Modelling Student Knowledge as a Latent Variable in Intelligent Tutoring Systems: A Comparison of Multiple Approaches

Qandeel Tariq, Alex Kolchinski, Richard Davis

December 16, 2016

1 Introduction

This paper describes an effort to model students' changing knowledge state during skill acquisition using Bayesian Knowledge Tracing (BKT). For this purpose, the BKT Hidden Markov Model (HMM) which predicts the probability of correct application of a skill as a function of the number of previous opportunities to apply that skill. We do this using synthetic data generated according to BKT and Item Response Theory (IRT assumptions) modeling a student's knowledge path and a contextual estimation of these probabilities, starting with a hard-coded value of prior probability projecting whether the student knows a certain concept. We also test the models on two actual datasets discussed later. We trained 5 models to estimate student knowledge from observed responses apart from the baseline approach: Bernoulli, Logistic Regression, Average Response, Bayesian Knowledge Tracing (BKT) and Clustered BKT. Our results showed that the last two models worked consistently well on all 4 datasets.

2 Related Work

Since its introduction, the Bayesian Knowledge Tracing Model has been used extensively in studies of student learning and for various intelligent tutoring systems. It allows us to make the model more complex to cater for different models of learning. Good results for knowledge tracing would allow us to suggest resources to students based on their individual needs and skip the contents that would be too easy or too hard for them; essentially, this implies an efficient implementation of adaptive learning. So far, the researchers have used BKT in two ways: by The Hidden Markov Model and the Knowledge Tracing Algorithm [5]. The HMM form of BKT predicts the probability that a student will correctly apply a skill when they have the opportunity to apply it [1]. The simplicity of the BKT model allows us to solve the HMM analytically.

Other works in the Knowledge Tracing domain include the Deep Knowledge Tracing (DKT) as proposed by Piech et al. which shows significantly better results than the previous models [3]. They apply flexible recurrent neural networks that are "deep" in time to the task of knowledge tracing. This family of models represents latent knowledge state, along with its temporal dynamics, using large vectors of artificial neurons and allows the latent variable representation of student knowledge to be learned from the data rather than feeding in hard-coded values.

3 Datasets

We tested our models on 4 different datasets out of which two were synthetic. Following is the description for each dataset.

- Synthetic data generated according to IRT assumptions: We simulated virtual students learning certain virtual concepts and tested the accuracy of or predictions for their responses in this controlled setting. We generated 500 synthetic students and 10 concepts. Each question was linked to a particular concept and difficulty level. For a given concept's difficulty level and student's skill level, we use the IRT (Item Response Theory) model's "squeezed sigmoid" function to model the probability the student gets a question of that concept right: $P(correct|skill, difficulty) = .25 + .75 * \frac{1}{1+e^{difficulty-skill}}$. Note that given a fixed concept difficulty level, this probability can take on only one of two values, corresponding to knowledge = 1 and knowledge = 0. The students have closely correlated prior per-concept knowledge and learning rate parameter.

- Synthetic data generated according to BKT assumptions: Using the BKT assumptions which claim that students do not forget a concept once they learn it, we modelled students' knowledge about different concepts using a Markov chain between "don't know" and "know" states. The emission states for the HMM consisted of "correct" or "incorrect" i.e. 1 or 0. The sequences of emissions were then generated from the Markov chain. This data was generated for a total of 500 virtual students and 10 virtual concepts.

- KDD Bridge to Algebra 2006–2007 This was one of the real datasets [4] that was used to test our algorithms and was taken from a public source: KDD cup, an annual Data Mining and Knowledge Discovery competition organized by ACM Special Interest Group on Knowledge Discovery and Data Mining. The full dataset consisted of 808 concepts and 5968 students. To reduce the execution time for testing our models, we used a subset of 10 concepts and 500 students. Within the subset, to ensure that we have enough data for each concept, we put two thresholds, t1 and t2. The first threshold checked the number of students that had answered questions from a particular concept; if the number of students was less than t1, that concept was dropped. Furthermore, we set a second threshold to check the number of questions that each student had answered on a particular concept. If the number of questions per student per concept was less than t2, that student was dropped. Our algorithms were tested on the remaining concepts and students, ensuring that we had enough data for each.

- Assistments: Similar to the KDD Cup data, we used a subset of data [2] in this case as well. The full dataset consisted of 125 concepts and 4218 students. We used a subset of 10 concepts and 1678 students. The same restrictions for number of students per concept and number of questions answered by students were applied in this case as well.

