Pardos, Z. A., Heffernan, N. T. In Press (2009) Detecting the Learning Value of Items In a Randomized Problem Set. In *Proceedings of the 14th International Conference on Artificial Intelligence in Education*. Brighton, UK. IOS Press.

Detecting the Learning Value of Items In a Randomized Problem Set

Zachary A. Pardos[1], Neil T. Heffernan

Worcester Polytechnic Institute
{zpardos@wpi.edu, nth@wpi.edu}

Abstract. Researchers that make tutoring systems would like to know which pieces of educational content are most effective at promoting learning among their students. Randomized controlled experiments are often used to determine which content produces more learning in an ITS. While these experiments are powerful they are often very costly to setup and run. The majority of data collected in many ITS systems consist of answers to a finite set of questions of a given skill often presented in a random sequence. We propose a Bayesian method to detect which questions produce the most learning in this random sequence of data. We confine our analysis to random sequences with four questions. A student simulation study was run to investigate the validity of the method and boundaries on what learning probability differences could be reliably detected with various numbers of users. Finally, real tutor data from random sequence problem sets was analyzed. Results of the simulation data analysis showed that the method reported high reliability in its choice of the best learning question in 89 of the 160 simulation experiments with seven experiments where an incorrect conclusion was reported as reliable ($p < 0.05$). In the analysis of real student data, the method returned statistically reliable choices of best question in three out of seven problem sets.

Keywords. Bayesian networks, randomized controlled experiments, learning gain, data mining, machine learning, expectation maximization.

Introduction

Researchers that make tutoring systems would like to know which bits of educational content are most effective at promoting learning by students, however a standard method of figuring that out does not exist in ITS, other than by running costly randomized controlled experiments. We present a method that can determine which bits of content are most effective. We believe this method could help other researchers with a variety of different datasets particularly systems that present items in a randomized order. Cognitive Tutor [6], ANDES [5], IMMEX [10], Mastering Physics [5] and SQL-Tutor [7] are examples of systems that sometime give students a sequence of items in a randomized order and also have vast amounts of data.

In addition to systems typically presented to the AIED audience, traditional Computer Aided Instruction (CAI) systems often have this property of sometimes giving students items of a given skill in a randomized order. For instance, a modern web-based CAI system called studyIsland.com has data of this type from over 1,000

[1] National Science Foundation funded GK-12 Fellow

participating schools. The research questions is, can we come up with a method that would allow us to analyze these existing datasets to realize which questions, plus tutorial help in some cases, are most effective at promoting learning.

The intuition for the method exhibited in this paper is based on the idea that if you consistently see correct answers come after a certain question more than other questions, you may be observing a high learning gain question. While questions of the same skill may differ slightly in difficulty, questions with high difficulty deviation from the mean are likely tapping a different, harder skill as shown in learning factors analysis [3]. We propose to use static Bayesian networks and Expectation Maximization to learn which items cause the most learning. Guess and slip rates will account for question difficulty variation. We will accommodate for all permutations of orderings of the items by building networks for each ordering but will allow the conditional probability tables of each question to be shared across the networks.

1. Simulation

In order to determine the validity of this method we chose to run a simulation study exploring the boundaries of the method's accuracy and reliability. The goal of the simulation was to generate student responses under various conditions that may be seen in the real world but with the benefit of knowing the underlying best learning question.

1.1. Model design

The model used to generate student responses is an eight node static Bayesian network depicted in Figure 1. The top four nodes represent a single skill and the value of the node represents the probability the student knows the skill at each opportunity. The bottom four nodes represent the four questions in the simulation. Student performance on a question is a function of their skill value and the guess/slip of the question. Guess is the probability of answering correctly if the skill is not known. Slip is the probability of answering incorrectly if the skill is known. Learning rates are the probability that a skill will go from "not known" to "known" after encountering the question. The probability of the skill going from "known" to "not known" (forgetting) is fixed at zero. The design of this model is similar to a dynamic Bayesian network or Hidden Markov Model with the important distinction that the probability of learning is able to differ between opportunities. This ability allows us to model different learning rates per question and is key to both the generation of student data in the simulation and analysis using the purposed method.

