Modeling Skill Acquisition Over Time with Sequence and Topic Modeling

José P. González-Brenes
Pearson School Research Network, Philadelphia, PA, USA

Abstract

Online education provides data from students solving problems at different levels of proficiency over time. Unfortunately, methods that use these data for inferring student knowledge rely on costly domain expertise.

We propose three novel data-driven methods that bridge sequence modeling with topic models to infer students' time varying knowledge. These methods differ in complexity, interpretability, accuracy and human supervision. For example, our most interpretable method has similar classification accuracy to the models created by domain experts, but requires much less effort. On the other hand, the most accurate method is completely data-driven and improves predictions by up to 15% in AUC, an evaluation metric for classifiers.

1 Introduction

In many instructional settings, students are graded by their performance on instruments such as exams or homework assignments. Usually, these instruments are made of *items* – questions, problems, parts of questions – which are graded individually. Recent interest in Massively Open Online Courses and intelligent tutoring systems promises large amounts of data from students solving items at different levels of proficiency over time. A challenge in educational technology is how to use these data to adapt the instruction to student needs. Modern personalization technologies in education use *student models* (VanLehn, 1988), an estimate of the skill proficiency of the students.

Student modeling techniques that allow *longitudinal data* – data collected at different time points – require a mapping of items to skills (Corbett and Anderson, 1995). Unfortunately, these mappings are mostly built manually by context experts and psychologists — an effort that can take years to accomplish (Beck, 2007). Moreover, these mappings are treated as irrefutable truth, while in fact, the context experts may be uncertain as how to map a specific item into a skill ontology. In this paper, we propose models that allow to reason on what and when students learn.

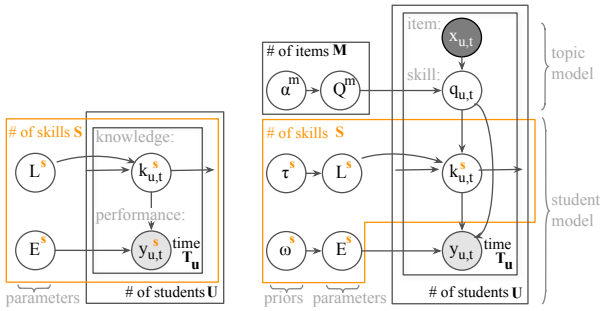
Appearing in Proceedings of the 18^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2015, San Diego, CA, USA. JMLR: W&CP volume 38.

2 Data-Driven Student Modeling

We operationalize a skill as a grouping – either through hard or soft clustering – of items that have similar response patterns by students. Other authors refer to skills as topic skills (Desmarais, 2011), knowledge components or factors (Cen et al., 2006). Usually, skills are identified by domain experts to understand the classroom progress. It is useful for skill definitions to be interpretable. For example, a teacher may want a list of students that have not mastered the skill of subtraction. On the other hand, a list of students that have not mastered a skill called #97 may be less desirable.

Existing student modeling techniques require a mapping from items to skills. For example, Knowledge Tracing (Corbett and Anderson, 1995), the *de facto* standard for student modeling from longitudinal data, uses a Hidden Markov Model (HMM) per skill to model the student's knowledge as latent variables. Figure 1a uses plate notation to describe the graphical model of Knowledge Tracing. The binary observation variable $y^s_{u,t}$ represents whether the student u gets the t^{th} practice opportunity of skill s correctly. The binary latent variable $k^s_{u,t}$ represents whether the student has learned the skill. L and E are the skill-specific parameters of the model. The *transition parameters* L are often referred as initial knowledge (or L0), learning, and forgetting probabilities. The *emission parameters* E are commonly referred as guess and slip.

Knowledge Tracing is not fully data-driven because it requires experts to define an item to skill mapping. We propose three data-driven methods:



(a) Knowledge Tracing (b) Topical HMM

Figure 1: Plate diagram of student models. The circles represent whether the variable is latent (white), observed in training (light), or fully observable (dark), and plates represent repetition.

- *Automatic Knowledge Tracing* is a novel pipeline that first discovers an item to skill mapping and then estimates the student model. The advantages of Automatic Knowledge Tracing are simplicity of implementation and efficiency (§ 2.1).
- *Topical Hidden Markov Models (Topical HMM)* is a joint model that combines topic modeling with HMMs. Topical HMM improves on the performance of Automatic Knowledge Tracing with a more computationally expensive model (§ 2.2).
- *ItemClue* discovers interpretable models by using low-cost domain knowledge (§ 2.3).

2.1 Automatic Knowledge Tracing

Algorithm 1 Automatic Knowledge Tracing

Require: performance sequences $\mathbf{y}$, # of skills S

1: **function** AUTOMATICKNOWLEDGETRACING
2: $\mathbf{Y} \leftarrow$ build_matrix($\mathbf{y}$)
3: Estimate $\mathbf{V}, \mathbf{K}$, such that $\mathbf{Y} \approx \mathbf{V} \times \mathbf{K}^\intercal$
4: $\langle g \rangle \leftarrow$ cluster($\vec{v_1}, \ldots, \vec{v_M}, S$), where $\vec{v_m} \in \mathbf{V}$
5: **for** $s \in 1 \ldots S$ **do**:
6: $\mathbf{y_s} \leftarrow$ filter_skill($\mathbf{y}, \langle g \rangle, s$)
7: $L_s, E_s \leftarrow$ train_hmm($\mathbf{y_s}$)
return $L, E, \langle g \rangle$

We formulate Automatic Knowledge Tracing using clustering and matrix factorization. Matrix factorization algorithms are useful to describe a large number of items with a small number of unobserved factors. Many matrix factorization techniques exist, some which are equivalent to classic assessment methods in education (Bergner et al., 2012). However, they typically do not generalize to unseen students, and do not handle longitudinal data. In § 3, we describe the specific matrix factorization implementation we use for our experiments.

