Int J Software Informatics, Vol.3, No.1, March 2009, pp. 29–44
International Journal of Software and Informatics, ISSN 1673-7288

E-mail: ijsi@iscas.ac.cn
http://www.ijsi.org
Tel: +86-10-62661040

Temporal Data Mining for Educational Applications*

Paul R. Cohen and Carole R. Beal

(University of Arizona, USA, cohen@cs.arizona.edu, crbeal@email.arizona.edu)

Abstract Intelligent tutoring systems (ITSs) acquire rich data about students' behavior during learning; data mining techniques can help to describe, interpret and predict student behavior, and to evaluate progress in relation to learning outcomes. This paper surveys a variety of data mining techniques for analyzing how students interact with ITSs, including methods for handling hidden state variables, and for testing hypotheses. To illustrate these methods we draw on data from two ITSs for math instruction. Educational datasets provide new challenges to the data mining community, including inducing action patterns, designing distance metrics, and inferring unobservable states associated with learning.

Key words: educational data mining; student modeling

Cohen PR, Beal CR. Temporal data mining for educational applications. *Int J Software Informatics*, **2009, 3(1): 29–44. http://www.ijsi.org/1673-7288/3/29.htm**

1 Introduction

Teachers, school administrators and parents have always wanted to know how their students are doing in the classroom. Recently, interest in tracking student learning has grown dramatically due to increased emphasis on accountability in educational settings. For example, in the United States, educators are in urgent need of accessible and meaningful information about how students are learning, in order to meet annual progress report requirements resulting from the 2002 "No Child Left Behind" act[1]. Schools face significant pressure to improve learning outcomes, yet improvement depends on the ability to identify in the short-term those student behaviors that are likely to be unproductive in the long term.

Intelligent tutoring systems (ITSs) potentially address the problem of assessing and tracking students and the problem of improving learning outcomes. It is possible to record keystroke-by-keystroke information as students use ITSs, and to process it quickly to provide teachers up-to-the-minute assessments of their students' performance, rather than waiting for weekly tests or annual end-of-year assessments[6]. ITSs have been shown to provide effective instruction, with some results showing improvement of 20-25% on pre- and post-test measures[2, 3]. Yet at the same time, there has been growing concern about the tendency of students to use such systems ineffectively. Indeed, students may actively avoid effort by adopting behavioral strategies

* This effort is sponsored by the the U.S. National Science Foundation and the U.S. Institute of Education Sciences.
Corresponding author: Paul R. Cohen, Email: cohen@cs.arizona.edu
Manuscript received 2009-02-10; revised 2009-07-13; accepted 2009-07-15.

that allow them to avoid learning. For example, a student may deliberately enter incorrect answers to elicit hints and, eventually, the correct answer from the ITS[4]. Thus, although ITS instruction is beneficial, its effectiveness might be increased if maladaptive student behaviors could be identified.

Thus, data mining techiques are essential to both assessing students' performance and enhancing the effectiveness of their efforts with ITSs. Only recently have researchers begun to examine ITS data in detail[5]. This paper surveys several techniques we developed to model the behaviors of students using two mathematics ITSs. One can look at ITS data on several scales, from characteristics of individual problems (Section 2) to sequences of problems for individual students (Section 3) to samples of entire tutoring sessions for groups of students (Sections 4,5). At each scale one can both extract descriptive information (e.g., the average time required to solve a problem or the distribution of the number of problems in a tutoring session), and estimate the structure and parameters of models (e.g., Markov models of actions or long-run patterns of attention during tutoring sessions). One also can cluster students based on parameters of models (Section 4.1) and test hypotheses about groups of students (Section 5). This paper is primarily concerned with models of student behavior that can be used to improve the experiences students have with tutoring systems. Thus, we model the amount of time students are willing to spend on problems and how it changes over the course of a session, and we model unobservable factors such as engagement, using hidden variable models (Section 4.1).

The techniques in this paper were developed to analyze data from two mathematics ITSs: Wayang Outpost is an ITS for secondary school math; the dataset includes two samples of students (MA and CA). The MA dataset was weighted with a larger number of low achieving students, whereas the CA sample included a full range of student achievement levels. The second ITS, AnimalWatch, is for middle school math. More information about these tutoring systems may be viewed at `http://www.cs.arizona.edu/~beal/projects/`.

The basic pattern of interactions with both the Wayang and AnimalWatch systems is this: The ITS selects a problem that it thinks will advance the student's learning. The student solves the problem and enters the answer, or selects an answer from a multiple-choice menu. The ITS uses the answer to update its model of the student's competence. If the answer is wrong, the ITS presents or recommends hints sequentially until the student gets the answer correct. The data include characteristics of problems (e.g., expected and empirical difficulty, and the topics and skills they exercise), characteristics of hints (e.g., whether they are multimedia or text hints), and characteristics of the student's performance (e.g., the number of hints they require and the time they devote to each).

2 Actions and Classifiers

The smallest scale at which data mining has a role is that of the individual problem-solving interactions, of which the most interesting involve wrong answers and hints. Because ITSs provide customized instruction to students, every interaction is literally unique. Said differently, the number of combinations of student attributes, versions of the ITS, problem topics, orders of presentation, selected hints, and latencies or student responses, is so large that some abstraction is necessary if we are to find

useful regularities in data. Consequently, we build classifiers of individual actions taken by students on single problems. These actions are different for Wayang Outpost and AnimalWatch, and thus required different pattern classifiers.

