	1	1	1	Yes
5	Tuesday	dog	2	1	1	1	No

2.5 Decide how to aggregate across individuals

Now that we have described the basic components of the model, a question left unresolved is how to construct the model with data from multiple students. There are at least two straightforward solutions: 1. fit all the data to a single pooled model; 2. fit an individual model for each student, and then average the parameters in some way. We decide to go with the second option, for several reasons. First, individual estimates avoid the bias of aggregating data across students, since better readers typically generate more data, and are likely to differ a lot from poor readers in both initial knowledge and learning rate. Second, building student-specific models affords us flexibility in how we combine the models later in the analysis stage. For example, we may want to group students to make planned inter-group comparisons, as in [7], or cluster them after the fact.

However, partitioning exposures into different types already gives rise to a sparse data problem for rare types; estimates for individual students are even sparser, as reflected in the standard error output by SPSS for each parameter estimate. The obvious symptom of sparse data is an outlandish estimate or a large standard error. We deal with this issue by using the median (instead of mean) of the per-student estimates so as to deemphasize outliers. A less obvious symptom is a standard error of 0.0, which occurs (fortunately just occasionally) when the non-linear regression leaves a model parameter at its initial value due to lack of data. We deal with this issue by excluding these unchanged parameter estimates before computing the median.

2.6 *Account for additional influences*

Other factors also affect word reading time. First is word length: longer words generally take longer to read. Another factor is whether a student gets help on a word. We've adjusted reading time to 3 seconds for helped words, but we haven't taken into account the influence of previous help on that word or similar words. The number of times a student has gotten help on a word also reflects the difficulty of the word for the student. Incorporating all three factors leads to the full model shown in Equation 3.

$$performance = L * word\_length + A * e^{-b*(D+\beta_M*M+\beta_{DS}*DS+\beta_{MS}*MS+H*\#help\ on\ the\ word+HS*\#help\ on\ similar\ words)}$$

Equation 3. Full learning decomposition model for estimating transfer

In this model, we add the term word_length (number of letters in a word), and use L to weigh its relative impact. We also keep count of how many times the student requests help on the word as well as on similar words. Note that we only count student-initiated help, because the Reading Tutor could initiate help under a lot of (possibly inappropriate) situations so the influence of tutor help may be too complex to analyze. We set the weight of type D exposure to 1 so as to use it as a baseline. The three free parameters β_M, β_{MS}, and β_{DS}, reflect the impacts of exposures of type M, MS, and DS relative to the exposures of type D. The central task of this paper is to estimate and analyze the parameters β_M, β_{MS}, and β_{DS}.

3. Studies and Results

The data for this paper was collected by Project LISTEN's Reading Tutor during the school year 2003-2004, including 6,213,289 observations of 650 students' word encounters. To ensure that we measure learning rather than recency effects, the model predicts only outcomes as defined earlier. This filter reduces the data size to 2,711,618, though the other exposures still contribute to counting different types of practice. We further filter the data on several other conditions: 1. Exclude encounters of the 50 most frequent words. 2. Exclude all word encounters in stories the student has read before; 3. Consider only a student's first 20 exposures to a word (i.e., $M+D \le 20$). 4. Exclude cases where a student's exposures to similar words (either massed or distributed) exceed 20 (i.e., we want $MS \le 20$ and $DS \le 20$). The rationale for these constraints is that under any one of the four conditions, the student should have gained too much familiarity with the word to exhibit any learning effect. 5. Exclude students who read fewer than 20 words in the Reading Tutor, because 20 data points is far from enough to give a reliable student-specific model. In fact, running non-linear regression with fewer than 20 data points for this model causes SPSS to abort the regression procedure. 6. Exclude students without paper test scores needed for some of our analyses. After filtering, we have 860,517 cases (trial outcomes), including 10,357 distinct words read by 346 students from Grade 1 to Grade 5.

We have some initial hypotheses before carrying out the experiment: 1. Students do benefit from exposures to similar words, perhaps not as much as they do from exposures to the word itself. 2. The transfer effect is stronger for students with higher initial reading skills. 3. Transfer is also greater for students with larger gains over the school year. We test our hypotheses in a series of analyses.

