Improving Intelligent Tutoring Systems: Using Expectation Maximization to Learn Student Skill Levels

Kimberly Ferguson, Ivon Arroyo, Sridhar Mahadevan,
Beverly Woolf, and Andy Barto

University of Massachusetts Amherst, Computer Science Department
140 Governor's Drive, Amherst, MA 01002
{kferguso, ivon, mahadeva, bev, barto}@cs.umass.edu

Abstract. This paper describes research to analyze students' initial skill level and to predict their hidden characteristics while working with an intelligent tutor. Based only on pre-test problems, a learned network was able to evaluate a students mastery of twelve geometry skills. This model will be used online by an Intelligent Tutoring System to dynamically determine a policy for individualizing selection of problems/hints, based on a students learning needs. Using Expectation Maximization, we learned the hidden parameters of several Bayesian networks that linked observed student actions with inferences about unobserved features. Bayesian Information Criterion was used to evaluate different skill models. The contribution of this work includes learning the parameters of the best network, whereas in previous work, the structure of a student model was fixed.

1 Introduction

Intelligent tutoring systems provide individualized instruction based on students' knowledge level. This is a hard problem mainly because knowledge is measured in terms of skill mastery, which are unobservable abstractions. Previous approaches include belief networks [12] and Bayesian networks [3],[1]. Traditional Bayesian approaches to determining students' understanding of a skill consist of the evaluation of observable student behavior on problems that are thought to require specific skills to be solved correctly. For example, the cognitive mastery approach performs Bayesian estimations of mastery given some observed evidence about students' correct or incorrect responses to problems [4]. Such models rely on parameters that link problems to skills, such as the probability of answering a problem correctly even though the skill is not mastered (guessing) and the probability of answering incorrectly given that the skill is mastered (slipping). These parameters are easier to estimate when a problem involves just one skill, but get more complicated as the number of skills involved in a problem increases.

We propose that a full model of student mastery of skills can be learned with machine learning techniques that deal with missing data. Our past work showed

M. Ikeda, K. Ashley, and T.-W. Chan (Eds.): ITS 2006, LNCS 4053, pp. 453–462, 2006.

that student models can be learned from simulated data of students' responses on problems [8]. In this paper, we present the results of learning a student model from real student data. We take students' actual responses from a pencil-and-paper pre-test and learn the parameters that link problems to skills, producing a model that allows us to make inferences about students' mastery levels. Last, we use the resulting model for inference and show how the mastery of skills improves after using our tutoring system.

One concrete future use of this model is a proper initialization of the student model before the tutoring session starts. Students will take an online pre-test, skill mastery will be inferred, and then the tutoring session will start, well informed about each student's strengths and weaknesses. More precisely, the ITS will have information about which skills a student has already mastered and what level of improvement is still needed for others.

2 Problem Definition

The goal of this research is to gain knowledge of the student's skill level and thereby improve the problem and hint selector of the Intelligent Tutoring System. Decisions made by the problem selector improve the tutor's ability to customize problems for an individual student.

Wayang Outpost is an Intelligent Tutoring System that emphasizes SAT-math preparation [2]. The system can individualize the tutoring based upon a student's specific needs. In particular, the intent is to develop a tutor that will select an action (i.e. a particular problem or hint) based on knowledge about a student's learning needs (i.e. problems he or she tend to get correct/incorrect/skip, inferred knowledge and skills, motivation level, mathematics fact retrieval time and learning style). The information we know about students increases as they use the system, but we also need to gather some student characteristics before the tutoring session begins, to initialize the student model and start proper tutoring. This information is collected by giving a pre-test to the students that includes short-answer problems similar to the problems within the tutoring session. This helps determine which skills a student initially has mastered so this information can be used to discern the best policy for optimizing the student's learning.

We cannot observe a student's mastery of a skill directly, so the tutor must infer this knowledge from student answers to problems involving these skills. Twelve basic geometry skills were identified by psychology and education researchers based on skills commonly used on the math portion of the SAT. These skills include: Skill 1. area of a square; Skill 2. area of a right triangle; Skill 3. properties of an isosceles triangle; Skill 4. identify rectangle; Skill 5. area of a rectangle; Skill 6. perimeter of a rectangle; Skill 7. identify right triangle; Skill 8. area of a triangle; Skill 9. Pythagorean theorem; Skill 10: corresponding angles; Skill 11: supplementary angles; Skill 12: sum of interior angles of a triangle.