Algorithm 1 describes the Automatic Knowledge Tracing algorithm. Its input is the set of ordered sequences $\mathbf{y}$, that describe the performance of students answering items, and the number of skills S. Automatic Knowledge Tracing is a pipeline with three major steps:

1. It first maps the sequences into the matrix (line 2) where each entry is the performance of a student solving an item. If a student answers an item multiple times, for simplicity, it only encodes the first attempt. It then uses a matrix factorization algorithm to discover the S latent loadings of items (line 3).
2. Then, it clusters the item loadings into S groups. We interpret the groups as the item to skill mapping (line 4).
3. Finally, it learns the Knowledge Tracing parameters using the item to skill mapping found (line 7). Automatic Knowledge Tracing algorithm can be applied to unseen students, because it learns skill-specific parameters, using the clusters found during training.

2.2 Topical Hidden Markov Model

Automatic Knowledge Tracing is an efficient method to model student knowledge from longitudinal data. However, it makes two strong assumptions: (i) the first encounter of the skill carries most of the information, discarding other temporal data, and (ii) each item requires exactly one skill. Albeit with a higher computational cost, Topical HMM is a solution for when these assumptions represent a limitation. Topical HMM does not converge with off-the-shelf training procedures. In preliminary work (González-Brenes and Mostow, 2012, 2013) we presented limited results of Topical HMM. Here, we only briefly summarize Topical HMM, but we report our novel training method and substantially new results with original experiments, analyses, datasets, and baselines.

2.2.1 Model

Figure 1b shows the plate diagram of Topical HMM. Topical HMM combines topic modeling with HMMs, by allowing topic (skill) knowledge to change over time. It differs from prior work on dynamic topic models (Gruber et al., 2007) by allowing output variables that we use for modeling performance over time, and by having topics that do not change temporally.

The priors α, τ, ω are modeled with Dirichlet distributions, because they allow easy calculations of the posteriors of multinomial parameters. We use multinomials to model the learning and emission parameters. These parameters exist for each skill (not per student), and apply to unseen students.

The parameters Q represent the topic model – the item to skill mapping. Topical HMM allows Q to be estimated by an expert, or from data. Each parameter Q^m is an S-dimensional multinomial representing the skills required for item m. We assume that we know a priori the total number of items M and the number of skills S. Items are a convex combination of skills; for example, if $Q^m = [0.5, 0.5, 0, 0]$, we interpret item m to be a mixture of skills 1 and 2, and not needing skills 3 and 4. For every practice opportunity, the random variable q represents the skill of the item. *Sensu stricto*, we allow only one skill q for each item, but by modeling uncertainty on which skill it is, we enable soft membership. The node $k^s_{u,t}$ is the knowledge student u has of skill s at the t^{th} learning opportunity, and $y_{u,t}$ is the binary output variable that models whether the student answer is correct. Unlike Knowledge Tracing, y is not indexed per skill, because the item to skill mapping may be discovered with data.

We want Topical HMM to enforce that the knowledge of a skill only changes when the skill is being practiced. This way, the knowledge is updated with evidence from data. We also want Topical HMM to be general enough to be able to model the popular requirement of disallowing forgetting (prohibit transitioning from a higher level knowledge state to lower knowledge). These constraints introduce determinism, which remove the theoretical convergence guarantees of conventional inference techniques. For example, consider a Topical HMM with two knowledge states—novice and master. If we disallow forgetting, once a knowledge node is sampled as a *master* state at time t, a Pointwise Gibbs sampler would never sample a *novice* value at time $t+1$. This implies non-ergodicity, and that the sampler may not visit some regions.

2.2.2 Inference

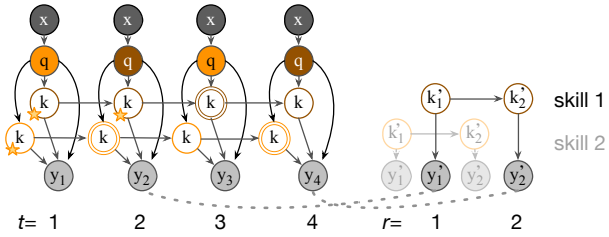
Pointwise Gibbs Sampling is appropriate for some nodes. For example, we can easily sample the topic mixture Q^m for item m. Let's define the S-dimensional vector $\tilde{\boldsymbol{\alpha}}^{\boldsymbol{m}}$ such that each entry $\tilde{\alpha}^m_{(s)}$ is the empirical count of item m being assigned to skill s:

$$\tilde{\alpha}^m_{(s)} = \sum_u^U \sum_t^{T_u} \delta(q_{u,t}, s) \cdot \delta(x_{u,t}, m) \qquad (1)$$