The data that we are working with is structured as follows. A group of n students answers set of p problems from different c concepts. Each student's answers on problems for each concept is stored as a vector a, where $a \in \mathbb{R}^p$ and $a_i \in \{0, 1\}$, with 0 corresponding to an incorrect answer and 1 corresponding to a correct answer. For example, if a student answers a series of 6 questions in the concept, "long division", and gets the first 3 incorrect and the last 3 correct, that student is represented by the vector $[0, 0, 0, 1, 1, 1]$. Each concept is a matrix $M \in \ltimes^{n \times p}$ containing n students each of whom who answered p questions from that concept.

4 Methods

We are faced with the following problem: given student responses to a series of questions on a number of concepts, which concepts does the student know? Framed in this way, student knowledge on a particular concept can be modeled as a latent variable, while answers to questions on that particular concept can be modeled as observed variables.

We trained six models to try and estimate student knowledge from observed responses. Each of the models varies in its assumptions. The major variations are as follows.

- The Baseline, Bernoulli, Logistic Regression, and Bayesian Knowledge Tracing (BKT) models all assume that students are identical (i.e., they do not model per-student parameters). The Average Response and Clustered BKT models model student differences.

- The Baseline and Average Response models ignore concepts (i.e., they do not model concepts). The other four models train different models for different concepts.

Each model allows us to estimate a student's knowledge state on a particular concept (1) before the student has seen any questions on that concept and (2) immediately after a student answers a question on that concept. We use this to estimate the student's probability of answering that question correctly ($P(\text{correct})$) and use this probability to make predications about the student's answer. Finally, we compare the predicted answer sequence to the actual answer sequence to evaluate each model's effectiveness.

4.1 Baseline

We implemented a simple baseline model where $P(\text{correct}) = 1$. That is, this model predicts that all students answer every question correctly. This model provided a lower-bound against which we could compare all other models.

4.2 Bernoulli

We trained a separate Bernoulli model for each concept in the data. In this model, $P(\text{correct}_k) = \phi_k$ where ϕ_k is equal to the percent of correct answers on that concept in the training data. If there are c concepts in the training data, we learn c distinct ϕ's. In this model, students are assumed to be identical; so when making a prediction about a question on concept k, we use ϕ_k no matter the student.

4.3 Logistic Regression

For each concept, model P(correct) with a logit link on (# of questions that the student saw for that concept) as the only independent variable. Students are asssumed to be identical.

4.4 Average Response

We model $P(\text{correct}_k)$ for concept k per student as the student's percent of correct answers on that concept k up to that point. For example, if a student is presented with a question on the concept 'Quadratic Equation,' we calculate

$$P(\text{correct}_{\text{Quadratic Equation}}) = \frac{\text{Number of prior correct answers on 'Quadratic Equation'}}{\text{Number of total prior questions on 'Quadratic Equation'}}.$$

4.5 Bayesian Knowledge Tracing

In Bayesian Knowledge Tracing (BKT) we fit one model per concept, which means we treat all students as identical. Furthermore, we assume a two-state learning model. That is, for a given concept, we assume that the student either knows the concept or does not know the concept. A student can make the transition from not knowing to knowing each time she has the opportunity to answer a question on that concept. In BKT, we assume that once a student learns a concept, she never forgets that concept. We also model the probability that the student answers incorrectly despite knowing the concept (called a slip) as well as the probability that the student answers correctly despite not knowing the concept (called a guess). All of the parameters for this model are summarized as follows:

- $P(L_k^c)$: The probability that the student knows concept c when answering the kth question on concept c.
- $P(L_0^c)$: The probability that the student knows concept c before answering any questions on that concept (a special case of the previous probability).
- $P(T^c)$: The probability of learning concept c when answering a question on that concept.
- $P(G^c)$: The probability of guessing correctly on concept c if in the not-knowing state.
- $P(S^c)$: The probability of answering incorrectly (slipping) on concept c despite being in the knowing state.

There are two distinct stages to using the BKT model. First, we learn the parameters for each model from the training data. Second, we use these models to infer a student's knowledge state as she works through questions.