Geometry problems were created utilizing these 12 skills, in the following fashion: 12 one-skill problems each of which used only 1 of each of the 12 skills;

12 two-skill problems which combine 2 of the 12 skills; 4 three-skill problems. Each set of three skills {1,2,3}, {4,5,6}, {7,8,9}, and {10,11,12} (see Figure 2) was combined to create the one-skill, two-skill and three-skill problems as described above for each set. This totals 28 problems for the student pre-test, 7 from each set of 3 skills. Some skills are obviously simpler than others (i.e. identify rectangle versus Pythagorean theorem); thus, we expect to find results that these skills are initially mastered more than others. Similarly, the one-skill problems were generally easier for students than the two-skill and three-skill problems.

We know exactly which problems involve which skills, and we know how each student answers each problem: incorrect, correct, or skipped. However, the actual mastery of a skill, which is what we really want to know, is a hidden variable. We treated these hidden skill variables as missing data and used Expectaction Maximization (EM) to estimate the parameters of the Bayesian network to learn skill mastery. This methodology is described in Section 4.

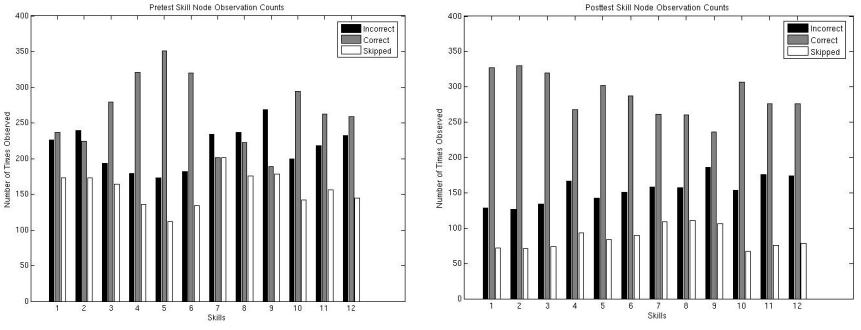
3 Data

The data used to create the models comes from a Spring 2005 study in an urban area High School in Massachusetts. Students took the pre-test, then spent two days using the tutor (50 minutes the first day and 30 minutes the next day) and then took the post-test. A total of 159 students took the pre-tests that were used to learn the Bayesian network and 132 students took the post-tests that were used to learn a second Bayesian network. The post-test involves different problems than the pre-test, but each problem is associated with the same set of skills as in the pre-test. Each problem resulted in one of four observable student actions: incorrect, correct, skipped or left blank[1]. Problems that are *skipped* supply information about the associated skill(s) being possibly unmastered, as opposed to problems that are *left blank* that we discount as uninformative.

Some information can be gathered based on the raw counts of how many student observed answers for problems involving each skill were incorrect, correct, or skipped (See Figure 1). Note when examining the raw counts that due to various reasons 27 more students took the pretest than the posttest, therefore small decreases/increases may not be significant.

The difference between the pre-test and post-test graphs suggests that the Intelligent Tutoring System is teaching students to improve their performance on these types of problems. More problems were answered correctly and less skipped from pre-test to post-test. More importantly, the non-uniform distribution of skill levels suggests that students improve more on certain skills than others, highlighting the value in modeling and learning the distribution of skill mastery levels across all students as well as within individual students.

[1] *Left blank* is different than *skipped* since a problem is *left blank* when the student runs out of time before reaching the problem (and thus is left blank at the end of the test); instead, a problem is considered *skipped* when the student may not know how to solve the problem and then skips to the following problem (and leaves the problem blank in between other answered problems).



(a) Pretest Observed Counts Across Skills

(b) Posttest Observed Counts Across Skills

Fig. 1. Improvement in student skill mastery from pre-test to post-test

4 Methodology

As stated earlier, the goal is to infer a student's skills based on evaluating only the pre-test. We now describe how to learn a Bayesian model of skill mastery that links problem outcomes to skill mastery levels. Such a network has hidden nodes representing the mastery of each skill (modeled by a binomial distribution) and observed nodes representing student answers (modeled by a multinomial distribution). Problems were created by hand to be associated with specific skills, so we have a clear idea of how problems are linked to skills. We constructed a Bayesian network (BN) linking each of the 12 skills to each of the problems that are associated with it. Each skill is used in four problems: 1 one-skill problem, 2 two-skill problems, and 1 three-skill problems (see Figure 2).