We stress that the classifiers we are about to discuss were designed by us, not learned from data. The automatic induction of classifiers for problems and hints is an open challenge for data mining, as discussed in Section 6, below. Because the classification of actions is so fundamental to all the analyses in the rest of this paper, we deal with it at some length.

Wayang Outpost, our secondary school math ITS, presented problems that included a figure, table or other graphic, the question or equation to be solved, and five answer options. The student received feedback (correct, incorrect) by clicking on an answer option. The student could also click on a "hint" icon to view an ordered series of audio and animation hints. The student could choose an answer at any point, or continue to view hints until the solution was displayed. Wayang Outpost thus required the student to request help.

The problems presented by the middle school math tutor, AnimalWatch, each included a text introduction with information necessary to solve the problem, the problem question or equation, an illustration, table or graphic, and an answer box. Incorrect answers triggered immediate feedback. Feedback hints were ordered: Hint 1 provided text feedback (correct, incorrect); Hint 2 offered an "operation" text hint (e.g., "Are you sure you are subtracting?"). Subsequent errors on the problem triggered Hint 3, which offered a multimedia worked example or interactive hint that could include multiple steps (e.g., dragging one-bars into tens-units on the screen to compose the additive quantity). The student had to request multimedia help by clicking on the "hint" icon.

We defined action patterns that might be associated with effective and ineffective learning behaviors. They are patterns (as opposed to atomic actions) because they include sequence and latency information, and are rough interpretations of the student's intentions. For example, for Wayang Outpost, a student who chooses multiple incorrect answers before the correct answer might be trying to correct his or her own errors, but might also be simply guessing. We use the time between clicks on the answers to distinguish these interpretations, e.g., inter-click intervals of less than 5 seconds signal guessing. Similarly, if the latency between presenting a problem and the student's first answer is less than ten seconds, the student may be guessing. These latencies were based on average times for the highest-performing students. Specifically, if an academically-proficient student required at least 10 seconds to read a math problem before solving it, it is unlikely that the average student would arrive at the correct answer in less than 10 seconds after the problem has loaded. Similarly, if a proficient student takes more than 10 seconds between the choice of an incorrect answer and the subsequent selection of the right answer, inter-click intervals of under 10 seconds are likely to mean that the student is guessing.

We defined five patterns associated with the high school math ITS: 1) Skipping the problem, 2) Guessing/abusing help, 3) Solving problem independently but with errors, 4) Learning by using multimedia help, 5) Solving problem independently and accurately. We defined a larger number of action patterns — nine in all — for the middle school ITS to differentiate the number of hints received by the student.

3 Student Behavior Within Problems

In both AnimalWatch and Wayang Outpost students get hints when they make mistakes, and in some cases they get a sequence of help interactions within a single problem. We call these interactions hints. Students respond differently to sequences of hints, so it is important to mine these sequences to extract patterns of student behavior. In particular, it is useful to learn if students actually pay attention to the hints, and how different versions of the ITS might influence whether students attend to the hints.

In AnimalWatch, the hints ranged from textual reminders and questions (e.g., "no, try again" and "are you using the right operator?") to sophisticated multimedia help that guides the student through the process of solving the problem. In one study, we compared students' attention to versions of AnimalWatch that varied in the availability of hints and the pace of instruction.

We asked the following question of AnimalWatch data: For every problem that required at least one hint, what fraction of the total time required to solve the problem was spent attending to each hint? The results are shown in Fig.1. The horizontal axis shows the proportion of the total time required to solve a problem. The vertical axis is organized by two variables: Hint number has six levels, for $1, 2, 3, 4, 5, > 5$ hints, respectively. Group is either Heuristic, denoting a version of AnimalWatch that provided up to two textual hints and then multimedia help on all subsequent hints; ML, denoting a version that followed the same hint schedule but "pushed" problems of increasing difficulty more aggressively than Heuristic; or Text, which provided no help at all besides the textual feedback that an answer was right or wrong.

The left and right lines in Fig.1 represent mean and median proportions of problem-solving time, respectively. Some aspects of Fig.1 were anticipated but some were surprising. We expected the proportion of time on the third hint to be high because this is where multimedia help is provided for the first time. We had no such expectation for the Text group because they do not receive multimedia help. We were surprised that the ML group spent so little time on the multimedia hints, in contrast to the Heuristic group. In fact, they spent most time on the first hint (which was "wrong, try again") and roughly the same amount of time as the Heuristic group on the second ("did you use the appropriate operator"), but then, unlike the Heuristic group, they spent even less time attending to the third hint. Thus, the detailed analysis of patterns in students' hint usage revealed that trying to accelerate the pace of students' learning had the unintended effect of reducing their attention to the multimedia hints.

3.1 Rule mining for action pattern sequences

ITS researchers are also interested in learning more about when students are most likely to spend time on a problem versus deciding to guess. That is, can we look at students' actions over time and make predictions about their future behavior? Such insights might help us to improve the design of pedagogical strategies used by an ITS to maintain students' interest in the learning activity.