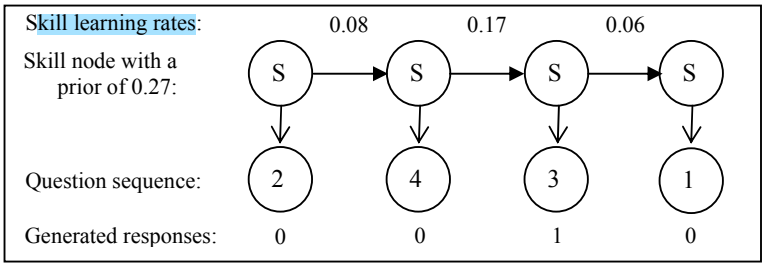

Figure 1. Simulation network model for a given student with a prior of 0.27 and question sequence [2 4 3 1]

While the probability of knowing the skill will monotonically increase after each opportunity, the generated responses will not necessarily do the same since those values are generated probabilistically based on skill knowledge and guess and slip.

1.2. Student parameters

Only two parameters were used to define a simulated student; a prior and question sequence. The prior represents the probability the student knew the skill relating to the questions before encountering the questions. The prior for a given student was randomly generated from a beta distribution that was fit to list of skill priors from a previous analysis of real tutor data [8]. The mean prior for that year across all skills was 0.31 and the standard deviation was 0.20. The beta distribution fit an α of 1.05 and β of 2.43. The question sequence for a given student was generated from a uniform distribution of sequence permutations.

1.3. Tutor Parameters

The 12 parameters of the tutor simulation network consist of four learning rate parameters, four guess parameters and four slip parameters. The number of users simulated was: 100, 200, 500, 1000, 2000, 4000, 10000, and 20000. The simulation was run 20 times for each of the 8 simulated user sizes totaling 160 generated data sets, referred to later as experiments. In order to faithfully simulate the conditions of a real tutor, values for the 12 parameters were randomly generated using the means and standard deviations across 106 skills from a previous analysis of real tutor data [8]. In order to produce probabilistic parameter values that fit within 0 and 1, equivalent beta distributions were used. Table 1 shows the distributions that the parameter values were randomly drawn from at the start of each run.

Table 1. The distributions used to generate parameter values in the simulation

Parameter type	Mean	Std	Beta dist α	Beta dist β
Learning rate	0.086	0.063	0.0652	0.6738
Guess	0.144	0.383	0.0170	0.5909
Slip	0.090	0.031	0.0170	0.6499

Running the simulation and generating new parameter values 20 times gives us a good sampling of the underlying distribution for each of the 8 user sizes. This method of generating parameters will end up accounting for more variance then the real world since guess and slip have a correlation in the real world but will be allowed to independently vary in the simulation which means sometimes getting a high slip and high guess, which is rarely observed in actual tutor data.

1.4. Methodology

The simulation consisted of three steps: instantiation of the Bayesian network, setting CPTs to values of the simulation parameters and student parameters and finally sampling of the Bayesian network to generate the students' responses.

To generate student responses the 8 node network was first instantiated in MATLAB using routines from the Bays Net Toolbox[2] package. Student priors and question sequences were randomly generated for each simulation run and the 12 parameters described in section 1.3 were assigned to the four questions. The placement of the question CPTs were placed with regard to the student's particular question sequence. The Bayesian network was then sampled a single time to generate the student's responses to each of the four questions; a zero indicating an incorrect answer and a one indicating a correct answer. These four responses in addition to the student's question sequence were written to a file. A total of 160 data files were created at the conclusion of the simulation program. Each of these data files were then analyzed by the learning detection method. The analysis method's accuracy and reliability results for the experiments are summarized in section 3.

2. Analysis

The purpose of the learning detection method is to calculate the learning rates of questions which are presented in a random sequence and determine which question has the highest learning rate and with what reliability. The simulation study gives us the benefit of knowing what the ground truth highest learning rate question is so we may test the validity of the method's results.

2.1. Model design

The analysis model was based on the same structure as the simulation model, however, the eight node simulation model only needed to represent a single question sequence at a time. The challenge of the analysis model was to accommodate all question sequences in order to learn the parameters of the model over all of the students' data. In order to accomplish this, 24 eight node networks were created representing all permutations of four question sequences. While the 24 networks were not connected in the Bayesian network's directed acyclic graph, they are still a part of one big Bayesian network whose parameters are tied together with equivalence classes, discussed in the next sub section.