Where $\delta(a, b)$ is the Kronecker function that is 1 iff $a = b$, or 0 otherwise. We sample Q^m as:

$$Q^m \sim \text{Dirichlet}(\underbrace{\tilde{\boldsymbol{\alpha}}^{\boldsymbol{m}}}_{\text{empirical count}} + \underbrace{\boldsymbol{\alpha}^{\boldsymbol{m}}}_{\text{prior}})$$

The other parameters can be sampled similarly. To sample the skill nodes, we use the student model and the topic model— the item to skill mixture. We justify sampling from the student model with an example. Suppose we are using Topical HMM to model data from two skills, multiplication and subtraction. Imagine a student who is an expert at subtraction, but a novice in multiplication. If this student gets an item wrong, it is likely that the item is using a skill the student does not know, in this case, multiplication. Therefore the student model can inform on the inference of the item to skill estimates. Thus, we sample:



(a) Unrolled model, $S = 2$ (b) HMM for skill 1

Figure 2: Knowledge sampling example, $q_2 = 1, q_4 = 1$

Algorithm 2 Knowledge sampling algorithm

Require: $y^s_{u,1} \dots y^s_{T_u}$, parameters E and L

1: **function** SAMPLE KNOWLEDGE(skill s, stud. u)
2: $y'_1 \dots y'_R \leftarrow$ filter_skill$(s, y^s_{u,1} \dots y^s_{u,T_u})$
3: $\gamma_1 \dots \gamma_R \leftarrow p(k'_1 \dots k'_R | y'_1 \dots y'_T; E^s, L^s)$
4: **for** $t \in 1 \dots T_u$ **do**
5: $r \leftarrow$ translate(t)
6: $p(k^s_{u,t} = \tilde{k}) \propto \begin{cases} \gamma_{1,\tilde{k}} & \text{if } r = 0^\star \\ \gamma_{r,\tilde{k}} & \text{if } q_{u,t} = s \\ \delta(k^s_{u,t-1}, \tilde{k}) & \text{otherwise}^\circledcirc \end{cases}$

$$p(q_{u,t} = \tilde{q}) \propto \sum_{q'} \underbrace{p(y_{u,t} | k^{q'}_{u,t}, E^{q', k^{q'}_{u,t}})}_{\text{student model}} \underbrace{p(\tilde{q} | Q^{x_{u,t}})}_{\text{topic model}} \qquad (2)$$

Here $E^{q', k^{q'}_{u,t}}$ is the emission probability for skill q' for the current level of mastery of the student.

Algorithm 2 describes how we infer the posterior probability of the knowledge nodes, and we illustrate it with an example in Figure 2. We show q nodes as visible, because we suppose that we have sampled them already ($q_2 = 1$, $q_1 = 2$, $q_3 = 2$, $q_4 = 1$). For a skill s:

1. We first build a sequence with only the time steps where the knowledge nodes are allowed to change their value because the skill is being practiced (Line 2). Figure 2b shows how such sequence may look like.
2. We calculate the maximum likelihood of the latent states of the newly-built sequence using the Forward-Backward algorithm for HMMs (Rabiner and Juang, 1986) (Line 3).

3. Finally, we sample Topical HMM's knowledge states (Lines 4-6):
   - The first time skill s is practiced: we sample the student knowledge with the probability of initial knowledge (nodes with a star $\star$).
   - Practicing skill s: we sample from the HMM's forward-backward probabilities.
   - Otherwise, we deterministically use the previous knowledge (nodes with two lines $\circledcirc$).

During prediction, we can infer the posterior distribution of the performance by sampling directly from the emission probability, $y_{u,t} \propto E^{s,k^s_{u,t}}$, where $s = q_{u,t}$. Alternatively, we can marginalize out the student knowledge and skill nodes:

$$p(y_{u,t} = \tilde{y}) \propto \sum_{s'} \sum_{l'} p(q_{u,t}|Q^{s'})p(\tilde{y}|E^{s',l'}) \quad (3)$$

In preliminary experiments we did not find any significant differences between these strategies, possibly because we collected a large number of samples. We only report the uncollapsed sampling results.

2.3 ItemClue

In their seminal work, Chi et al. (1981) suggest that novice students categorize items by surface features, such as "words in problem text." On the other hand, more seasoned students group items that require the same principle together, such as "conservation of momentum".

Here, we operationalize interpretability as skills that are cohesive on their surface features. We argue that defining a function to parse surface features is less labor intensive than annotating each item from a potentially very large pool. Our novel method, ItemClue, builds a Bayesian prior to bias similar items together in Topical HMM. ItemClue uses an expert-defined feature extraction function to quantify the similarity between a pair of items. It then uses the output of a clustering algorithm as a prior to bias the estimation of Topical HMM towards clusterings that group similar items together. However, unlike prior work that requires experts to annotate each of the thousands of items that the system may have, we only require experts to create a feature extraction function.

Algorithm 3 describes how to build an ItemClue prior for Topical HMM: it inputs items $x_1 \dots x_M$, a feature extraction function to specify the similarity of the items, the number of clusters S, and a bias intensity parameter of how much the item similarity should influence discovery of the item to skill mappings. Lines 2-4 build a similarity matrix comparing each item to each other using Euclidean distance and a feature extraction function. Line 5 uses clustering to group similar items together into S clusters. Finally, lines 6–11 build a Dirichlet prior for Topical HMM.