Each student's data record included an ordered sequence of action patterns (as discussed in Section 2) representing her or his behavior on the math problems over the course of the tutoring session. Each action pattern was coded as a letter, and

the sequence of problems solved by a student could be coded as a sequence of letters such as $[AAECBD]$. Can we find predictive rules in these sequences? For example, if a student has guessed on two problems in a row, can we infer anything about what the student will do on the current problem? Predictive rules will have the form $A_{n-j} \ldots A_n \rightarrow A_{n+1}$, where the left hand side is a subsequence of actions and the right hand side is the one we wish to predict. Can we find rules of this form, and if so, what is the best value of j? If we view the generation of action patterns as a Markov process then j is the order of the Markov chain. Or, j might be allowed to vary, as described below.

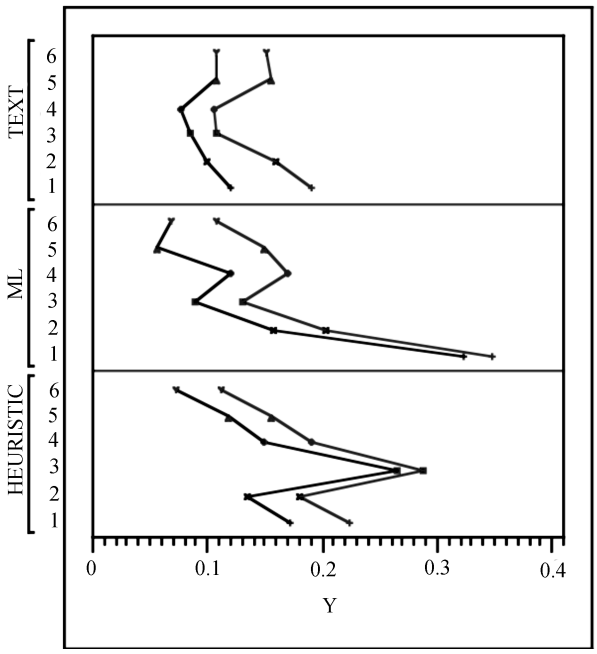

Figure 1. Median and mean proportion of problem solving time in hint sequences for three versions of AnimalWatch ITS

First, a word about estimating the accuracy of predictions: In the case of Wayang Outpost, students can exhibit one of five action patterns on a math problem. However, these patterns are not equally likely so the default accuracy of predictions is not 1/5. We define the default accuracy to be the probability of the majority class action pattern because, lacking any other information, one's best guess is that the student's next action will be the majority class action. This turns out to be guess/abuse help. With 2028 instances in the combined corpus of MA and CA students, this is the most common of 7316 actions. Thus, the default accuracy is (2028 / 7316) = .277. This is the accuracy level we have to beat with any predictor.

3.2 Accuracy of prediction rules

We developed a rule-mining algorithm to find rules of the form $A_{n-j} \ldots A_n \rightarrow A_{n+1}$. This algorithm, called Very Predictive Ngrams (VPN) finds rules with different values of j (unlike a Markov chain predictor of fixed order) so as to maximize predic-

tiveness for a given number of rules[7]. The rules found by VPN predict the action that maximizes the conditional probability $Pr(A_{n+1}|A_{n-j} \dots A_n)$. The algorithm searches iteratively for K such rules. At each iteration the algorithm greedily adds the rule that will most reduce the errors in predictions. Let $A_{n-j-1}, A_{n-j} \dots A_n \rightarrow A_{n+1}$ be a child of the parent rule $A_{n-j} \dots A_n \rightarrow A_{n+1}$. The VPN algorithm is greedy in the sense that it will not necessary find the child, even if it has better prediction accuracy than the parent, if it first finds another rule with better prediction accuracy than the parent. Nevertheless, VPN finds relatively small sets of rules that have prediction accuracies comparable to exhaustive sets of all possible rules of a given length (i.e., to Markov chain predictions of fixed order j).

The performance of VPN on Wayang Outpost action pattern sequences is shown in Fig.2. The horizontal axis is K, the number of ngrams found by VPN. The curve that drops from left to right is the error rate associated with each corpus of K rules. The curve that rises from left to right is the average length of a rule in the corpus (i.e., the average value of j).

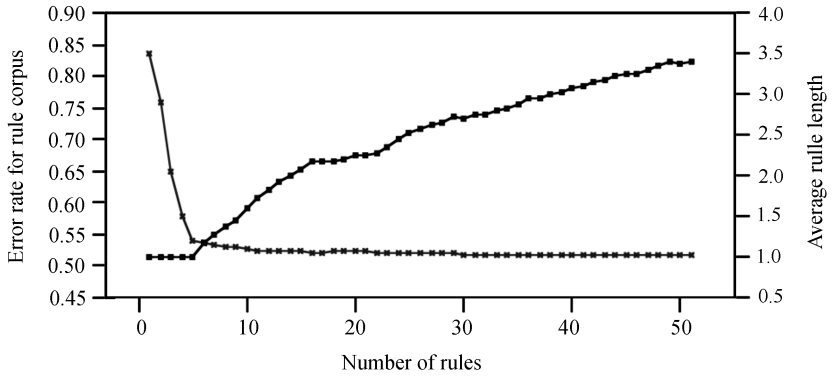

Figure 2. Ngrams for Wayang Outpost action pattern sequences

VPN immediately finds five rules of length 1. With each, the error rate drops, going from roughly 85% to roughly 55%. After that, the error rate decreases very slowly, almost imperceptibly, while the average rule length increases. For these data, it is hard to get much better accuracy than is provided by a Markov chain predictor of order one. (For comparison purposes, an exhaustive corpus of 554 rules had an error rate of 0.506, while VPN 50 rules have an error rate of 0.516.) The lesson for designers of ITSs seems to be that the action pattern on the next problem depends to a very great extent on the previous pattern and not much on earlier patterns.

