Towards Automatic Discovery of Prerequisite Structure of Skills from Student Performance Data

Leave Authors Anonymous
for Submission
City, Country
e-mail address

Leave Authors Anonymous
for Submission
City, Country
e-mail address

Leave Authors Anonymous
for Submission
City, Country
e-mail address

ABSTRACT

Understanding the prerequisite structure of skills is crucial for designing curriculum, assessing mastery and for student modeling. Automatic discovery of the prerequisite structures from educational data is intriguing yet challenging since student's mastery of skills are latent variables. In this work, we proposed a novel data-driven approach to discover the prerequisite structure of skill given student performance on test items. By modeling the prerequisite relations as a Bayesian network, we then estimate the causal structure and the probabilistic dependence among the skills via a two-stage structural learning algorithm. In the first stage, the skeleton of the Bayesian network is constructed using Structural Expectation Maximization (Structural EM) algorithm; In the second stage, the edges are oriented by enforcing the constraints on estimated conditional probability distributions. We validate the proposed approach using simulations and by post-hoc analysis of student data. We show the discovered prerequisite structure can improve the student model in predicting student performance.

Keywords

ACM proceedings, LaTeX, text tagging

1. INTRODUCTION

Students learn much better when the skills are not randomly introduced but organized in a meaningful order which starts from relatively simple concepts and gradually introduces more complex ones. Further, among these skills, some are preliminary of others such that they must be mastered before the subsequent concepts can be learned. For instance, students have to know how to do addition before they learn to do multiplication. In this work, we use prerequisite structure to refer to the relationships among skills that place strict constraints on the order in which these skills can be acquired. Determining the prerequisite relations among skills is crucial for designing curriculum and for assessing mastery.

Most prerequisite structures of skills are specified by domain or cognition experts. However, it is time-consuming and different experts may disagree on the prerequisite structure of the same set of skills. Further, the prerequisite structures specified by the experts are seldom tested and might be unreliable in the sense that experts may have "blind spots".

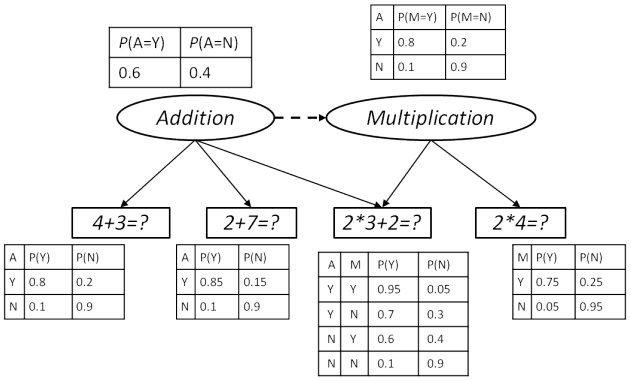


Figure 1: A hypothetical Bayesian network learned with Algorithm 1. Solid edges are given by the item to skill mapping, dashed edges between skill variables are to be discovered from data. The conditional probability tables are to be learned.

1.1 Learning an Intervention Model

REMIND learns the prerequisite structure of the skills using data with a statistical model called Bayesian network [12, 15]. Bayesian networks are also called probabilistic graphical models because they can be represented visually and algebraically as a collection of nodes and edges. A tutorial description of Bayesian networks in education can be found elsewhere [1], but for now we say that they are often described with two components: the nodes represent the random variables, which we describe using *conditional probability tables* (CPTs), and the set of edges that form a *directed acyclic graph* (DAG) represent the conditional dependencies between the variables. Bayesian networks are a flexible tool that can be used to model an entire curriculum.

Figure 1 illustrates an example of a prerequisite structure modeled with a Bayesian network. Here, we relate four test items with the skills of addition and multiplication. Addition is a prerequisite of multiplication thus there is an arrow from addition to multiplication. Modeling prerequisites as edges in a Bayesian network allows us to frame the discovery of the prerequisite relationships as the well-studied machine learning problem of learning a DAG (with the presence of latent variables).