Algorithm 3 Item Clue

```
1:  function ItemCluePrior(Item text x_1,...x_M,
    feature extraction function f, number of clusters
    S, intensity a)
2:      for each item i = 1...t_M do
3:          for each item j = 1...t_M do
4:              D_{i,j} = distance(f(t'_i), f(t'_j))
5:      ⟨x_i → cluster c_i⟩ = cluster(D, S)
6:      for each mapping x_i → c_i do
7:          for each skill s ← 1...S do
8:              if s = c_i then
9:                  α^i_(s) ← a
10:             else
11:                 α^i_(s) ← 1
        return α
```

We consider extracting surface features from the correct answer (Li et al., 2013) and the text of the item (Karlovčec et al., 2012). In preliminary experiments we do not find substantive differences in predictive performance of these two strategies. Therefore, we just report results on the correct answer. For a Math tutor, we parse the correct answers by replacing the whole and fractional parts of numbers with N, and v for variables. For example, a correct response that is $-2.5y + 100$ becomes $-N.Nv + N$. We build a bag of letter n-grams ($n = 2, 3$) from the parsed text.

3 Empirical Evaluation

Student models are typically evaluated with a classification evaluation metric that assesses their forecasts of whether a student will answer an item correctly (Dhanani et al., 2014). However, high accuracy is not a sufficient condition for a model to be useful for personalizing education.

Consider the *item difficulty classifier* that makes predictions based on the item difficulty. This is, it estimates the likelihood of a student answering an item correctly as the fraction of correct answers of an item in the training set. This classifier is not a function of practice opportunities: its decision boundary is optimized to predict always "correct answer" or "incorrect answer" independently of amount of practice. Therefore, such classifier does not signal *when* students acquire knowledge, and is not useful for adaptivity. We aspire to have models that are useful for adapting tutoring decisions.

In this section, we evaluate our methods with classification metrics following conventional practice (§ 3.1). To address our concerns with classification metrics, we

then select the most accurate method and investigate if its parameters may be useful for adaptivity (§ 3.2). Lastly, we report on interpretable item to skill mappings (§ 3.3).

3.1 Model Comparison

We evaluate on two datasets from the PSLC Datashop (Koedinger et al., 2010), a repository for educational data. In these datasets, each observation is labeled with problem and step identifiers. We conduct our analysis on the step level by assigning a unique identifier to them. The *Carnegie Learning Bridge to Algebra Cognitive Tutor*® prepares students to an Algebra I class. This dataset has an item bank of 5,233 different items and 123 students. Each student answered an average of 340.73 items (st. dev. 102, min. 48, max. 562, median 341), for a total of 41,910 observations. The *Carnegie Learning Algebra I Cognitive Tutor* is a first-year Algebra course for core instruction. This dataset includes data from 205 students and an item bank of 3,081 questions. Each student answered an an average of 373.37 items (st. dev. 286, min. 3, max. 989, median 314), for a total of 73,181 observations. Both datasets are very unbalanced — over 75% of the items were answered correctly. We split the datasets into 3 sets of non-overlapping students: a training set, a development set to tune hyper-parameters, and a test which we queried only once. No tuning was done on the test set, and because it only contains unseen students, it is a harder prediction task.

We evaluate our student models as the classification task of predicting future student performance. We observe the history preceding the time step we want to predict. For example, to predict on the third time step, we use the data up to the second time step. We evaluate the model predictions using the *Area Under the Curve* (AUC) of the Receiver Operating Characteristic (ROC) curve. The AUC assigns random chance a score of 0.5 and a perfect classifier a 1.

We encode two biases in the priors' hyper-parameters α, τ and ω: (i) *Practice helps learning, and there is no forgetting.* We favor students transitioning to a level of better performance, and not going back to the previous level. For this, we tune for the combinations of magnitudes of this effect for $\tau = 10, 100$ and $\omega = 10, 100$. (ii) *Sparse item to skill mapping.* For our experiments with Topical HMM, we encourage sparsity on the item to skill parameter (Q), motivated by the assumption that each item uses only a few skills. We set $\alpha = 0.1$.

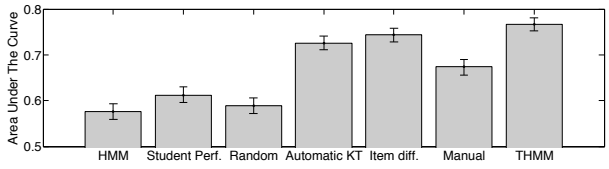
We compare student models and classifiers:

- **HMM**. Can we find evidence of multiple skills? Our methods should perform better than assuming there are no skills in the data.
- **Student Performance**. The likelihood of answering item at time t correctly is the fraction of items answered correctly up to time $t-1$. Intuitively, this is the student "batting average".
- **Random**. Does the item to skill mapping matter? We create a random item to skill mapping with five skills and assign items randomly to one of five categories. We then train Topical HMM to learn the student model (transition and emission probabilities), without updating the item to skill mapping.
- **Item difficulty**. We described this classifier earlier in the section. It is not useful for adaptive tutoring.
- **Domain Experts**. How accurate are experts at creating item to skill mappings? We use Topical HMM with an item to skill mapping previously designed using domain knowledge. We initialize the parameter Q of Topical HMM with the expert model and do not update its values. If the expert decided that an item uses multiple skills, we assign uniform weight to each skill even though the experts may have used a different interpretation (e.g., conjunctive). Modeling multiple skills per item is an active area of research (González-Brenes et al., 2014).
- **Automatic Knowledge Tracing**. We implement Automatic Knowledge Tracing using a Bayesian HMM, and with an existing matrix factorization implementation used in education (Thai-Nghe et al., 2010). We leave experimenting with alternative matrix factorizations algorithms for future work. We tune the hyperparameters in the development set (e.g., the number of skills), this results in four and six skills for the Bridge to Algebra® and the Algebra I datasets, respectively. Unseen items in the test set are set to random skills.
- **Topical HMM**. For Topical HMM, we also tune the hyperparameters and infer five and six skills for the Bridge to Algebra®, and Algebra I, respectively.