4 Session-Scale Patterns

At the scale of short subsequences of action patterns, the behavior of students appears to be nearly Markov, but at larger scales, useful patterns emerge. We will focus on changes in the amount of time students spend on problems and hints over the course of a session with the AnimalWatch ITS.

The algorithm for finding these larger-scale patterns is simply to track the proportions of mutually exclusive and exhaustive attributes of problems over time for each student, and then average these proportions over all students within a group.

To illustrate, consider the time students spend on problems. Although this is a continuous variable, it can be binned into mutually exclusive and exhaustive ranges. Each problem falls into one bin, and we can track the proportions of all problems that fall into each bin over all tutoring sessions. The results, smoothed with a simple exponential smoother, are shown in Fig.3. The three trajectories correspond to spending less than 15 seconds on a problem (the trajectory that starts lowest), spending 15-35 seconds (the trajectory that remains roughly constant), and spending more than 35 seconds (the trajectory that starts highest). The probability of each of these is plotted on the vertical axis. The horizontal axis represents the number of problems the student has seen.

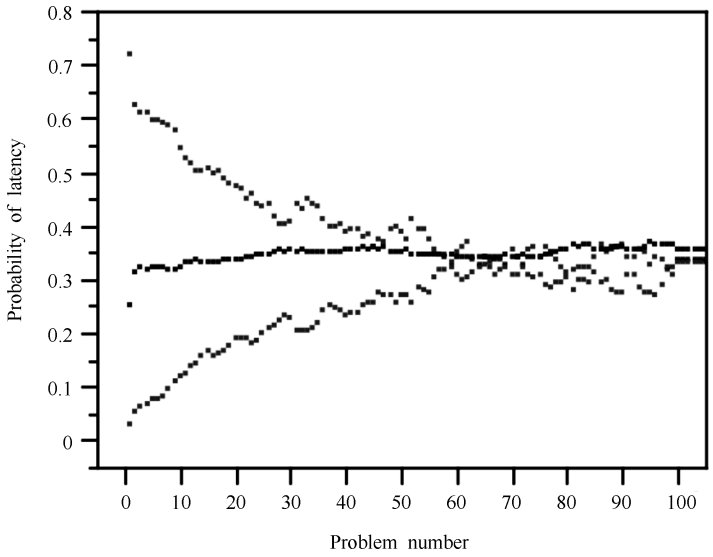

Figure 3. Time in problem across problems

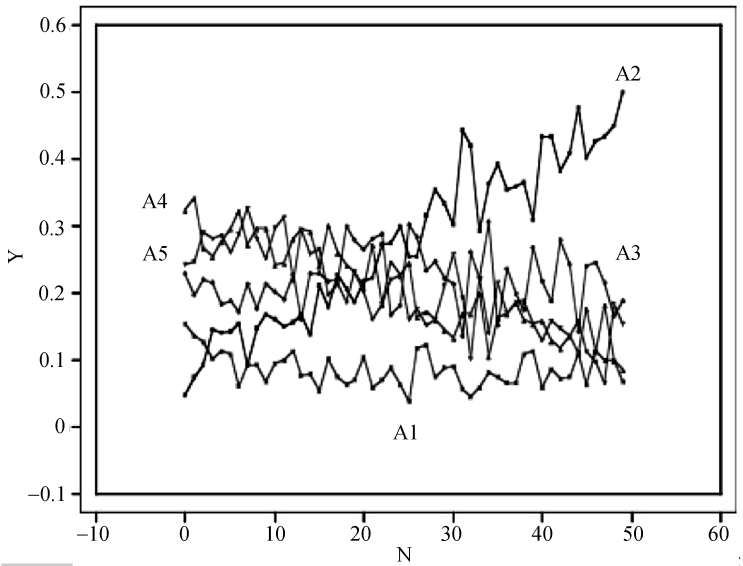

Figure 4. Action patterns across problems

Note that the mean probability (over all students' sessions) of spending more than 35 seconds on a problem falls sharply as the number of problems increases, just as the average probability of spending fewer than 15 seconds increases. Interestingly, the average probability of spending 15-35 seconds on a problem remains roughly constant throughout sessions.

The amount of time that a student spends on a problem is to a large extent under the student's control. Thus, Fig.3 suggests that students are increasingly willing to rush through problems. Conversely, they are less willing to spend a long time working on a problem as a session wears on. However, they are willing to spend 15-35 seconds working on a problem, irrespective of the number of problems they have already seen. This kind of information is valuable for the designers of ITSs, and for teachers who integrate technology-based instruction into the classroom.

A similar analysis can be done for action patterns. Figure 4 shows five series, one for each of the five action patterns observed in interactions with the Wayang Outpost ITS. Each point on a line represents the probability of observing the action pattern associated with the line, on the *nth* problem.

Note that Pr(A1), which is action pattern "skip problem", is quite low and remains fairly constant across the session. Pr(A2), which is "guess/abuse help", starts low and increases steadily through the sequences, while Pr(A4), "solve with help", and Pr(A5), "solve without help", start high and decline. An enigmatic pattern is A3, which is attempting but failing to solve the problem, and not using multimedia help.