3.1 *Are there transfer effects from practice on similar words?*

We build a model in the form of Equation 3 for each student. Due to sparse data problems, the model construction fails for 18 students (one or more parameters aren't updated at all), so we take the medians only of the remaining 328 models as the overall parameter estimates. For each parameter, we also derive its two-sided 95% confidence interval using a non-parametric bootstrap test [8], by bootstrap sampling 10,000 times from the original estimates. The sample size is equal to that of the original set (328). Table 2 shows the result. The confidence intervals for different parameters vary in tightness, largely because some parameters are estimated from more data than others. The bottom row of the table shows the percentage of outcomes that contributes to each parameter estimate. For example, this percentage is 100% for L, because every word has a length, but only 7% for H, because students received help on only 7% of the words.

Table 2. Overall median parameter estimates (with 95% Confidence Interval)

Parameter	L	A	B	β_M	β_{MS}	β_{DS}	H	HS
Estimate	0.133 ±0.004	0.822 ±0.063	-0.082 ±0.014	0.171 ±0.069	0.063 ±0.114	0.551 ±0.126	-1.286 ±0.196	-0.665 ±0.257
% of outcomes	100	100	55.13	26.38	10.41	18.51	7.33	3.08

Overall, the model seems very reasonable. The positive value for L means that the reading time should grow with word length, on average 0.13 seconds per letter. The positive A represents students' initial reading time of a word, minus the approximated reading time for words of that length. The number seems a bit high at first look, but it's because we assign a value of 3 seconds to some words. Actually, in our dataset, the average reading time (after adjustments as described in section 2.3) for students' first word root encounters is 1.53 seconds. If we multiply estimated L by the average length of a word (5.5 letters in our data), and add the result (0.7315) to A, we get 1.5535, which is fairly close to the actual value of 1.53. The negative B shows the general trend that reading time decreases with more practice. Both H and HS are significantly less than 0. The negative coefficients for help presumably occur because help indicates that the student is having difficulty with the word, not that help is hurting the student.

Our result shows that M is significantly less than 1, which is consistent with the finding in [5] that massed practice is worth less than distributed practice. Practice of type MS and DS are also less effective than distributed reading of the same word, as expected. However, the impact of DS is quite notable ($\beta_{DS} > 0$ with $p < 0.0001$). In fact, β_{DS} is even significantly larger than β_M ($p < 0.0001$), which means that distributed practice of a similar word is more effective for learning a word than seeing the word itself again on the same day. In contrast, the impact of practice of type MS is negligible: the 95% confidence interval for MS straddles both sides of 0.

3.2 *Higher reading proficiency, more transfer?*

To investigate whether there is a positive relationship between initial reading skill and the amount of transfer, we divide all the students into 3 equal-sized bins: low proficiency, medium proficiency and high proficiency, according to their grade-equivalent pretest score on the Woodcock Reading Mastery Word Identification (WI) subtest [9]. Low proficiency students have a test score of less than 1.6. High proficiency students have scores greater than 2.5. Medium proficiency students are those with a test score in between. Then we calculate the median parameters for each subgroup of students. The result is shown in Table 3.

Table 3. Median parameter estimates (with 95% confidence interval), disaggregated by student proficiency

Bin	β_M	β_{MS}	β_{DS}
Low Proficiency	0.275 ± 0.162	0.102 ± 0.232	0.184 ± 0.315
Medium Proficiency	0.245 ± 0.083	-0.044 ± 0.133	0.655 ± 0.197
High Proficiency	0.002 ± 0.088	0.189 ± 0.208	0.947 ± 0.431

Do the relative values of the different types of practice follow the same pattern within each bin as the general estimates in Table 2? Mostly yes. The effect of MS stays trivial, and all the β_{DS}'s are larger than 0. Yet in some bins there are discrepancies from the general pattern. In particular, for the high proficiency students, the impact of massed practice on a word is not different from 0, significantly less than the impact of the same practice for the other two lower proficiency groups ($p < 0.05$). This trend of decreasing benefits from massed practice for more advanced readers is also reported in [5]. At the same time, for the same group, we find no significant difference between β_{DS} and the base line β_D ($\beta_D = 1$). The high values of β_{DS} indicates that proficient readers benefit almost as much from practice on similar words as they do from the word itself, which suggests that they may be sensitive to the morphemic structure of a word.

How do the transfer effects change as proficiency increases? Examining the column for β_{DS} values gives a suggestive picture. The effectiveness of distributed exposures to similar words increases from low proficiency to high proficiency students. Bootstrapping tests show that β_{DS} of high proficiency students is significantly larger than that of low proficiency students. The result supports our initial hypothesis that transfer increases with the growth of students' reading skills.