Even though the dependencies between skills and problems are known, it is not clear what the structure of the skills should be–whether the domain skills should be linked within the network (i.e., does the mastery of one skill affect the mastery of another skill?). Thus, we analyzed different models using the following methodology: 1) Expectation Maximization method (EM) was used to learn the

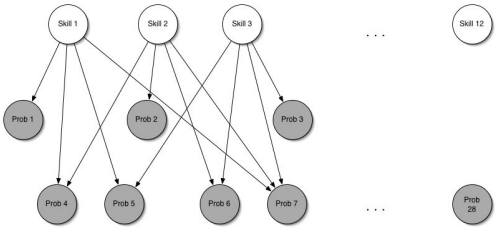


Fig. 2. Bayesian Network for Flat Skill Model: This identical linking pattern is repeated for each group of 3 skills and 7 problems. In total, there are 12 Skills (hidden nodes), and 28 Problems (observed nodes). Skills have 2 possible values: Not Mastered, Mastered. Problems have 3 possible observed actions: Incorrect, Correct, Skipped.

parameters of each Bayesian network on the pre-test data; 2) different BN models were evaluated using the Bayesian Information Criterion (BIC) to decide which model fit the data best; 3) the best fitting model was used for inference of student mastery levels given some outcome of responses to problems.

- *Parameter Learning using Expectation Maximization:* EM is a framework for maximum-likelihood parameter estimation with missing data [5]. EM finds the model parameters that maximize the *expected* value of the log-likelihood, where the data for the missing parameters are "filled in" by using their expected value given the observed data. In general, EM is trying to learn the pattern that best associates the observed data and the hidden parameters within the context of the specified graphical model. The log-likelihood value maximized though EM is used to calculate the BIC score of that model.
- *Model Evaluation using Bayesian Information Criterion:* The BIC [13] is used to evaluate different models. Simply put, given a set of data and probability distribution generating the likelihood, the larger the likelihood, the better the model fits. The BIC score is calculated with the following formula: $-2 * ll + npar * log(nobs)$, where ll is the log-liklihood, $npar$ represents the number of parameters and $nobs$ the number of observations in the model. Thus, the model with the highest BIC score is assumed to have the best structure for this task.
- *Inferring Mastery Levels:* Inference can be thought of as querying the model. The junction tree inference algorithm was used, which is an exact inference engine that uses dynamic programming to avoid redundant computation. Given a set of observations, the algorithm provides the probability of other events. For example, when a student answers Problem 1 (using only Skill 1) and Problem 2 (using only Skill 2) correctly, how likely is it that he or she will answer Problem 4 (using Skills 1 and 2) correctly as well. Inference is used to predict a student's overall skill mastery. Given a pre-test for a new student, inference is run on the learned model given the new student answers to estimate this student's skill levels.

Flat Skill Model

Based on the parameters learned when using uniform priors (where the probabilities of *mastered* and *not mastered* are both initially 50%), we discovered that it is not the *incorrect* answers from which we infer a lack of mastery, but the *skipped* answers instead. See technical report for more results [6].

Hierarchical Skill Difficulty Model

Figures 3 and 4 capture the idea that the skills are not mutually exclusive and may have different difficulty levels. For example, if the skill of identifying a triangle is not mastered, it seems probable that the skill of finding the area of a triangle is not mastered either. In particular, this Hierarchical Skill Difficulty Model is arranged so that each parent node is a skill which should be mastered before its child node can be mastered.