These results for Wayang Outpost (Fig. 4) echo those for AnimalWatch (Fig. 3). The interpretation of both results is that students are less willing to work on problems as the tutoring session goes on. Under this interpretation it would seem students are less and less engaged or motivated as the session goes on. Is there a way to model students' engagement, even though it is not directly observable? Little is known about how students' engagement evolves over time, or whether there are consistent trends that might be exploited in the design of more effective pedagogical models.

4.1 Unobservable states

A challenge for data mining is to infer unobservable factors that might explain patterns in students' data. Students do not always perform in the most optimal manner, but we do not yet have ways to estimate the intentional states that influence their actions. Engagement has been suggested as a possible mechanism, referring to transient processes such as fluctuations in the learner's attention, willingness to engage in effortful cognition, and emotions associated with successful and unsuccessful problem solving. These processes are not observable, but they might be inferred by models with hidden states, such as Hidden Markov Models (HMMs).

We fit HMMs to sequences of action patterns for students who worked with the Wayang Outpost high school math ITS, with three hidden states representing levels of engagement: low, average and high[8]. Although we have no direct evidence that the hidden state actually corresponds to processes such as attention and cognitive effort, the HMM approach allows us to evaluate qualitative differences between the patterns, e.g., that the behaviors classified as Solving and Learning are likely to be emitted when students are attending and trying hard, whereas Guessing is more likely to be observed when students are not engaged, that is, not trying, perhaps due to

fatigue, lack of confidence, or problems that are too difficult.

We tested the HMMs in several ways summarized in Table 1. First, we used the Viterbi parse of each student's HMM to predict his or her next action pattern and compared the accuracy of these predictions with a) an order one Markov chain fit to the student's action sequence and b) an order one Markov chain fit to the entire group of students (CA or MA). Prediction accuracies for students' individual HMMs were 10-15% higher than using the student's own Markov chain and compared very well with the Markov chain fit to the entire group.

It is very encouraging to see that the HMMs for individual students added enough predictive power to handily surpass the individual students' Markov chains and exceed or nearly match the accuracy attained by pooling all the students' data into one Markov chain.

Table 1 Prediction accuracies using the HMMs and Markov chain models for the two datasets

	Student's HMM	Student's Markov Chain	Pooled Markov Chain
CA	42.13%	33.43%	45.6%
MA	48.35%	32.48%	45.7%

Next, we clustered students' individual HMMs using a variant of the BCD algorithm[9]. We produced three clusters for the CA students and four for the MA students. Remarkably, there were strong correspondences between the clusters across groups. For instance, both groups had a cluster in which students persisted in medium to high engagement states, and both had a cluster in which students started with low engagement and gradually increased their engagement over time. The MA group had a fourth cluster, not observed in the CA group, of students whose engagement gradually got worse. The declining engagement of the MA students was consistent with the fact that the MA sample included many students with weak math skills.

We used the average transition matrices for clusters, and also the average Markov chains, to predict action patterns for individual students. The results, in Table 2, are mostly worse than chance (recall, the default accuracy rate is .277). In sum, while an individual's HMM does a good job of predicting an individual's action patterns, one cannot predict action patterns from the average HMMs or Markov chains for clusters of students.

Table 2 Prediction accuracies using the group HMMs and Markov chain models for the two datasets

	Group HMM	Group Markov Chain
CA	34.40%	18.73%
MA	26.00%	15.52%

4.2 Session-Scale patterns of engagement

We can also track probabilities of engagement levels in exactly the same way as probabilities of action patterns, above. First, for each student, we produce a Viterbi parse of engagement levels. This is the sequence of transitions through engagement states that makes the observed sequence of action patterns most likely. Then, having inferred what a student's engagement levels were likely to be over time, we get time series of proportions of engagement levels. Figure 5 shows engagement levels across

problems for students who worked with the high school math ITS. It shows that students tend to start out the sessions in high engagement states, meaning that their action choices and latencies suggest that they are attending and concentrating. The probability of high engagement starts to decline around the 10th math problem, and continues to decline throughout the rest of the session. This decline is paralleled by a corresponding increase in the low engagement state.

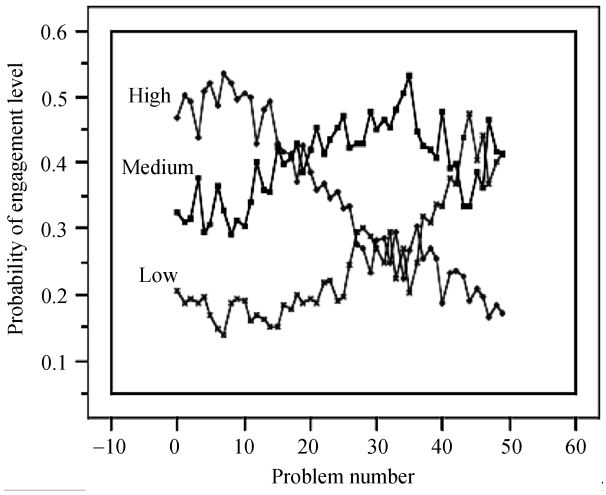

Figure 5. Estimated engagement with the high school math ITS over problems

4.3 Between-Session patterns

The previous analyses indicate that we can model a student's engagement by using the student's action sequence. However, from the perspective of an ITS designer, this conclusion is somewhat limited in that it requires that we have the action pattern sequence already in hand in order to diagnose the learner's likely engagement transitions. In other words, we do not know what kind of student we have until the session is over. However, as ITSs become more capable and contentful, students will work with them repeatedly, over several sessions, and so we might be able to use models from earlier sessions to customize interactions with students in later sessions. Roughly three dozen students had multiple sessions with the Wayang Outpost ITS, so we could test whether an HMM fit to a student in one session can make good predictions about that student in a later session.