Algorithm 1 describes the simple but effective REMIND pipeline. Suppose we collect data from n students, answering p items. Then, the input of REMIND is a matrix $\mathbf{D}$ with $n \times p$ dimensions, an item to skill mapping, and (optionally) some constraints on what content can trigger a remediation. Each entry in $\mathbf{D}$ encodes the performance of a student (see Table 1 for an example). REMIND first constructs the prerequisite relationships among the set of skills using constraints. Then, it learns the parameters of a student model that infers what skills a student has mastered.

Table 1: Example data matrix to use with REMIND. The performance of a student is encoded with 1 if the student answered correctly the item, and 0 otherwise.

User	Item 1	Item 2	Item 3	Item p
Alice	0	1		0
Bob	1	1	...	1
Carol	0	0		1
		...		

REMIND relies on a popular machine learning algorithm called Structural Expectation Maximization (EM), which has not been used in educational applications. A secondary contribution of our work is introducing Structural EM for learning Bayesian network structures from educational data. One of the advantages of Structural EM algorithm over prior work is that it allows to combine expert beliefs into the inference process. We now describe the steps of REMIND in detail.

Algorithm 1 The REMIND algorithm

Require: A matrix $\mathbf{D}$ of student performance on a set of test items, skill-to-item mapping Q (containing a set of skills $\mathbf{S}$) and a set of constraints $\mathbf{C}$ reflecting experts' beliefs on the prerequisite structure

1: $G_0 \leftarrow \text{Initialize}(\mathbf{S}, Q, \mathbf{C})$ } Initialization
2: $i \leftarrow 0$ }
3: **do**
4: E-step:
5: $\theta_i^* \leftarrow \text{ParametricEM}(G_i, \mathbf{D})$
6: $\mathbf{D}_i^* \leftarrow \text{Inference}(G_i, \theta_i^*, \mathbf{D})$
7: M-step:
8: $\langle G_{i+1}, \theta_{i+1} \rangle \leftarrow \text{BNLearning}(G_i, \mathbf{D}_i^*, \mathbf{C})$
9: $i \leftarrow i+1$
10: **while** Stop criteria is not met
11: $M \leftarrow \text{LearnStudentModel}(G_i, \theta_i, \mathbf{D})$ } Student modeling

Learn prerequisites

1.1.1 Initial Bayesian Network

REMIND represents the prerequisite structure using Bayesian networks that use latent variables to represent the student knowledge of a skill, and observed variables that represent the student performance answering items (e.g, correct or incorrect). We first create an initial Bayesian network that complies to the skill-to-item Q-matrix and a set of constraints reflecting experts' belief on the prerequisite structure (step 1 of Algorithm 1). That is, we create an arc to each item from each of its required skills. Further, if we believe one skill is the direct prerequisite of another skill, we create an arc between them, otherwise we leave all skill variables disconnected. With the created Bayesian network as an initial network, we learn the arcs between the skill variables using Structural EM.

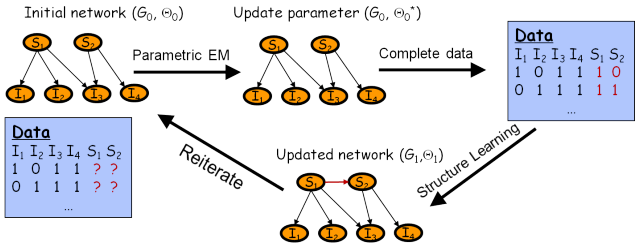


Figure 2: An illustration of the Structure EM algorithm to discover the structure of the latent variables.

1.1.2 Learn prerequisite structure from data

A common solution to learning a Bayesian network from data is the score-and-search approach [4, 9]. This approach uses a scoring function to measure the fitness of a Bayesian network structure to the observed data, and manages to find the optimal model in the space of all possible Bayesian network structures. However, the conventional score-and-search approaches rely on efficient computation of the scoring function, which is only feasible for problems where data contains observations for all variables in the Bayesian network. Unfortunately, our domain has skill variables that are not directly observed. An intuitive work-around is to use the Expectation Maximization (EM) to estimate the scoring function. However, EM in this case takes a large number (hundreds) of iterations to converge and each iteration requires Bayesian network inference, which is computationally prohibitive. Further, we need run EM for each candidate structure. The number of possible Bayesian network structures is super-exponential with respect to the number of nodes. The Structural Expectation Maximization algorithm [6, 7] is an efficient alternative.