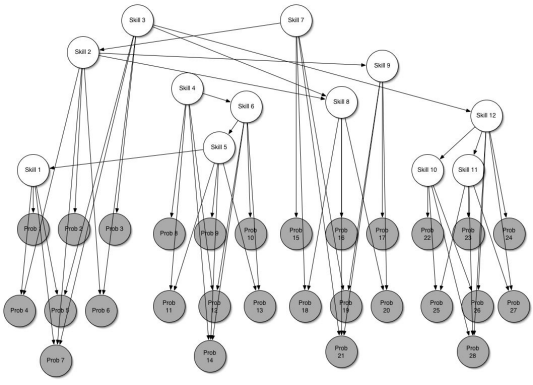


Fig. 3. Bayesian Network for Hierarchical Skill Difficulty Model: The hierarchy is based on the order in which skills should be learned (i.e. a student should know how to identify a right triangle (Skill 7) before learning how to take its area (Skill 8) which should all be mastered before learning the Pythagorean Theorem (Skill 9)). See Figure 4 for a higher level view.

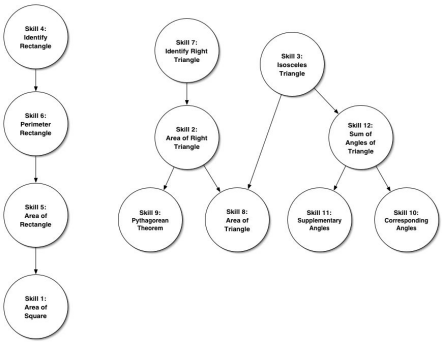


Fig. 4. Higher Level Look at Skill Difficulty Hierarchy Model (hidden skill nodes shown only)

5 Results and Discussion

Flat Skill Model With Simulated Priors. This section describes and compares the results achieved for the different models. As a first sanity check, we created an experiment using hand-coded priors to get baseline conditional probability Tables (CPTs). The structure of the model is identical to that of the flat skill model (Figure 2) only with user-specified parameters for all 40 nodes. We then sampled from this network to create simulated training data and built an identical model with the skill nodes hidden and initialized randomly. Finally, we found the maximum likelihood estimates of the parameters using the generated data on the model with random parameters. We compared the learned parameters to the

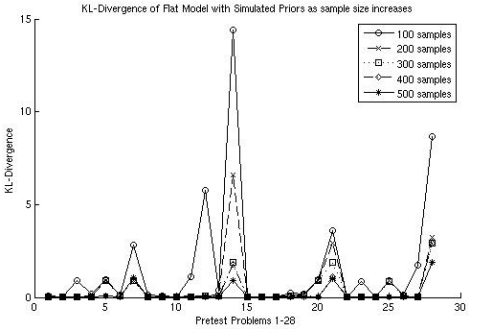


Fig. 5. Kullback-Leibler Divergence across 28 problems for increasing sample size

parameters set in the initial model using the Kullback-Leibler Divergence (see Figure 5). The KL-Divergence is a distance metric used to measure the difference between two probability distributions. We see that the learned parameters are fairly close to the "true" ones. The divergence decreases as the number of samples is increased, but begins to converge at an achievable sample size (400 students). The improvement to estimating the distribution with just 100 more samples is significant. We are in the process of collecting additional data which will increase our model's accuracy.

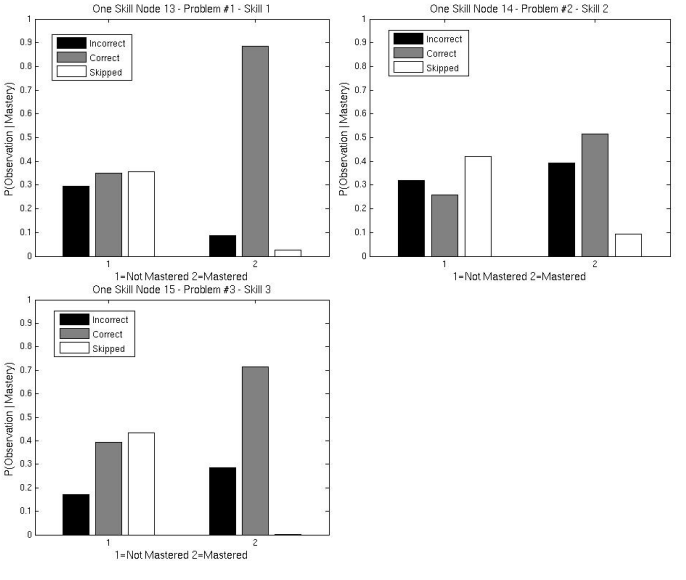


Fig. 6. Reflects the learned conditional probabilities of seeing each of the observations given it is know whether the skill is mastered or not: Flat Skill Model with uniform priors, One-skill problems involving Skills 1, 2 & 3