4.4 Transferring models across sessions

For students who have multiple sessions with the ITS (10 from the CA group and 25 from the MA group), we trained an HMM for each student based on data from the first session and tested the model with data from the second session. The test was simply to predict the next action pattern in the sequence of action patterns in the second session. We compare our prediction results from the HMMs to those from Markov chain models (MCs) that do not take into account any hidden variable information. For the CA students, the average accuracy of a Session 1 HMM prediction to Session 2 action patterns was .459, whereas the average accuracy for MA

students was .365. Most encouragingly, the accuracy of a predictor on Session 1 is very strongly correlated with its accuracy on Session 2. For CA and MA students the correlations were .885 and .798, respectively. This means an ITS has good diagnostic information: If a student's Session 1 HMM does a good job of predicting her scores on Session 1, then it will probably do a good job on Session 2. If not, the HMM should not be used to predict performance on Session 2.

5 Learning Outcomes

We have explored opportunities for mining structure at several scales, from individual problem solving actions to long-term changes in unobservable factors such as engagement. Now we ask a different though related question: How can we tell whether an ITS works well? One answer is to look at outcome variables such as scores on a test after a tutoring session. However, outcome variables tell us little about what students actually do during ITS sessions that might explain the outcomes.

By "explain" we mean that an experimental group exhibits a particular behavior associated with a high learning outcome whereas a control group does not exhibit this behavior and has a lower learning outcome. Thus, explanation requires us to compare behaviors across groups in a statistically valid way[10]. Here is one such method for comparing two groups:

1. Derive one or more functions $\theta(\sigma_i, \sigma_j)$ to compare students' sequences of actions. Typically this function returns a real number.

2. Let $C_i = n_i(n_i - 1)/2$ be the number of pairwise comparisons between students within group G_i which contains n_i students. Define C_j in the same way.

3. Let $C_{i \cup j} = ((n_i + n_j)^2 - (n_i + n_j))/2$ be the number of pairwise comparisons between students across groups.

4. Let $\delta(i) = \sum_{a,b \in G_i} \theta(a, b)$ be the sum of all pairwise comparisons within G_i. Define $\delta(j)$ in the same way.

5. Let $\Delta(i, j) = \frac{(\delta(i)+\delta(j))/(C_i+C_j)}{\delta(i \cup j)/C_(i \cup j)}$.

6. If groups G_i and G_j are not different then one would expect $\Delta(i, j) = 1.0$. If $\Delta(i, j) \neq 1.0$, then we will wish to test whether it is significantly so. For this, we use a randomization procedure:

7. Randomization: Throw all the sequences in G_i and G_j into a single bucket G_{i+j}. Draw n_i sequences at random from G_{i+j} and call them G_i^* . Call the remaining n_j sequences G_j^* . Repeat steps 1-5 to get a single value of $\Delta(i, j)^*$. Repeat this process to get a few hundred values of $\Delta(i, j)^*$. The distribution of $\Delta(i, j)^*$ serves as a sampling distribution under the null hypothesis that groups G_i and G_j are not different. We compare $\Delta(i, j)$ to this distribution to get a p value, a probability of incorrectly rejecting the null hypothesis when it is true. (See Ref.[11] for details on randomization and bootstrap hypothesis testing.)

This procedure generalizes to multiple groups in the obvious way: If there are no differences between the groups then the average comparison among elements in

each group will equal the average comparison among elements of the union of all the groups.

5.1 Comparing sequences for text and heuristic students

We used the preceding method to compare progress for students who worked with either the Text or Heuristic version of the AnimalWatch ITS (Text provided only feedback about answer accuracy; Heuristic provided multimedia help). We looked at each student after $0, 10, 20, \ldots, 90$ problems and recorded how many problems on each of nine mathematics topics the student solved. Students' sequences were compared with the following function:

$$\theta(\sigma_i, \sigma_j) = \sum_{t=0,10,20,\ldots} \sqrt{\sum_{i=1,2,\ldots,9} (m_{i,a} - m_{i,b})^2}$$

That is, for two students, a and b, we look at the number of problems, m, of each class $i = 1, 2, \ldots, 9$ solved by each student, and square the differences. In other words, we treat a student's sequence as a trajectory through a nine-dimensional space, where each dimension is a problem class and each location in the space is given by a nine-vector of the number of problems of each class solved at that point. Then we compare sequences by the sum of Euclidean distances between corresponding points (i.e., points with equal values of t) in nine-space.

The test statistic was rejected only twice in 1000 randomization trials, so we can reject the null hypothesis that progress through the nine-topic problem space is the same for students in the Text and Heuristic conditions, with $p = .002$, a highly significant result.

It is one thing to test whether students in different experimental groups are different, another to visualize how they are different. This is not easily done when progress is in a nine-dimensional space. However, we can track the number of problem classes a student has mastered to some level at each point in a session. We chose a 50% criterion level, meaning that at a given point in a sequence, a student must have mastered 50% of the problems within a class to be given credit for mastery of that class at that point. We divided the students' sequences into subsequences of size 10 and calculated mastery, as defined, for each subsequence. Then we averaged the results over students, shown in Fig.6.