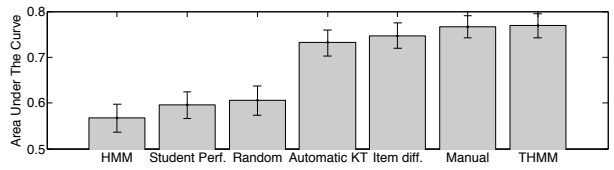
Our optimization method for Topical HMM does not update its beliefs with new observations, requiring a large memory and runtime footprint to re-sample the entire sequence. To speed up computations, in all of our experiments we predict up to the 150^{th} time step in the development set, and up to the 200^{th} time step in the test set. For sampling, we collect 2,000 samples, discarding the first 500 samples.

For all of the methods we pick the best random restart using the development set. Figures 3a and 3b show the AUC of the different methods on the test set of the Algebra I and Bridge to Algebra® datasets. The error bars show the 95% confidence intervals calculated with an implementation of the Logit method[1], which corrects for the non-independence of the points of the

[1] www.subcortex.net/research/code/area_under_roc_curve



(a) Algebra I AUC results



(b) Bridge to Algebra® AUC results

ROC curve.

In both datasets, Topical HMM outperforms all other approaches. In the Algebra I dataset, the difference between the expert model and Topical HMM is statistically significant because the confidence intervals do not overlap. Topical HMM discovers the most predictive item to skill mapping with an AUC of $.77 \pm .01$ (with 95% confidence), while the model handcrafted by experts only achieves an AUC of $.67 \pm .02$; thus, the automatic approach is 15% better than the expert approach. Automatic Knowledge Tracing also performs statistically better than the expert (AUC= .73), but worse than Topical HMM, and the item difficulty (AUC= .74).

In the Bridge to Algebra® dataset, the fully data-driven Topical HMM achieves the same performance as experts using domain knowledge (AUC$\approx$ .77), but requires much less human effort. Unfortunately, we do not have an estimate of how long annotation took, but if we make a conservative estimate that an expert takes two minutes per item to label the skills needed, it would take her around 6 days of non-stop work to process a single dataset. On a 2007 laptop (Intel® Xeon® 3Ghz, with 16Gb of RAM), Automatic Knowledge Tracing requires over 2 hours, while Topical HMM requires 6 hours of computation. Although the higher performance of Topical HMM over Automatic Knowledge Tracing comes with higher computational cost, it is over 24 times faster than a human estimate, yet it achieves the same classification performance in this dataset.

In both datasets, our data-driven models are significantly more accurate than assuming an item to skill mapping with a single skill (HMM), using the student performance (Student Perf.), or assigning items to skills randomly (Random). The random item to skill mapping performs significantly better than chance. We hypothesize that tuning multiple restarts in the development set caused this.

Topical HMM is not meaningfully better than the item difficulty classifier, which can be significantly better than the human experts. Koedinger et al. (2012) argue that classification accuracy is not as important as having an item to skill mapping and a student model. Unfortunately, eventhough the item difficulty classifier has high accuracy, it is not useful for building adaptive tutors, an item to skill mapping or a student model. The item difficulty classifier has limited usefulness for adaptivity because its decision boundary is not a function of how much a student has practiced a skill. Prior work has been only partially successful on using difficulty in improving student models for adaptive tutoring. For example, Khajah et al. (2014) argue that item difficulty confounds with student learning. In the next section we study if our most accurate model, Topical HMM, is suitable for adaptive tutoring.

3.2 Best-Model Drill Down

Table 1: Learned Parameters with Topical HMM in the Bridge to Algebra® datset

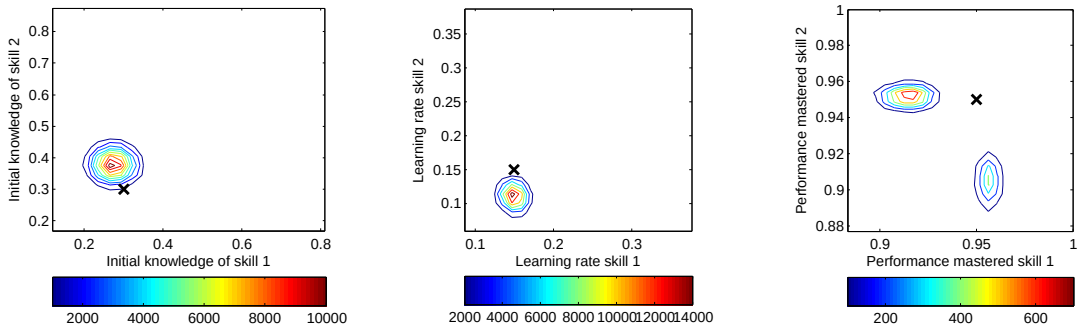
skill	Learn	Forget	L0	Guess	Slip
0	.70	.27	.13	.11	.06
1	.75	.21	.18	.10	.04
2	.79	.20	.43	.08	.06
3	.85	.12	.44	.08	.06
4	.91	.05	.91	.01	.55

Table 1 shows the learned parameters of Topical HMM for the Bridge to Algebra® dataset. These parameters suggest that Topical HMM can be used to infer *when* a student has mastered a skill. Knowing when students have learned content is useful for adaptivity, for example, to stop teaching a skill that the student has already mastered. Again, the item difficulty classifier is not useful for this purpose.