2.2. Equivalence classes

Equivalence classes allow the 120 CPTs of the 24 networks to be reduced to eight shared CPTs and a single prior. Even though there are 96 (24*4) question nodes in the full network, they still only represent 4 unique questions and therefore there are still only four learning rates to be determined. Equivalence classes tie all of the learning rate CPTs for a given question into a single CPT. They also tie the 96 question guess and slip CPTs in to four CPTs, one per question. In the Bayesian network, the learning rate CPTs for a question is represented in the CPT of the skill node following question. Therefore the learning rate equivalence class for question 2, for instance, is always set in the CPT of the skill node that comes after the skill node for question 2. Question 2's learning rate equivalence class would appear in 18 of the 24 networks since in 6 of

[2] Kevin Murphy's Bayes Net Toolbox is available at: http://bnt.sourceforge.net/

those networks question 2 is the last question in the sequence. The first skill node in a sequence always represents the prior.

2.3. Methodology

The analysis method consisted of three steps: splitting the data file into 20 equal parts, loading the data in to the appropriate evidence array location based on sequence ID and then running Expectation Maximization to fit the parameters of the network for each of the 20 parts individually.

The motivation behind splitting the data was to test the reliability of the results among independent groups of student. By counting the number of times the most frequent high learning rate question appears we can compare that to the null hypothesis that each of the four questions is equally likely to have the highest learning rate. This was calculated with a two-tailed binomial probability for hypothesis testing. The binomial is of the "k out of N" type where k is the number of times the most frequent high learning rate question occurred (the mode) and N is the number of samples (20). P is the probability that the outcome could occur by chance. Since the outcome is a selection of one out of four questions, the P value here is 0.25. This binomial p value calculation tells us the probability that the outcome came from the null hypothesis that all questions have an equal chance of being chosen as best. A count of 10 or more would result in a p of < 0.05.

Since the 192 (24*8) node analysis network represented every permutation of question sequences, care had to be taken in presenting the student response evidence to the network. We used the sequence ID from each line of the data file to place the four responses of each student in the appropriate position of the evidence array. Expectation Maximization was then run on the evidence array in order to learn the equivalence class CPTs of the network. Starting points for the EM parameter estimation were set to mean values from previous research [8] (learning rates: 0.08, guess: 0.14, slip: 0.06) with the exception of the prior which was initialized at 0.50.

One of the limitations of our method is that it does not scale gracefully; the number of network nodes that need to be constructed is exponential in the number of items. This is one reason why we did not consider problem sets greater than four. We encourage researchers to investigate ways of scaling this method to large problem sets.

3. Results

The purpose of the simulation was to provide a means for verifying the validity of the Bayesian learning detection method. While real data was the ultimate goal, the simulation study was necessary to seed ground truth in question learning rates and verify that the method could detect the correct highest learning rate question and that the p value was a good indicator of the believability of the result.

We found that the method reported a reliable ($p < 0.05$) highest learning rate question in 89 out of the 160 experiments and in 82 of those 89 the reported highest learning rate question was the correct one as set by the simulation (7.8% error). In order to analyze what size learning rate differences the method could detect, the learning rate difference of the simulation's set highest and second highest learning rates was calculated for each experiment. The minimum learning difference was 0.001 and the max was 0.234. This list of differences was then discretized into four bins

corresponding to a learning difference range. The learning ranges were set to achieve equal frequency such that each bin contained 40 experiment results. Bins corresponded to the following learning difference rages: (0.001-0.0165], (0.0165-0.038], (0.038-0.0715] and (0.0715-0.234). For each range, the percentage of results, with $p < 0.05$ and a correct question choice, was calculated for each number of simulated users and plotted. The results are exhibited in the plot shown in Figure 2.