Flat Skill Model With Uniform Priors

Learned Model

In Figure 6, we consider one-skill problems. We have learned the conditional probabilities of a student getting a specific problem incorrect/correct/skipped assuming we know if the skill is mastered or not. Notice that it is not the *incorrect* answers from which we infer a lack of mastery, but *skipped* instead. This makes sense since the pre-test problems are not multiple choice, so if a student has no mastery of the skill needed, he or she will usually skip it. When the observation is *correct* we can usually infer the skill is mastered and when the observation is *skipped* we can usually infer the skill is unmastered, but *incorrect* can mean either. Perhaps, the student does not have the skill fully mastered and thus answers incorrectly, or he or she may make a simple math error or misunderstand the wording of the problem. For example, the graph on the top left of Figure 6 shows that a correct answer on problem 1 indicates a 90% certainty that Skill 1 (area of a square) is mastered.

Table 1. Bayesian Information Criterion For Flat and Hierarchical Skill Model (maximum BICs and average BICs are based on 50 random runs)

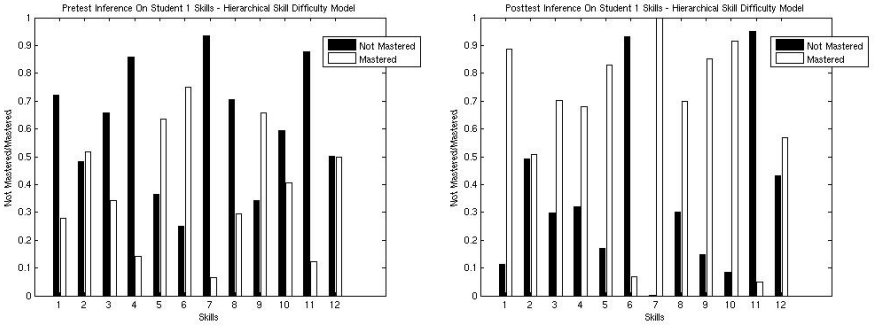
	Max BIC	Average BIC	Standard Deviation	Variance
Flat Skill Model	−5202.4	−5294.7	47.022	2216.7
Hierarchical Skill Difficulty Model	−4927.7	−5041.1	47.4178	2248.4

Hierarchical Skill Difficulty Model

In Table 1 we see that the Hierarchical Skill Difficulty Model yields the maximum BIC, as well as the highest average BIC. Thus, we conclude that this has the best structure to fit our data. This may be because some sets of skills/problems which were independent previously are now relevant to each other. For example, Skill 2 (area of a right triangle) is actually a subset of Skill 8 (area of a triangle), but in the flat BN these skills are not linked, and are no way dependent on each other. However, in this model, these skills are conditionally dependent on each other. Recall that the skills and their associated problems were originally split up into the following 4 independent sets of skills: {1,2,3}, {4,5,6}, {7,8,9}, {10,11,12}. In this model, these sets of skills are no longer independent so this model is actually more informative than the flat skill model.

Inference. We will look at a student's results to show the inference results of the best model: Hierarchical Skill Difficulty Model. We will look at the observations of the last set of 7 problems involving Skills 10, 11 and 12.

For Student 1 (Figure 7) we see an overall improvement in skill mastery from pre-test to post-test. This aligns with the student's performance on the tests. The test scores are calculated to evaluate individual improvement as follows: $corr/attmp$, the number of problems the students answered correctly divided by the number of problem the student attempted to answer (did not skip). Student 1 got a score of 27 on the pre-test and 83 on the post-test. For most skills,



(a) Student 1 Pretest Skill Mastery Inferred Probabilities

(b) Student 1 Posttest Skill Mastery Inferred Probabilities

Fig. 7. Student 1 Improvement in the probabilities of overall student skill mastery from pre-test to post-test

Student 1 shows a higher certainty of mastery in the post-test than in the pre-test. However, the certainty that Skill 6 (perimeter of a rectangle) is mastered has lowered from pre-test to post-test. This may be because the student did poorly on the post-test problem involving this skill. It is unclear as to whether this assumption should lead to the conclusion of mastery or non-mastery of Skill 6. Notice that Skill 7 (identify right triangle) initially had the most certainty of being unmastered in the pre-test, but shows a high certainty of being mastered in the post-test. This is an easy skill, and this result allows us to assume our tutor does a good job of teaching it. However, Skill 11 (supplementary angles) actually increases in it's probability of being not mastered from pre-test to post-test. This may show that the tutor is not doing a good job of teaching Skill 11, but may also be caused simply by the answers this student supplied on the tests and not the tutor's capability.