3.3 *Greater learning gains, more transfer?*

This analysis is similar to that in section 0. But instead of grouping students by their proficiency, we disaggregate the students by their improvement over the school year. We quantify students' improvement as the difference between their pretest and posttest grade-equivalent scores on the WI test. Again, the students are grouped into 3 bins: low gain, medium gain, and high gain. WI scores of low gain students increase no more than 0.6 from pretest to posttest. High gain students improve by more than 1.1 in WI score. Medium gain students lie in between. Table 4 gives the medians and 95% confidence intervals of the parameter estimates for each bin.

Table 4. Median parameter estimates (with 95% Confidence Interval) disaggregated by student gain

Bin	β_M	β_{MS}	β_{DS}
Low Gain	0.214 ± 0.193	0.076 ± 0.181	0.476 ± 0.218
Medium Gain	0.254 ± 0.165	0.086 ± 0.171	0.637 ± 0.212
High Gain	0.134 ± 0.081	0.056 ± 0.190	0.549 ± 0.211

Contrary to our hypothesis, none of the differences between bins turn out to be significant. This null result is a bit surprising and disappointing. The degree of transfer from similar words should be closely correlated with the students' awareness of the morphemic structure of words. In many previous studies, it has been shown that morphological awareness plays an important role in both vocabulary acquisition [10] and long-term reading development [11]. Why doesn't this trend show up in our analysis? There could be many reasons. For one thing, we are using the noisy test score difference to measure students' improvement. It is quite possible that a student did particularly well or poorly in one test, so test scores are imperfect reflections of students' ability. When we use only one test score, as in section 0, such noise should be washed out by aggregating over many students. However, when we use pre- to posttest gain as the measure, the variance in the two test scores combines, which results in even more noise: the difference between two noisy estimates is noisier.

3.4 Which student characteristics predict transfer?

In both section 3.2 and section 3.3, our studies are driven by our prior expectations. In this study, we adopt a different, bottom up approach to study which factors could affect the amount of transfer. Specifically, we build a linear regression model with various properties of a student as independents, to predict the estimated DS parameter for the student. We use DS as the dependent rather than MS, because the transfer effect from practice of type DS is significantly larger.

However, as mentioned before, individual estimates are extremely unreliable. In the case of DS, the standard deviation of the 328 students' estimates is 1600, which makes it unreasonable to model the exact value of DS directly. To get around this problem, we model the *rank* of DS for a student instead. Since there are 328 sets of parameter estimates, our dependent variable ranges from 1 to 328. We use three predictors. One is the students' grade (1—5). Another is the student's *grade-relative* WI pretest score, computed as WI score minus the student's grade, to eliminate the strong correlation between grade and WI test score. We use both grade and grade-relative WI, instead of just grade-equivalent WI. We treat grade-relative WI as a proxy for the student's relative proficiency at his or her grade level, while grade adds extra information about the student's experience in reading. The third predictor is the student's pretest score (20—80 in our dataset) on the ERAS measure of reading motivation [12].

The resulting model is barely significant ($p \approx 0.05$), with an R^2 of only 0.024. Among the three independents, grade is most predictive and reliable ($p \leq 0.05$), and grade relative WI score is borderline reliable ($p < 0.1$). Coefficients for all the predictors are positive, showing that transfer tends to increase with higher grade, higher WI test score, and greater interest in reading.

Considering that individual DS estimates are very noisy, the small R^2 is not surprising. The problem is that a lot of the variance is caused by random noise, which is essentially unexplainable, so R^2 is not a viable measure here. Therefore we take a different approach to evaluating our model, by looking at whether it separates the students into interesting groups.

Accordingly, we split all the students to 2 bins based on the standardized prediction of the model, which ranges from -1 to +1. We rank DS in increasing order, so the higher the rank, the larger the DS. Students in bin 0 (low transfer) have a standardized DS rank of less than 0. The rest of the students are assigned to bin 1 (high transfer). We adopt this 50-50 split because it's a reasonable and natural starting place in the absence of more information about the dividing point between low and high transfer. Then we compare the median of estimates for other parameters of the two groups of students, as shown in Table 4. For comparison, we also bin students into two groups by their original DS estimates, and compute the medians of parameter estimates for each group, given in Table 5.