4.5.1 BKT as a Hidden Markov Model

What we have described in the previous sections is a Hidden Markov Model with the following parameters:

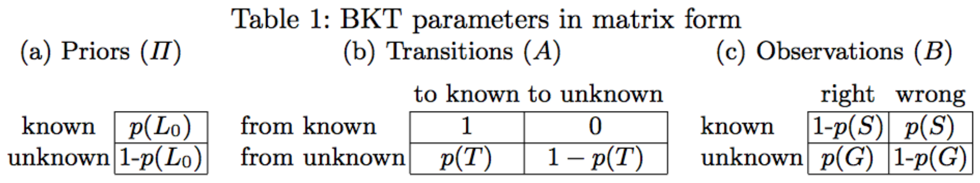


Table 1: BKT parameters in matrix form

(a) Priors (Π)

known	$p(L_0)$
unknown	1-$p(L_0)$

(b) Transitions (A)

	to known	to unknown
from known	1	0
from unknown	$p(T)$	$1-p(T)$

(c) Observations (B)

	right	wrong
known	1-$p(S)$	$p(S)$
unknown	$p(G)$	1-$p(G)$

Modeling the problem as a HMM allows us to use the EM algorithm to learn prior probabilities, transition probabilities, and emission probabilities from the training data.

4.5.2 Updating Student Knowledge

Given an observation of the student's response at time opportunity k (correct or incorrect) on concept c, the probability $P(L_k^c)$ that a student knows concept c is calculated using Bayes rule. When a correct response is observed, this probability is as follows:

$$P(L_n^c|\text{correct}_k^c) = \frac{P(L_k^c)\cdot(1-P(S^c))}{P(\text{correct}_k^c)}$$

When an incorrect response is observed, this probability is as follows:

$$P(L_k^c|\text{incorrect}_k^c) = \frac{P(L_k^c)\cdot P(S^c)}{P(\text{incorrect}_k^c)}$$

Finally, we show how the student's knowledge of concept c is updated given her interaction with the system. This estimate is the sum of two probabilities: the posterior probability that the student already knew the skill (contingent on the evidence), and the probability that the student did not know the skill, but was able to learn it.

$$P(L_k^c) = P(L_{k-1}^c|\text{evidence}_{k-1}^c) + (1-P(L_{k-1}^c|\text{evidence}_{k-1}^c))\cdot P(T^c)$$

4.5.3 Making Predictions with BKT

The probability of a student getting a question on concept c correct at time opportunity k is:

$$P(\text{correct}_k^c) = P(L_k^c) \cdot (1 - P(S^c)) + (1 - P(L_k^c)) \cdot P(G^c).$$

4.6 Clustered Bayesian Knowledge Tracing

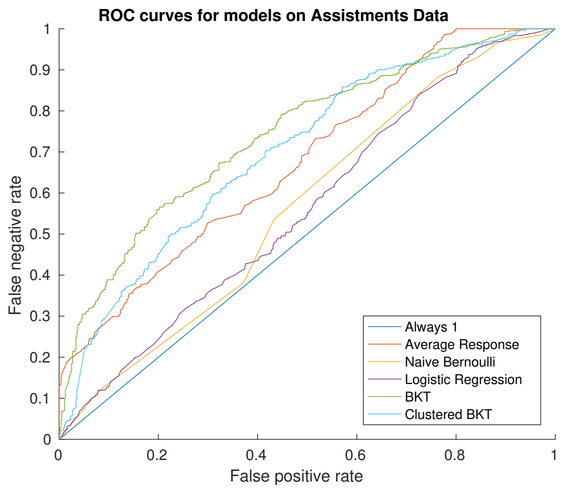
In Clustered BKT, we start by clustering the students into distinct groups. We then train a distinct set of HMMs for each group. For example, if we cluster students into g groups and have c concepts in our data, we will train $g \cdot c$ models (instead of the c models for BKT).

Initially, we represented students as n-dimensional vectors and used k-means clustering to divide them into groups. When this did not provide any additional gains over BKT, we implemented a pseudo-clustering algorithm as follows. We first divided students in the training data using a median split on the percent of total correct answers across all concepts. Next, we trained one HMM per group per concept, leaving us with double the number of models as BKT. This method assumes there are high-achieving and low-achieving students in the data, and that the BKT parameters that best model the high-achieving group are different from those that best model the low-achieving group.