6 Conclusions

In summary, a data-centric approach was demonstrated to build a model of how student outcomes in a pre-test are linked to skill mastery. Graphical models were built to infer student mastery of skills, and the Bayesian Information Criterion used to evaluate several models and pick the most accurate one. A hierarchical model that links skills that are dependent on each other gave a better fit than a flat skill model that did not use this dependency information about the domain. In this best fitting model, related skills were linked so that parent nodes represented more basic skills that should be mastered before their more difficult child node skills are mastered (prerequisites).

We can now make predictions not only of how a student will do on a particular problem, but also of how much a student's performance on problems reflects their ability of individual specific skills. With the sparsity of data our current predictions are not optimally accurate, however, the Kullback-Leibler divergence

results prove that with a little more data, accuracy will greatly improve. Something interesting learned from this data-centric approach to estimation of skill mastery was the fact that the resulting models automatically found that if a skill is not mastered the student is more likely to skip the problem than answer incorrectly. This makes sense if we think that when students do not know the underlying skills, then they do not even attempt it; meanwhile, if the student has some idea, they probably attempt an answer and give an incorrect response. In general, data-centric models can help reveal valuable information such as this, which was not obvious at first sight. Finally, student improvement from pre- to post-test was demonstrated through raw counts, learned parameters, and inference, again highlighting the positive effect of the Intelligent Tutoring System.

References

1. Arroyo,I., Murray, T., Woolf, B., Beal, C.: Inferring Unobservable Learning Variables from Students Help Seeking Behavior. Intelligent Tutoring Systems (2004)
2. Beal, C., Arroyo, I., Royer, J., Woolf, B.: Wayang Outpost: An intelligent multimedia tutor for high stakes math achievement tests. American Educational Research Association annual meeting, Chicago IL (2003)
3. Conati, C., Gertner, A., VanLehn, K.: Using Bayesian Networks to Manage Uncertainty in Student Modeling. Journal of User Modeling and User-Adapted Interaction (2002), 12, 371-417
4. Corbett, A., Anderson, J., O'Brien, A.: Student modeling in the ACT Programming Tutor. Cognitively Diagnostic Assessment (1995) Hillsdale, NJ: Erlbaum, 19-41
5. Dempster, A., Laird, N., Rubin, D.: Maximization-likelihood from Incomplete Data via the EM Algorithm. Journal of Royal Statistical Society, Series B (1977)
6. Ferguson, K., Arroyo, I., Mahadevan, S., Woolf, B., Barto, A.,: Improving Intelligent Tutoring Systems: Using Expectation Maximization To Learn Student Skill Levels. University of Massachusetts Amherst, Technical Report (2006) TR-2006-09
7. Friedman, N.: Learning Belief networks in the Presence of Missing Values and Hidden Variables. Fourteenth International Conference on Machine Learning (1997)
8. Jonsson, A., John, J., Mehranian, H., Arroyo, I., Woolf, B., Barto, A., Fisher, D., Mahadevan, S.: Evaluating the Feasibility of Learning Student Models from Data. AAAI Workshop on Educational Data Mining (2005) Pittsburgh, PA
9. Jordan, M.: An Introduction to Graphical Models. unpublished book (2001)
10. Mayo, M., Mitrovic, A.: Optimising its behaviour with Bayesain networks and decision theory. International Journal of Artificial Intelligence in Education (2001)
11. Murphy, K.: The Bayes Net Toolbox for Matlab. Computing Science and Statistics (2001) vol. 33
12. Reye, J.: Student Modeling based on Belief Networks. International Journal of Artificial Intelligence in Education (2004) 14, 1-33
13. Schwarz, G.: Estimating the Dimension of a Model. Annals of Statistics (1978)