Structural EM is an iterative algorithm that inputs a matrix $\mathbf{D}$ of student performance (see example Table 1). Figure 2 illustrates one iteration of the Structural EM algorithm. The relevant steps are also sketched in Algorithm 1. Each iteration consists of an Expectation step (*E-step*) and a Maximization step (*M-step*). In *E-step*, we first find the maximum likelihood estimate θ^* of the parameters for the current structure G calculated from previous iteration using parametric EM.[1] We then do Bayesian inference to compute the expected values for the hidden variables using the current model (G, θ^*) and use the values to complete the data. In the *M-step*, we use the conventional score-and-search approach to optimize the structure according to the completed data. Since the space of possible Bayesian network structures is super-exponential, exhaustive search is intractable and local search algorithms, such as greedy hill-climbing search, are often used. The *E-step* and *M-step* interleave and iterate until some stop criteria is met, e.g., the scoring function does not change significantly. Contrast to the conventional score-and-search algorithm, Structural EM runs EM only on one structure in each iteration, thus is computationally more efficient.

REMIND's initialization step fixes the arcs from skills to items according to the Q-matrix. In the *M-step* of our Structural EM implementation we only consider the candidate structures that comply with the Q-matrix.

An advantage of using Structural EM to discover the prerequisite

[1] In the first iteration, the current network is created from the initialization step.

relationship of skills, is that it is easily extensible to incorporate domain knowledge. For example, we can place constraints on the output structure to force or to disallow a skill to be a prerequisite from another other skill. Consider, an intelligent tutor that teaches the content of a book. The book content structure provides a natural ordering of the chapters. We may use the book structure to engineer domain knowledge that an introductory chapter cannot be remediated with content that appears later in the book.

1.1.3 Learning a student model

A statistical model that infers whether or when a skill is mastered by a student is often called a student model. We now compare remedial strategies in two contexts:

- § 1.1.3 discusses training a measurement model where students solve an evaluation that test their competencies. Depending on their performance, students are suggested different remediation tasks.
- § **??** discusses learning remedial intervention that are suggested while the student is still being tutored.

Assessment. Bayesian networks have been used for a long time in educational assessment. For example, in one of their first usages in measurement [16], researchers manually designed a network to diagnose students depending on how they solve addition problems. However, networks are often designed by a subject matter expert, while REMIND discovers them automatically from data. We can use standard techniques, like the EM algorithm, to learn the conditional probability tables of the prerequisite structure Bayesian network.

1.2 Using an Intervention Model

We envision two use cases of REMIND. We may use student data collected from an assessment instrument, like a quiz or an exam, for REMIND to decide whether to give a remedial intervention. Alternatively, a tutoring system may use REMIND to decide if a student should do some remediation. For both use cases, REMIND uses the student model (either a Bayesian Network or Prerequisite PFA) to infer the posterior probability of a student mastering the skills and prerequisites. If the probability of a student knowing a prerequisite is below a threshold, REMIND suggests remediation to a student.

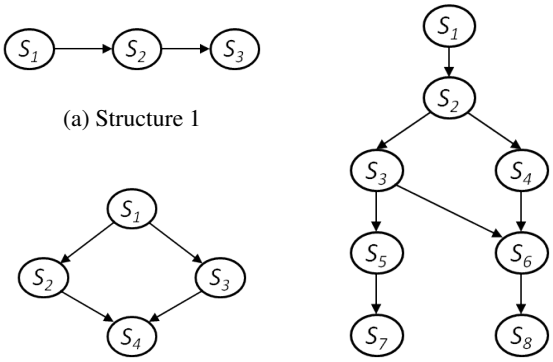
For example consider similar data to Table 1, where some of the entries are missing if a student has not answered an item. We can use a Bayesian network to calculate the posterior probability of student competencies on skills and their prerequisites.

2. EVALUATION

In § 2.1, we evaluate REMIND with simulated data to assess the quality of the discovered prerequisite structures. Then, in § 2.2 we use data collected from real students.