Table 4. Comparison of median parameter estimates for two bins based on model prediction

Bin	L	A	B	M	H	HS	MS	DS
Low Transfer	0.132	1.067	-0.045	0.265	-1.311	-0.528	-0.004	0.402
High Transfer	0.136	0.464	-0.136	0.069	-1.232	-1.003	0.116	0.875
Sig. of Difference	0.2165	<0.0001	<0.0001	0.0165	0.4486	0.0044	0.245	<0.0001

Table 5. Comparison of median parameter estimates for bins based on original DS estimates

Bin	L	A	B	M	H	HS	MS	DS
Low Transfer	0.129	0.894	-0.072	0.163	-1.454	-0.500	0.282	-0.211
High Transfer	0.137	0.720	-0.091	0.184	-1.164	-1.590	-0.214	1.460
Sig. of Difference	0.066	0.0105	0.1179	0.3789	0.1455	0.0001	<0.0001	<0.0001

The result shows that the model is doing a decent job at separating the data, even better than the original DS does. The contrasts between the two bins in Table 4 are sharp (more so than in Table 5): students predicted to have low transfer have slower initial reading time (A = 1.07 vs. 0.464), learn slower (B = -0.045 vs. -0.136), benefit more from massed practice (M = .265 vs. .069), and transfer less (DS =.402 vs. .875). Moreover, all these differences are statistically significant. Therefore even though the model's R^2 is low and barely significant, the model separates DS much better than we expected. The clearer separation may be because our model smoothes out some random noise in the original DS estimates.

4. Limitations and Future Work

Though our learning decomposition model has refined the trial counts in the exponential model, it's still much simplified with respect to the actual complex cognitive process of learning. For example, the model assumes that the parameters are static for a specific student, while in reality the learning rate as well as the relative effectiveness of different types of practice will change over time with the development of the students' learning strategies. By simply partitioning all the practice on a word into non-overlapping types, our model also overlooks the interaction between different types of practice, such as order effects. We could introduce more complex terms into the model, but at the risk of reducing the data for each term and further increasing the variance in parameter estimates. As computationally efficient as it seems now (it takes only about 10 minutes to fit the nearly one million outcomes), the training process guarantees only a local optimum, and there is usually large error in parameter estimation for each individual model (though the median point is relatively stable compared to individual estimates). The model fit is rather low, only about 11.2% on average, similar to the number reported in the original learning decomposition model [5], hence not dismaying. Though we could argue that fitting individual data points tends to result in a poorer model fit than fitting a curve to aggregate performance, it's still a priority to improve the fit and robustness of the model.

The fact that our data comes from noisy ASR output also makes the estimates error-prone. In a test on data collected from the 2001-2002 school year, the ASR detected only about 25% of students' misreadings, and misclassified 4% of correctly read words as misread [13]. The best thing we could hope for, is that the ASR errors are random, and their effect on our measure of reading time would be diminished by aggregating over individual estimates so as not to blur the overall pattern.

Our future work will focus on improving the accuracy of the model, perhaps by modifying the form of the model to better fit the data. One possible improvement we are considering is to add another term that accounts for a student's total text exposures outside the Reading Tutor. Then we'll have a more accurate count of the student's prior word exposures, by adding this outside-tutor exposure estimate to the original within-tutor count. Another future direction is to extend our current study to other types of transfer. For example, we are analyzing transfer between rhyming words (i.e., words that share the same rime, such as "cat" and "hat"). We may also take into account the context effect (e.g., seeing a word in a novel vs. familiar context) in our model. For instance, we could use our approach to analyze whether it is better to read "the cat in the hat" twice or see "cat" the second time in some other sentence.

5. Conclusion

The contributions of this paper lie both in its methodology and in its results. There have been many studies in transfer of learning [14, 15], the paradigm of which is to carry out carefully designed controlled experiments. The method we describe is a simple yet potentially powerful alternative, and could be applied to many other intelligent tutoring systems. Our model exploits the massive and detailed (albeit noisy) machine observations of students' activities to make quantitative inference about their cognitive process. It is easily adaptable to changes in our underlying assumptions about the form of transfer. For example, we could look into transfer from rhyming words simply by making minor changes to Equation 3 and retraining the model. Moreover, so far as we know, previous studies of transfer at the individual word level [e.g., 14] looked only at transfer to previously unseen words. In contrast, our large data set lets us look at transfer to a word across multiple instances.