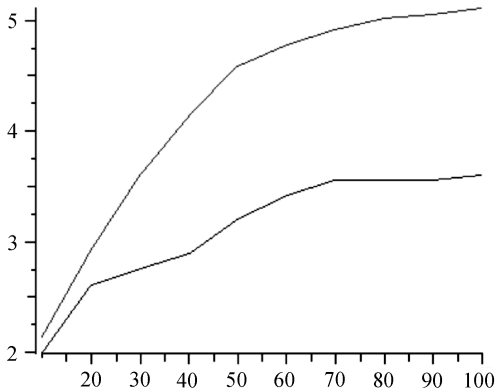

Figure 6. Problem classes mastered to criterion for two versions of the AnimalWatch ITS

The vertical axis is the mean number of problem classes mastered at the 50% level, the horizontal axis is actually ten points, one for each subsequence of ten problems, and so represents 100 problems. The higher of the two lines corresponds to the Heuristic condition, the lower to Text. One sees that on average, a student in the Heuristic condition masters roughly five topics to the criterion level of 50% in the first 100 problems, whereas students in the Text condition master only 3.5 topics to this level in the same number of attempts. These curves also can be compared with our randomization procedure, and are significantly different. Thus, the results help to establish that different pedagogical strategies used in the ITS do have an overall effect on students' progress, even though individual students solve different sequences of problems.

5.2 Linking action patterns to learning outcomes

Above, we defined examples of specific patterns of behavior that students exhibit when working with the Wayang Outpost ITS, such as guessing or learning by using the multimedia help, and showed that the pattern sequences could be used to predict how students would behave on subsequent problems. Here, we investigate whether the actions that students take with Wayang problems could be linked to actual changes in their math performance as indicated by improvement from pre- to post-tests of math skill. In contrast to the work presented earlier, we model the dynamics of students' performance with a dynamic Bayesian network.

Data from one sample of students was used to train a Dynamic Bayesian Network (DBN) model to recognize sequences of actions and interaction latencies that suggested the student was trying to learn by using the ITS help resources, or was trying to learn through independent effort[12]. Low estimates for both learning goals would suggest that the student was trying to avoid effort, for example, by guessing. The training data set included sequences of actions and interaction intervals captured from 115 students who completed 2151 Wayang Outpost problems. The DBN estimates of the student's learning goals (learn with hints, learn independently) were updated after each action was logged with a timestamp.

The resulting DBN model was evaluated with test data from an independent sample of students (N = 115). These students were given a pre-test which included math problems similar to those tutored in the ITS. Students then worked with the Wayang ITS, completing an average of 30 problems, and finally took a post-test. The pre and post-tests were presented to students on the computer, with answers scored automatically and downloaded for analysis.

The DBN model estimates for each student were averaged across the number of ITS problems completed by that student. Model estimates were used to generate a score for hint-based learning, ranging from 0 - 3, as well as a score for independent learning, ranging from 1 - 4. A student could receive a score of 0 in the model for hint-based learning if he or she never requested to view the multimedia hints in the ITS. In contrast, the scale for independent learning was anchored at 1 because if the problem was available on the computer screen, it was possible the student was attempting to solve it. The resulting scores were used to group students into four groups: Students who had low scores for both hint-based and independent learning; those with high hint-based and low independent learning scores; students with low hint-based and high independent learning scores; and students with high scores for

both hint-based and independent learning.

We examined the relation of students' pre-test scores and their scores for independent learning while working with the Wayang ITS, as estimated by the DBN model. The prediction was that students who could learn to solve the ITS problems without using the multimedia help features should be those who had relatively strong math skills, as indicated by higher pre-test scores. This prediction was supported; there was a significant relation between pre-test scores and independent-study scores. Thus, the DBN model captured differences in students' sequences of actions with the ITS that were significantly related to independent estimates of their initial math skills.

Next, we looked the relation of the DBN estimates to changes in pre- to post-test scores. Students who scored below the pre-test mean (those with weak math skills) fell into one of two groups: Students whose DBN estimates indicated that they were trying to avoid effort, and those with estimates suggesting that they were trying to learn by using the ITS help resources. Students with weak math skills who were estimated to be avoiding effort did not improve on the post-test. However, students who started with equally poor scores on the pre-test but who were estimated by the DBN to be trying to learn did show significant improvement on the post-test.

The model was also predictive for students who scored above the mean on the pre-test. The DBN model estimated that most of these students were trying to learn independently and, in fact, they showed significant improvement on the post-test. Thus, the results indicated the DBN model trained on data from one sample could successfully detect meaningful patterns in the actions of students in an independent sample, and that the model estimates predicted their actual learning outcomes.

6 Challenges for Educational Data Mining

We showed that ITS data has structure at several scales, from micro-sequences of hints, to short sequences of actions, and to long-term patterns during sessions and between sessions. We demonstrated the utility (in terms of prediction accuracy) of models with hidden state. We introduced a method to derive p values for statistical comparisons of groups of sequences. Some of our analyses were done algorithmically, others the old-fashioned way, with statistics packages. Thus, the first challenge to the data mining community is to develop algorithms to find automatically some of the regularities we found by hand.

Of these, the most important are what we call action patterns (Section 2), which are the abstractions on which all our subsequent analyses are based. To define action patterns, we used information about problem difficulty, the distribution of latencies, the best-performing students, and the latencies and correctness of responses on particular problems. A fine challenge for data mining is to exploit these and other kinds of information about problems to induce action patterns automatically.