2.1 Simulated Data

Synthetic data allows us to study how REMIND compares to ground truth. For this, we engineered three prerequisite structures (DAGs), shown in Figure 3. Here, each figure represents different causal relations between the simulated latent skill variables. Given observational data, the direction of some edges cannot be determined



(a) Structure 1

(b) Structure 2

(c) Structure 3

Figure 3: Three different DAGs between latent skill variables. Item nodes are omitted.

because the edges are *reversible*[2]. We represent these edges as undirected lines.

For clarity, Figure 3 omits the item nodes; but each skill node is parent of six item variables. All of these nodes are modeled using binary random variables. More precisely, the latent nodes represent whether the student achieves mastery of the skill, and the observed nodes indicate if the student answers the item correctly. Notice that these Bayesian networks include the prerequisite structures as well as the skill-item mapping.

We consider simulated data with different number of observations ($n = 150, 500, 1000, 2000$). For each sample size and each DAG, we generate ten different sets of conditional probability tables randomly, with two constraints. First, we enforce that achieving mastery of the prerequisites of a skill will increase the likelihood of mastering the skill. Second, mastery of a skill increases the probability of student correctly answering the test item. Thus, in total we generated 120 synthetic datasets (3 DAGs x 4 sample sizes x 10 CPTs), and report the average results.

We evaluate how REMIND can discover the true prerequisite structure using metrics designed to evaluate Bayesian networks structure discovery. In particular, we use the F_1 *adjacency score* and the F_1 *orientation score*. The adjacency score measure how well we can recover connections between nodes. It is a weighted average of the true positive adjacency rate and the true discovery adjacency rate. On the other hand, the orientation score measures how well we can recover the direction of the edges. It is calculated as a weighted average of the true positive orientation rate and true discovery orientation rate. This metric does not account for the directionality incorrect edges that are reversible. In both cases, the F_1 score reaches its best value at 1 and worst at 0. Moreover, for comparison, we compute the F_1 *adjacency score* for Bayesian network structures whose skill nodes are fully connected with each other. These fully connected DAGs will serve as baselines for evaluating the adjacency discovery.[3] For completeness, we list these formulas in tables 2 and 3,

[2] Bayesian network theory states that some Bayesian networks are statistically equivalent. These networks have the same skeleton and the same *v*-structures. A *v*-structure in a Bayesian network G is an ordered triple of nodes (u, v, w) such that G contains the directed edges $u \to v$ and $w \to v$ and u and w are not adjacent in G. [17].

[3] We do not compute F_1 orientation score for fully connected DAGs

respectively.

Table 2: Formulas for measuring adjacency rate (AR)

Metric	Formula
True positive (*TPAR*)	$\frac{\text{\# of correct adjacencies in learned model}}{\text{\# of adjacencies in true model}}$
True discovery (*TDAR*)	$\frac{\text{\# of correct adjacencies in learned model}}{\text{\# of adjacencies in learned model}}$
F_1-*AR*	$\frac{2 \cdot TPAR \cdot TDAR}{TPAR + TDAR}$

Table 3: Formulas for measuring orientation rate (OR)

Metric	Formula
True positive (*TPOR*)	$\frac{\text{\# of correctly directed edges in learned model}}{\text{\# of directed edges in true model}}$
True discovery (*TDOR*)	$\frac{\text{\# of correctly directed edges in learned model}}{\text{\# of directed edges in learned model}}$
F_1-*OR*	$\frac{2 \cdot TPOR \cdot TDOR}{TPOR + TDOR}$

We use these metrics to evaluate the effect of varying the number of observations of the training set (sample size) on the quality of learning the prerequisite structure. We designed experiments to specifically answer the following two questions:

1. How useful is the prior domain knowledge for improving the prerequisite structure discovery? We are interested in the case where prior knowledge on the node ordering is provided by experts.
2. How well does the algorithm perform when there is noise in the data? We focus on studying noise due to the presence of unaccounted hidden variables.

We now investigate these questions.