We now investigate whether Topical HMM's recovered parameters are reliable. For this purpose, we construct a synthetic data set with 300 students and 30 items that we assigned to one of two skills. We want synthetic data to be plausible; for example, the probability of answering an item correctly if guessed should be lower than the probability of answering an item correctly if known. Therefore, we hand-crafted some of the behavior of the synthetic students:

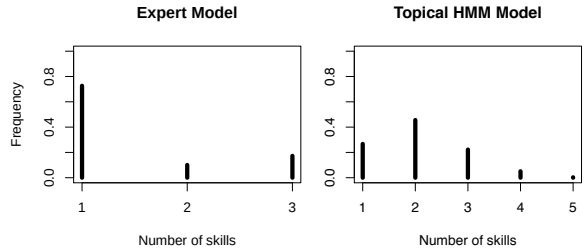
- The initial knowledge probability is 0.3.
- The learning rate is 0.15.
- The (1-guess) probability is 0.95.
- The slip probability is 0.05.

Figure 4 marks with an 'x' the true parameters that generate the synthetic data. We ran 100 different random restarts of our Topical HMM's optimization



(a) initial knowledge L_0 (b) learning rate L (c) emission E

Figure 4: Distribution of estimated parameters using Topical HMM. The 'x' indicates the true parameter value and the contours represent the frequency of values out of 100,000 samples.



(a) Expert model (b) Topical HMM

Figure 5: Histogram of number of skills mapped to an item in the Algebra I dataset

procedure, with 1,000 different samples each. We discard the first 400 for burn-in. For inference in Topical HMM, we set up informationless flat priors: $\alpha = -1$, $\beta = 1$, $\tau = 1$, $\omega = 1$. For the parameters that determine the initial knowledge and learning rate, most of the samples are in the vicinity of the true parameter value. However, for determining the emission probabilities (guess and slip) the sampler often gets stuck in two local optima, each one near the true value of only one of the parameters.

3.3 Interpretability

The item to skill mappings discovered by Topical HMM are not very easy to interpret, mainly because an item is often assigned to multiple skills. For example, Figure 5 counts the number of skills mapped to an item. For this, we take the last sample of the parameter Q, and count the number of entries for every item that has probability higher than a threshold of 0.05. In both datasets, the experts assign most items to a single skill for around 80% of the time. Topical HMM assigns a single skill with a frequency of about 40%, with a mode of 2 skills per item.

We now use ItemClue to bias the estimation towards more interpretable models. We report extracting features from the correct answers of the Algebra I dataset with a strong prior ($a = 10^3$):

i. Positive integer constants, abstracted as N.
ii. Plotting operations, are responses that require students to manipulate a plot.
iii. Simple (1 operator) constant expressions, e.g. $N \cdot N$, $N + N$.
iv. Complex (2+ operator) constant expressions, e.g. $N - N \cdot N$, $N + N \cdot N$.
v. Simple (1 operator) variable expressions, e.g. $N/N \cdot x$, $Nx + N$.
vi. Complex (2+ operator) variable expressions, e.g. $Nx(x + N)$, $x(N.N) + N$.

ItemClue discovers skills automatically, but naming the skills required human analysis. The six clusters discovered are more coarse than the ones discovered by the expert.

While the interpretability of the model increases, the performance of the model decreases. The AUC when ItemClue is not used (intensity $a = 10^0$) is approximately 0.8, and it decreases when using a stronger prior ($a = 10^3$) to 0.65. At the strongest prior ($a = 10^3$), Topical HMM is using only the prior to discover the item to skill mapping.

4 Relation to Prior Work

We now review prior work that uses student performance data to find the item to skill mapping. However, other sources of data are available (Li et al., 2011). Prior methods that used performance data for discovering the skill definition are restricted due to (i) inability to handle longitudinal data or (ii) not being fully automatic:

i. Matrix-based methods (Winters et al., 2005), such as SPARFA (Waters et al., 2013), Non-Negative Matrix Factorization (Desmarais, 2011) and the Q-Matrix Method (Barnes, 2005) are not designed for longitudinal data, and therefore do not model student learning. These methods do not distinguish between poor performance at early time steps and poor performance after a lot of practice. Future work may extend Automatic Knowledge Tracing to use these techniques, instead of the linear factorization we used. Matrix-based techniques often suffer from the cold start problem— they cannot predict on unseen items or students. Automatic Knowledge Tracing can predict on unseen students, and Topical HMM has no such limitations.
ii. Semi-automatic approaches, such as Learning Factors Analysis (Cen et al., 2006), are designed for temporal data, but require the labor-intensive task of having experts to annotate every item of the pool. As we have seen, modern tutoring systems may have thousands of items, and these methods result very costly. Future work may provide a thorough comparison of Learning Factors Analysis and ItemClue.