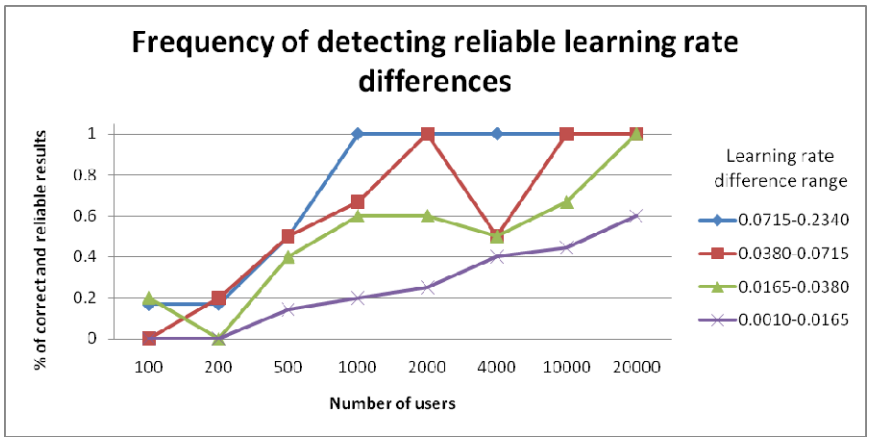

Figure 2. Plot of the frequency of detecting a correct and reliable learning difference of various size ranges

The plot shows a general increase in the likelihood of a reliable result as the number of users increase. The plot also shows that it was harder to detect smaller learning rate differences than large learning rate differences.

While 20,000 users were required to have a greater than 50% chance of reliably detecting the smallest difference of 0.0010-0.0165, only 500 were needed to detect any of the larger and more common differences with the same chance of a reliable result.

To test how well the method could identify no difference in learning we ran 14 experiments where the learning rates of all questions were set to zero and 14 experiments where the learning rates of all questions were set to 0.08. In these cases where the learning rates were all set the same, the method correctly concluded that there was no reliable best question in 26 of the 28 experiments (7% error).

4. Analysis of real tutor data

We applied this technique on real student data from our math tutoring system called ASSISTment. High school students ages 16-17 answered problem sets of four math questions at their school's computer lab two to three times per month. Each problem set was completed in a single day and the sequence of the problems were randomized for each student. Each problem contained hints and scaffolds that students would encounter if they answered the problem incorrectly. The method does not distinguish between the learning value of the scaffold content and the learning value of working through the main problem itself. Only responses to the main problem were considered and answers were only marked as correct if given on the first attempt.

4.1. Dataset

Student responses from seven problem sets of four questions each were analyzed. While there are problem sets of different sizes on the system, four is the average size of these problem sets. The problems in a given problem set were chosen by a subject matter expert to correspond to a similar skill. The data was collected during the 2006-2007 school year and the number of users per problem set ranged from 160 to 800. This data from the tutor log file was organized in to the same format as the simulation study data files. A sequence ID was also given to each student's response data indicating what order they saw the questions in.

4.2. Results

The analysis calculated a separate learning rate and guess and slip parameter for each of the four questions in the seven problem sets. The mean of the learning rates was 0.081 (similar to the mean used in the simulation) with a standard deviation of 0.035. The mean guess value was 0.18 which was within 1 std of the simulation guess mean, however the mean slip value was unusually high at 0.40. The average number of EM iterations was 95 with many of the runs stopping at the pre-set 100 iteration max.

Table 1. Learning rate results from analysis of student response from problem sets in the ASSISTment tutor

Problem set	Users	Best question	p value	prior	q1 rate	q2 rate	q3 rate	q4 rate
16	800	2	0.0652	0.6738	0.1100	0.1115	0.1017	0.1011
11	560	4	0.0170	0.5909	0.0958	0.0916	0.0930	0.1039
14	480	3	0.0170	0.6499	0.1365	0.0977	0.1169	0.1063
25	440	1	0.0652	0.7821	0.1392	0.0848	0.1157	0.1242
282	220	1	0.0039	0.7365	0.1574	0.0999	0.0991	0.1004
33	200	4	0.4394	0.7205	0.1124	0.1028	0.1237	0.1225
39	160	3	0.0652	0.6180	0.0853	0.1192	0.1015	0.0819

Statistically reliable results were reported in three of the seven problem sets as shown in Table 1. The numbers in the best question column and question learn rate column headers correspond to the IDs that were arbitrarily assigned to the questions.