Another challenge is to invent new distance measures for comparing sequences. The generalized Euclidean distance in the previous section is adequate but it does not acknowledge that every problem instance has structure, which includes latencies, the sequences of hints, levels of engagement, the problem topic, the number of problems a student has seen, and so on. It is one thing to compare the raw counts of each of nine kinds of problems with Euclidean distance, another to compare highly-structured records as just described.

A third challenge is to develop more sophisticated ways to induce intentional states such as engagement. Our HMMs induced something we called engagement, and we were happy to see engagement profiles for students cluster nicely and comparably across student populations. Yet we have no independent verification that the hidden states in our HMMs actually correspond to intentional states. This experimental work needs to be done and the HMM models (or other hidden-state models) need to be refined.

A fourth challenge is to use the discoveries revealed by data mining to improve the design of tutoring systems. For example, we have found structures in the sequences of problem solving actions that suggest variations in students' engagement over the course of a class session. Yet it is not entirely clear how the ITS should respond to shifts in the learner's interest and attention. In a pilot study, we added pedagogical messages to Wayang Outpost that were designed to help students re-focus their attention and to think about their learning goals. The messages were triggered to appear on the screen when guessing behavior was detected by the ITS. Students were asked to rate the value, helpfulness and appeal of the messages at the end of the activity. Although students rated the messages as believable and appropriate, they also indicated that they found the messages to be intrusive. Thus, although the ITS was able to detect guessing behavior, students had a strongly negative reaction to the intervention. It is possible that other responses by the ITS might be more successful in terms of helping students to sustain their attention on learning, but additional research will be required to design, deploy and evaluate such interventions.

The value of data mining for educational applications is enormous. National standards pertain to end results — performance — and provide no information about the process of learning. Teachers cannot easily individualize instruction because they do not have fine-grained analyses of how their students learn. Report cards are crude assessments and arrive too late to help. Whether one applies data mining techniques to the rich information available from ITSs or to other information gathered in classrooms, there is great potential for data mining to improve the quality of educational experiences. In addition, data mining approaches can help the designers of tutoring systems evaluate the performance of their systems when deployed with students in realistic classroom situations, and provide an empirical foundation for improving the software.

Acknowledgments

The research described in this paper was supported by grants from the U.S. National Science Foundation and the U.S. Institute of Education Sciences. The views expressed are not necessarily those of the funding agencies. We would like to recognize and thank our project colleagues Sinjini Mitra, Erin Shaw, Jean-Philippe Steinmetz, and Lei Qu at the Information Sciences Institute-USC, and Ivon Arroyo and Beverly Woolf at the University of Massachusetts-Amherst.

References

[1] http://www.ed.gov/nclb/landing.jhtml. Retrieved July 8, 2006.

[2] Beal CR, Qu L, Lee H. Classifying learner engagement through integration of multiple data sources. In: Proc. of the 21st National Conference on Artificial Intelligence. Menlo Park:

AAAI Press, 2006.

[3] Koedinger KR, Corbett AT, Ritter S, Shapiro LJ. Carnegie Learnings Cognitive Tutor: Summary of research results. White Paper. Pittsburgh PA: Carnegie Learning, 2000.

[4] Baker RS, Corbett AT, Koedinger KR, Roll I. Detecting when students game the system, across tutor subjects and classroom cohorts. In: Ardissono L, Brna P, Mitrovic A, eds., User Modeling: Lecture Notes in Artificial Intelligence, 3538, Berlin: Springer Verlag, 2005. 2002–224.

[5] Beck J. Engagement tracing: Using response times to model student disengagement. In: Looi C, McCalla G, Bredeweg B, Breuker J, eds., Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology. Amsterdam: IOS Press, 2005. 88–95.

[6] Stevens R, Johnson D, Soller A. Probabilities and prediction: Modeling the development of scientific problem solving skills. Cell Biology Education. Berlin: Springer-Verlag, 2005. 42–57.

[7] Sutton D, Cohen PR. Very predictive Ngrams for space-limited probabilistic models. In: Pfenning F, et al., eds., Advances in Intelligent Data Analysis V. Berlin: Springer Verlag, 2003. 134–142.

[8] Beal CR, Mitra S, Cohen PR. Modeling learning patterns of students with a tutoring system using Hidden Markov Models. In: Luckin R, Koedinger K R, Greer J, eds., Artificial Intelligence in Education: Building Technology Rich Learning Contexts That Work. Amsterdam: IOS Press, 2007. 238–245.

[9] Ramoni M, Sebastiani P, Cohen PR. Bayesian clustering by dynamics. Machine Learning, 2001, 47, 91–121.

[10] Beal CR, Cohen PR. Computational methods for evaluating student and group learning histories in intelligent tutoring systems. In: Looi C, McCalla G, Bredeweg B, Breuker J, eds., Artificial Intelligence in Education: Supporting Learning Through Intelligent and Socially-Informed Technology. Amsterdam: IOS Press, 2005. 80–87.

[11] Cohen PR. Empirical Methods for Artificial Intelligence. Cambridge MA: MIT Press, 1995.

[12] Qu L. Modeling the learner's attention and learning goals using a Bayesian Network approach [Ph.D. Thesis]. Dept. of Computer Science, University of Southern California, 2007.