Promising recent work (Lindsey et al., 2014) allows temporal data but does not allow multiple skills. The authors have contacted us for follow-up comparisons, but their work was not yet published during the preparation of this paper.

Dynamic collaborative filtering techniques have been applied with very limited success. For example, tensor factorization (Thai-Nghe et al., 2010) assumes that student performance changes over time, independently of recent student performance.

We formulate Topical HMM as a hierarchical Bayesian model in which each item is modeled as a mixture over an underlying set of skills. Our formulation of Topical HMM is related to Input-Output HMM (Bengio and Frasconi, 1994) and Factorial HMM (Ghahramani and Jordan, 1997). We now briefly discuss these approaches. The Input-Output HMM, as well as the conventional HMM, is tractable only with a relatively small number of states: to represent b bits of information about the history of a time sequence, an Input-Output HMM would need 2^b distinct states (Ghahramani and Jordan, 1997). A Factorial HMM works around this exponentially large number of states with a distributed state representation that can achieve the same task with b binary state variables. However, Factorial HMMs only model an output sequence. Topical HMM combines concepts from both Input-Output HMMs and Factorial HMMs: it uses a distributed state representation and is able to map input sequences to output sequences.

5 Conclusion

We present fully data-driven methods that discover how items map into skills and when students master them. Our main contributions are the novel Automatic Knowledge Tracing and ItemClue algorithms, a new optimization technique for Topical HMM, and a novel evaluation of these techniques.

We make substantial progress from what was possible – we are unaware of prior methods that can discover an item to skill mapping from longitudinal performance data automatically. Our three methods have different trade-offs of complexity, interpretability, accuracy and human effort required. Topical HMM is fully automatic and has the best prediction accuracy at a higher computational cost and low interpretability. Automatic Knowledge Tracing has the best runtime performance, yet it only has a minor decrease of accuracy. ItemClue finds the most interpretable models at the expense of accuracy and the requirement of low cost domain knowledge.

The gain of our data-driven model seems dependent on the domain. For example, in the pre-algebra dataset, our automatic approaches perform similar to the expert model, albeit with less human effort. However, on the Algebra I dataset, our best method improves by 15% the expert model. A secondary contribution is that we provide evidence that it is possible to build high accuracy classifiers that are not appropriate for personalizing education. Our experiment suggests that Topical HMM's may recover the student model parameters, and therefore, may be useful for personalizing education. In future work we will study evaluation techniques for adaptive algorithms.

Follow-up work may improve Topical HMM's optimization algorithm to allow efficient online updates. A multi-skill model is not trivial to update because the assignment of an item to skill may change when further performance evidence is presented (Hooker et al., 2009). We believe our methods are suited for guiding tutoring activities, but not high-stakes assessments. Future work may evaluate our methods in a controlled experiment or with simulations (Lee and Brunskill, 2012).

Our algorithms are relevant to the topic modeling, graphical models, and collaborative filtering communities. Collaborative filtering tasks are often evaluated on movie datasets (Koren et al., 2009), eventhough their usefulness in this domain is disputed (Vanderbilt, 2013). We are looking forward for future applications of collaborative filtering techniques in education as well as cross-domain comparisons.

References

Barnes, T. (2005). The Q-matrix method: Mining student response data for knowledge. In Beck, J., editor, *Proceedings of AAAI 2005: Educational Data Mining Workshop*, pages 978–980, Pittsburgh, PA.

Beck, J. (2007). Difficulties in inferring student knowledge from observations (and why you should care). In *Educational Data Mining: Supplementary Proceedings of the 13th International Conference of Artificial Intelligence in Education*, pages 21–30, Marina del Rey, CA.

Bengio, Y. and Frasconi, P. (1994). An input output hmm architecture. In Tesauro, G., Touretzky, D. S., and Leen, T. K., editors, *NIPS*, pages 427–434. MIT Press.

Bergner, Y., Droschler, S., Kortemeyer, G., Rayyan, S., Seaton, D., and Pritchard, D. (2012). Model-based collaborative filtering analysis of student response data: Machine-learning item response theory. In Yacef, K., Zaane, O. R., Hershkovitz, A., Yudelson, M., and Stamper, J. C., editors, *Educational Data Mining*, pages 95–102.

Cen, H., Koedinger, K., and Junker, B. (2006). Learning factors analysis: A general method for cognitive model evaluation and improvement. In Ikeda, M., Ashley, K., and Chan, T.-W., editors, *Intelligent Tutoring Systems*, volume 4053 of *Lecture Notes in Computer Science*, pages 164–175. Springer Berlin / Heidelberg.

Chi, M. T., Feltovich, P. J., and Glaser, R. (1981). Categorization and representation of physics problems by experts and novices. *Cognitive Science*, 5(2):121 – 152.

Corbett, A. and Anderson, J. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4(4):253–278.

Desmarais, M. (2011). Conditions for Effectively Deriving a Q-Matrix from Data with Non-negative Matrix Factorization. In M. Pechenizkiy and T. Calders and C. Conati and S. Ventura and C. Romero and J. Stamper, editor, *Proceedings of the 4th International Conference on Educational Data Mining*, pages 169–178, Eindhoven, Netherlands.

Dhanani, A., Lee, S. Y., Phothilimthana, P., and Pardos, Z. (2014). A comparison of error metrics for learning model parameters in bayesian knowledge tracing. Technical Report UCB/EECS-2014-131, EECS Department, University of California, Berkeley.