We also get some interesting and plausible results regarding transfer from practicing a word to learning similar words. We find that there is significant transfer from distributed practice on a word to similar words, but the effect becomes negligible when the student has already seen a word with the same root that day. Our hypothesis about the positive relationship between reading proficiency and the ability to learn from word roots is also supported by the result. Such findings could assist us in designing or evaluating fluency practice texts by predicting their effects for students at different levels of proficiency. For example, our result suggests that proficient readers learn significantly more than less proficient readers learn about a particular word from exposures to similar words. So for a less proficient reader learning to read a word fluently by practicing it on multiple occasions, it's important to practice the word itself each time, whereas for more proficient readers, it's almost as good to practice different words with the same root. Also, the less proficient reader derives some benefit from seeing a word repeated on the same day, though not as much as from seeing it on different days. In contrast, more proficient readers gain nothing from same-day repetitions, so text designed to improve their fluency should minimize unnecessary word repetition.

Another conclusion is that learning decomposition is a useful technique for investigating students' representation of knowledge. In particular, it provides a way to model transfer among similar skills without assuming a shared more general underlying skill, as previous approaches do [3, 4]. Our particular example here assumes such a skill (the shared root), but the model doesn't have to. For instance, we could have defined "similar" as "90% identical," or as "same except for 1 letter" (or *k* letters). This type of similarity doesn't correspond to sharing a single more general skill. Thus a model with explicit transfer has more (or different) expressive power than one based on shared abstract skills, thereby enabling us to gain more insight into a hidden cognitive process such as transfer of reading skill.

References

1. Gough, P.B. The Beginning of Decoding. *Reading and Writing: An Interdisciplinary Journal*, 1993. *5*(2): p. 181-192.
2. Mostow, J., G. Aist, P. Burkhead, A. Corbett, A. Cuneo, S. Eitelman, C. Huang, B. Junker, M.B. Sklar, and B. Tobin. Evaluation of an automated Reading Tutor that listens: Comparison to human tutoring and classroom instruction. *Journal of Educational Computing Research*, 2003. *29*(1): p. 61-117.
3. Chang, K.-m., J.E. Beck, J. Mostow, and A. Corbett. Using speech recognition to evaluate two student models for a reading tutor. *Proceedings of the AIED 05 Workshop on Student Modeling for Language Tutors, 12th International Conference on Artificial Intelligence in Education*, 12-21. 2005. Amsterdam.
4. Koedinger, K.R. and S. Mathan. Distinguishing Qualitatively Different Kinds of Learning Using Log Files and Learning Curves. *ITS2004 Workshop on Analyzing Student-Tutor Interaction Logs to Improve Educational Outcomes*, 39-46. 2004. Maceio, Brazil.
5. Beck, J.E. Using learning decomposition to analyze student fluency development. *ITS2006 Educational Data Mining Workshop* 2006. Jhongli, Taiwan.
6. Porter, M.F. An algorithm for suffix stripping. *Program*, 1980. *14*(3): p. 130-137.
7. Beck, J.E. Does learner control affect learning? *Proceedings of the 13th International Conference on Artificial Intelligence in Education* 2007. Los Angeles, CA.
8. Cohen, P.R. *Empirical Methods for Artificial Intelligence*. 1995, Cambridge, Massachusetts: MIT Press. 405.
9. Woodcock, R.W. *Woodcock Reading Mastery Tests - Revised (WRMT-R/NU)*. 1998, Circle Pines, Minnesota: American Guidance Service.
10. McBride-Chang, C., R.K. Wagner, A. Muse, B.W.-Y. Chow, and H. Shu. The role of morphological awareness in children's vocabulary acquisition in English. *Applied Psycholinguistics*, 2005. *26*: p. 415-435.
11. Deacon, S.H. and J.R. Kirby. Morphological awareness: Just "more phonological"? The roles of morphological and phonological awareness in reading development. *Applied Psycholinguistics*, 2004. *25*: p. 223-238.
12. McKenna, M.C., D.J. Kear, and R.A. Ellsworth. Children's attitudes toward reading: a national survey. *Reading Research Quarterly*, 1995. *30*: p. 934-956.
13. Banerjee, S., J.E. Beck, and J. Mostow. Evaluating the Effect of Predicting Oral Reading Miscues. *Proc. 8th European Conference on Speech Communication and Technology (Eurospeech 2003)*, 3165-3168. 2003. Geneva, Switzerland.
14. Benson, N.J., M.W. Lovett, and C.L. Kroeber. Training and Transfer-of-Learning Effects in Disabled and Normal Readers: Evidence of Specific Deficits. *Journal of Experimental Child Psychology*, 1997. *64*(3): p. 343-366.
15. Taguchi, E. and G.J. Gorsuch. Transfer Effects of Repeated EFL Reading on Reading New Passages: A Preliminary Investigation. *Reading in a Foreign Language*, 2002. *14*(1).