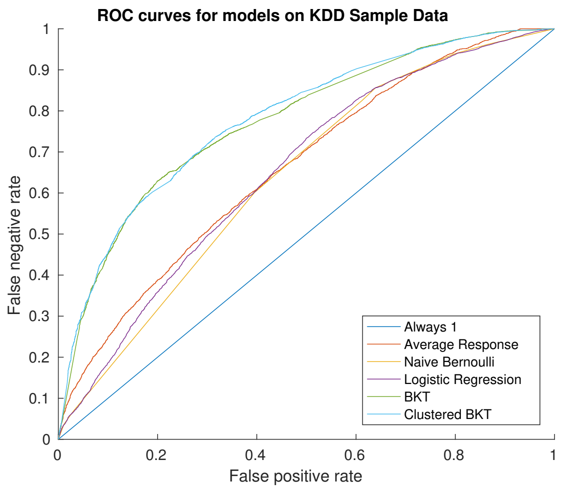
4.6.1 Making Predictions with Clustered BKT

We made predictions for novel answer sequences such that at each index k, the percent of correct answers up to k determined which HMM's predicted state sequence was used to predict the next emission. Once we decided on the model, we used the method described in the BKT section above to find $P(\text{correct}_k)$.

5 Results



ROC Curves for Assistment data set



ROC Curves for KDD data set

We evaluated each of the five models on each of the four data sets, for a total of twenty combinations. In each case, we proceeded via Monte Carlo cross-validation, where at each iteration of the testing process, the data was pointwise randomly split in a 90:10 ratio training:validation data. In each data set, each training point is a student sequence of correct or incorrect answers, so at each iteration there would be a random set of students (real or fake, depending on whether the data set was real or synthetic) whose answer sequences were used to train the algorithm and a random set of students whose sequences would be used to evaluate the algorithm. Each model was trained on the training set, and then evaluated on the validation set in the following manner. Each sequence would be fed into the model answer by answer, and at each answer index i the model would output its estimate of the probability that the $i + 1'th$ answer would be correct (i.e., a 1).

In this manner, the model's outputs would comprise a vector of probabilities $p_i; p_i \in [0, 1] \forall i$ which could be compared to the true sequence of outputs $x_i; x_i \in \{0, 1\} \forall i$. Our per-model metrics are calculated relative to these vectors, averaged over all student sequences in the validation set. MSE (mean-squared error) is simply $\sum_{i=1}^{n} (x_i - p_i)^2$. For the error rate, we set a threshold of .5 and reported the proportion of answer indices where the rounded probability did not match the true student output. For the AUROC curve, we used Matlab's library to generate false-positive and false-negative rates for different values of this rounding threshold, and calculated the area under the resulting curve.

For the results shown in the graphs (Figure 1), we report the results of 100-iteration Monte Carlo cross-validation, where the error bars show the sample standard deviation of the reported statistics. There was no need for optimizing