Ghahramani, Z. and Jordan, M. (1997). Factorial Hidden Markov Models. *Machine learning*, 29(2):245–273.

González-Brenes, J., Huang, Y., and Brusilovsky, P. (2014). General Features in Knowledge Tracing: Applications to Multiple Subskills, Temporal Item Response Theory, and Expert Knowledge. In Mavrikis, M. and McLaren, B. M., editors, *Proceedings of the 7th International Conference on Educational Data Mining*, London, UK.

González-Brenes, J. P. and Mostow, J. (2012). Dynamic Cognitive Tracing: Towards Unified Discovery of Student and Cognitive Models. In Yacef, K., Zaïane, O. R., Hershkovitz, A., Yudelson, M., and Stamper, J. C., editors, *Proceedings of the 5th International Conference on Educational Data Mining*, pages 49–56, Chania, Greece.

González-Brenes, J. P. and Mostow, J. (2013). What and When do Students Learn? Fully Data-Driven Joint Estimation of Cognitive and Student Models. In Olney, A., Pavlik, P., and Graesser, A., editors, *Proceedings of the 6th International Conference on Educational Data Mining*, pages 236–240, Memphis, TN.

Gruber, A., Weiss, Y., and Rosen-Zvi, M. (2007). Hidden topic markov models. In *International Conference on Artificial Intelligence and Statistics*, pages 163–170.

Hooker, G., Finkelman, M., and Schwartzman, A. (2009). Paradoxical results in multidimensional Item Response Theory. *Psychometrika*, 74(3):419–442.

Karlovčec, M., Córdova-Sánchez, M., and Pardos, Z. (2012). Knowledge component suggestion for untagged content in an intelligent tutoring system. In Cerri, S., Clancey, W., Papadourakis, G., and Panourgia, K., editors, *Intelligent Tutoring Systems*, volume 7315 of *Lecture Notes in Computer Science*, pages 195–200. Springer Berlin Heidelberg.

Khajah, M., Huang, Y., González-Brenes, J. P., Mozer, M. C., and Brusilovsky, P. (2014). Integrating knowledge tracing and item response theory: A tale of two frameworks. In *In submission*.

Koedinger, K. R., Baker, R. S. J., Cunningham, K., Skogsholm, A., Leber, B., and Stamper, J. (2010). A data repository for the EDM community: The PSLC DataShop. In Romero, C., Ventura, S., Pechenizkiy, M., and Baker, R., editors, *Handbook of Educational Data Mining*, pages 43–55, Boca Raton, FL. CRC Press.

Koedinger, K. R., Corbett, A. T., and Perfetti, C. (2012). The knowledge-learning-instruction framework: Bridging the science-practice chasm to en-

hance robust student learning. *Cognitive Science*, 36(5):757–798.

Koren, Y., Bell, R., and Volinsky, C. (2009). Matrix Factorization Techniques for Recommender Systems. *Computer*, 42(8):30–37.

Lee, J. I. and Brunskill, E. (2012). The impact on individualizing student models on necessary practice opportunities. In Yacef, K., Zaïane, O. R., Hershkovitz, A., Yudelson, M., and Stamper, J. C., editors, *Proceedings of the 5th International Conference on Educational Data Mining*, pages 118–125, Chania, Greece.

Li, N., Cohen, W., and Koedinger, K. (2013). Discovering student models with a clustering algorithm using problem content. In D'Mello, S. and Calvo, R. A., editors, *Proceedings of the 6th International Conference on Educational Data Mining*, pages 98–105, Memphis, TN.

Li, N., Cohen, W. W., Koedinger, K. R., and Matsuda, N. (2011). A machine learning approach for automatic student model discovery. In M. Pechenizkiy and T. Calders and C. Conati and S. Ventura and C. Romero and J. Stamper, editor, *Proceedings of the 4th International Conference on Educational Data Mining*, pages 31–40, Eindhoven, Netherlands.

Lindsey, R. V., Khajah, M., and Mozer, M. C. (2014). Automatic discovery of cognitive skills to improve the prediction of student learning. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K., editors, *Advances in Neural Information Processing Systems 27*, pages 1386–1394. Curran Associates, Inc.

Rabiner, L. and Juang, B. (1986). An introduction to Hidden Markov Models. *ASSP Magazine, IEEE*, 3(1):4–16.

Thai-Nghe, N., Drumond, L., Krohn-Grimberghe, A., and Schmidt-Thieme, L. (2010). Recommender system for predicting student performance. *Procedia Computer Science*, 1(2):2811 – 2819.

Vanderbilt, T. (2013). The science behind the netflix algorithms that decide what you'll watch next. Wired Magazine. Last retrieved June/6/2014. `http://www.wired.com/2013/08/qq_netflix-algorithm/`.

VanLehn, K. (1988). Student modeling. In Polson, M. C. and Richardson, J. J., editors, *Foundations of Intelligent Tutoring Systems*, pages 55–78. Erlbaum, Hillsdale, NJ.

Waters, A. E., Lan, A. S., and Studer, C. (2013). Sparse probit factor analysis for learning analytics. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 8776–8780. IEEE.

Winters, T., Shelton, C., Payne, T., and Mei, G. (2005). Topic extraction from item-level grades. In Beck, J., editor, *American Association for Artificial Intelligence 2005 Workshop on Educational Datamining*, Pittsburgh, PA.