Contribution

We have a presented a method that has been validated with a simulation study and shown to provide believable conclusions. While the power of the method could be improved with a different significance test procedure, the algorithm in its current form reports false conclusions less than 8% of the time, roughly in line with a 0.05 p value threshold. This method has broad applicability and can be used by many scientists who have collected responses in a randomized order. We believe researchers could easily adapt this method to identify poor learning content as well as identifying the learning of items that give no tutoring or feedback.

We know of no prior work that has shown how to learn about the effectiveness of a question, other than the typical method of conducting costly randomized controlled experiments. In some aspects, this method seems similar to treating a randomized

sequence of items as a set of randomized controlled experiments and could possibly be modified as an approach to a more general problem.

We claim this method could be important, for if we can learn what content is effective at promoting learning, we are one step closer to the elusive dream of building self-improving intelligent tutoring systems that can figure out the most effective material to present to students.

Future Work

A comparison between this Bayesian method of question analysis and an application of learning decomposition [2] should be made. Our colleague [4] is pursuing the same research questions as we are, using the learning decomposition method and the same dataset. There is evidence to suggest that a Bayesian method may be the most powerful [1] however we would like to confirm this by applying both methods to the same simulated datasets. Our analysis of the effect of item order on learning is in submission [9] and makes use of similar modeling techniques that were introduced here.

Acknowledgements

We would like to thank the Worcester Public Schools and the people associated with creating ASSISTment listed at www.ASSISTment.org including investigators Kenneth Koedinger and Brian Junker at Carnegie Mellon. We would also like to acknowledge funding from the U.S. Department of Education's GAANN and IES grants, the Office of Naval Research, the Spencer Foundation and the National Science Foundation.

References

[1] Beck, J. E., Chang, K., Mostow, J., & Corbett, A. T. (2008) Does Help Help? Introducing the Bayesian Evaluation and Assessment Methodology. In *Proceedings of the 9th International Conference on Intelligent Tutoring Systems*, 383-394.

[2] Beck, J. E., & Mostow, J. (2008) How Who Should Practice: Using Learning Decomposition to Evaluate the Efficacy of Different Types of Practice for Different Types of Students. In *Proceedings of the 9th International Conference on Intelligent Tutoring Systems*, 353-362.

[3] Cen, H., Koedinger, K., Junker, B. (2006) Learning Factors Analysis - A General Method for Cognitive Model Evaluation and Improvement. In *Proceedings of Intelligent Tutoring Systems 2006*, pp. 164-175.

[4] Feng, M., Heffernan, N.,Beck, M. In Press (2009) Using learning decomposition to analyze instructional effectiveness in the ASSISTment system. In *Proc. of Artificial Intelligence in Education 2009*.

[5] Gertner, A. G., & VanLehn, K. (2000) Andes: A Coached Problem Solving Environment for Physics. *Proc. of 5th International Conference on Intelligent Tutoring Systems*, Springer-Verlag, pp. 133-142.

[6] Koedinger, K. R., Anderson, J. R., Hadley, W. H., & Mark, M. (1997). Intelligent tutoring goes to school in the big city. *International Journal of Artificial Intelligence in Education*, 8, 30-43.

[7] Mitrovic, A. (2003) An Intelligent SQL Tutor on the Web. *International Journal of Artificial Intelligence in Education*, 13, 171-195.

[8] Pardos, Z. A., Heffernan, N. T., Ruiz, C. & Beck, J. In press (2008). Effective Skill Assessment Using Expectation Maximization in a Multi Network Temporal Bayesian Network. *Proc. of The Young Researchers Track at the 9th International Conference on Intelligent Tutoring Systems*, pp. 31-40.

[9] Pardos, Z. A., Heffernan, N. T. (in submission). Determining the Significance of Item Order in Randomized Problem Sets. *Proc. of the 2nd International Conference on Educational Data Mining.*

[10] Stevens, R. H., & Thadani, V. (2006) A Bayesian Network Approach for Modeling the Influence of Contextual Variables on Scientific Problem Solving. *ITS 2006*, LNCS 4053, Springer-Verlag. pp.71-84.