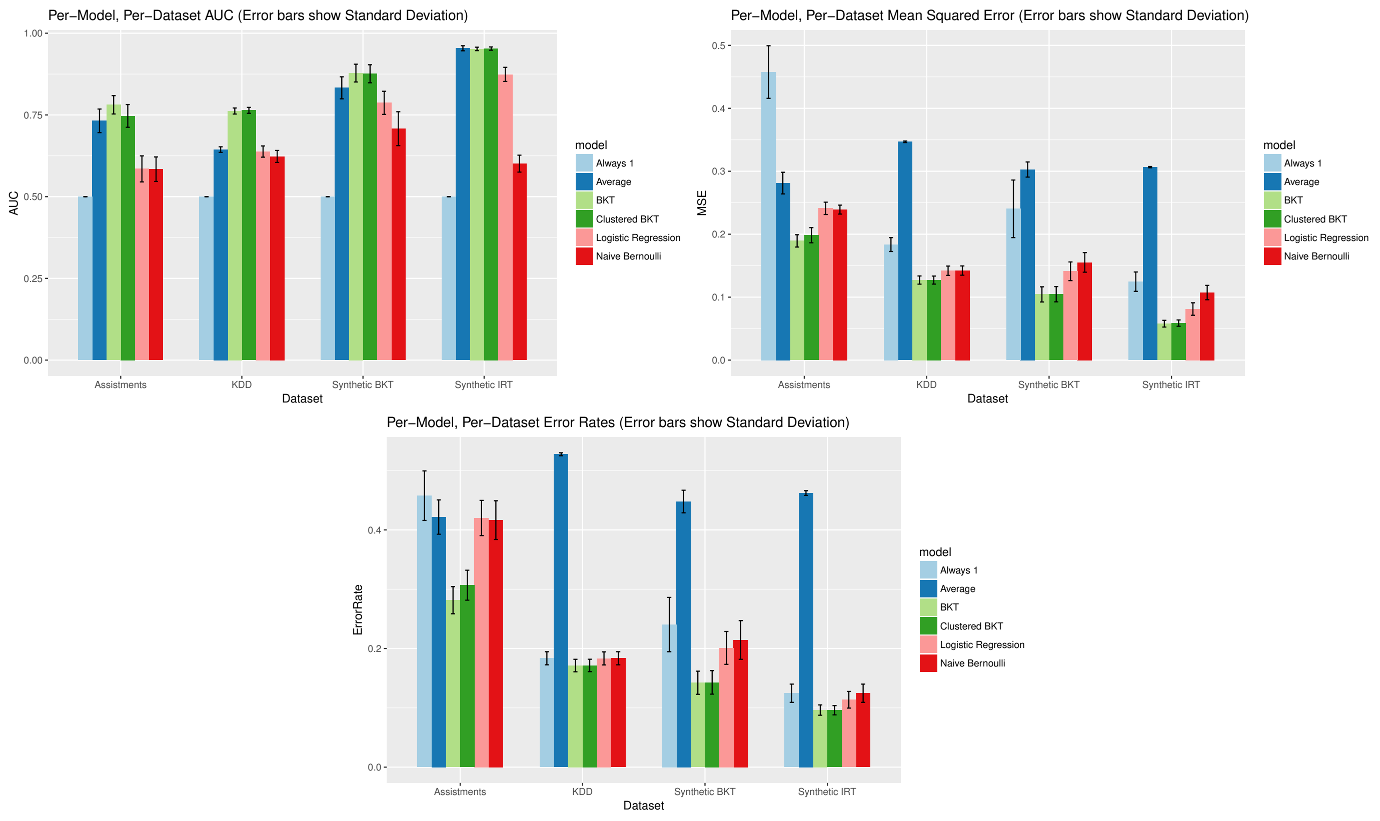


Figure 1: Per-model metrics: AUROC, MSE, overall error rate

fitting meta/hyper parameters, as each model that we used does not have parameters other than those that we fitted empirically. For the HMM-based BKT models, we used the EM algorithm and restarted it 10 times per iteration to ensure convergence to a global optimum.

We found that the simplest useful models, Naive Bernoulli and Logistic Regression, performed consistently well on the synthetic data sets (.15 and .1 MSE on synthetic BKT and synthetic IRT, respectively) and on the KDD data set (.15 MSE), but significantly worse on the Assistments data (.25 MSE). Notably, logistic regression outperformed Naive Bernoulli on the synthetic data but not on the real-world data, meaning that the time-dependency assumption of students being more likely to answer related questions correctly as time went on seems to have been false. The BKT models performed the best on each data set (.06, .1, .13 and .2 MSE on IRT, BKT, KDD and Assistments respectively), validating the choice to use them despite the relative complexity of the model. Notably, the clustered BKT model performed almost identically to the unclustered model, suggesting that clustering by student response patterns did not distill any useful information for the sub-models to differentiate by. The always-1 and average-response models were used for comparison and as might be predicted, their performance was generally much worse than the other models. The other quality metrics (AUROC and average error) confirmed the overall best performance of the BKT models, although the BKT models' lead for the KDD and synthetic IRT data was relatively slim ($<.05$ better than the next best models). For all data sets, the ROC curves of the BKT models were consistently higher than the other models.

We do not believe that our results reflect significant overfitting, as the standard deviations across Monte Carlo cross-validation trials are very small relative to the means. This means that regardless of the training and test set, our models learned similar information from the data, suggesting that they did not overfit to the structure of our training data itself. That said, the very large difference in performance of all models between the two real-world data sets (Assistments and KDD) suggests that variation in performance on further real-world data is likely to be high.

6 Conclusion

Only BKT and Clustered BKT reliably outperform baseline on all datasets. The average response model performed nearly as well as BKT and Clustered BKT on the Assistments and Synthetic data, but performed far worse than BKT on the KDD data. Although results show that BKT is the most reliable model to use when making predictions about student knowledge across multiple datasets, it did not outperform baseline by very much in some cases. Clustered BKT and BKT achieved nearly identical performance across all datasets. This was unexpected implying that in the future a different set of features should be chosen to perform clustering.

References

[1] Beck, J. E., and Chang, K.-m. Identifiability: A fundamental problem of student modeling. In *International Conference on User Modeling*, Springer, pp. 137–146.

[2] Heffernan, N.Heffernan, C. The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching.

[3] Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L. J., and Sohl-Dickstein, J. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*, pp. 505–513.

[4] Stamper, J., N.-M. A. R. S. G. G. . K. K. Bridge to algebra 2008-2009. challenge data set from kdd cup 2010 educational data mining challenge.

[5] van De Sande, B. Properties of the bayesian knowledge tracing model. 1–10.