2.1.1 *Single-skill vs Multi-skill Items*

Figure 4 compares the F_1 of adjacency discovery and edge orientation result of two conditions. In the *without ordering* condition, we learn a Bayesian network only using data. In the *with ordering* condition, we provide the node ordering of the true Bayesian network as constraints. The purpose of this condition is to simulate the effect of a subject matter expert that provides useful constraints.

We observe that the accuracy for both the adjacency and the edge orientation improves with the amount of data. With just 2000 observations, the algorithm can recover the true structures almost perfectly. Additionally, using domain knowledge constraints significantly improves the edge orientation accuracy. In particular, it substantially reduces the *FPOR* and *FNOR* (Figure **??**).

2.1.2 *Sensitivity to Noise*

Real-world data sets often contain various types of noise. For example, noise may occur due to hidden variables that are not explicitly modeled. To evaluate the sensitivity of REMIND to noise, we synthesize Bayesian networks including a *StudentAbility* node that takes three possible states (low/med/high). In these Bayesian networks, students' performance depends not only on whether they have mastered the skill, but also on their individual ability. We first simulated data from Bayesian networks that have a *StudentAbility* variable to generate "noisy" data samples, and then use this data to recover the prerequisite structure. Figure 5 illustrates the procedure of this sensitivity analysis experiment.

because all edges in a fully connected DAG are reversible.

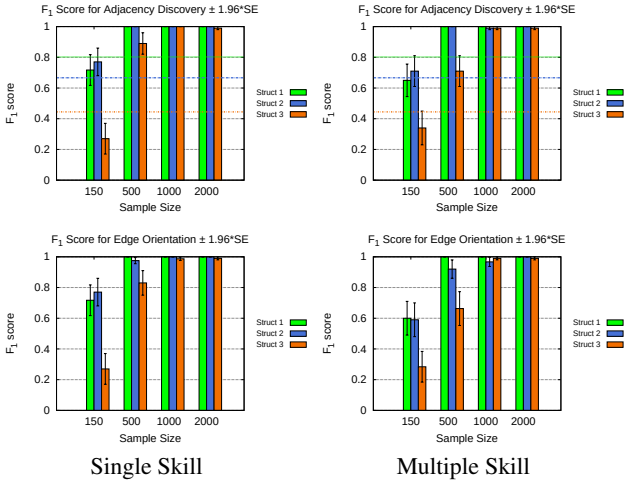


Figure 4: Comparison of F_1 scores for adjacency discovery (top row) and for edge orientation (bottom row). Horizontal lines are baseline F_1 scores computed for fully connected (complete) Bayesian networks.

Figure 6 compare the results were noise was introduced or not. Interestingly, the noise does not harm the learning accuracy at all, and actually improves the accuracy. We hypothesize that the existence of additional hidden variable increases the variance of the data which can help the structure learning.

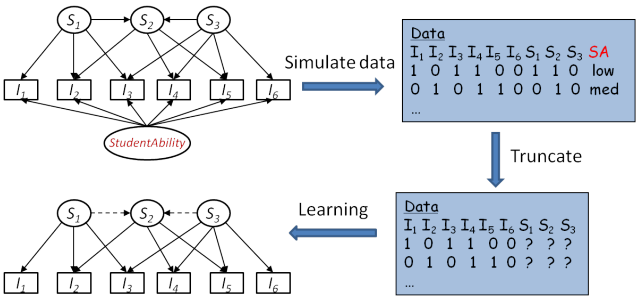


Figure 5: Evaluation of REMIND with noisy data

2.1.3 *In Presence of Missing Data*

Real-world student performance data often has missing data, i.e., some students do not respond to all test items. Structural EM can also be applied with the missing data. The general idea is, in the E-step, in addition to completing the values for hidden skill variables, the algorithm has to complete the missing data points. In this experiment, we generated data sets of size 1000 with varying fraction of missing data (10%, 20%, 30%, 40%, 50%). We ran our algorithm to recover the BNs. The results are presented in .

2.1.4 *Comparison With PARM Method*

2.2 Real Student Performance Data

We now evaluate REMIND using data collected from a commercial non-adaptive tutoring system. We use data from anonymized students interacting with two implementations of a seventh grade math curriculum. One implementation is used by conventional brick-and-mortar schools (we call it `traditional`), and the other

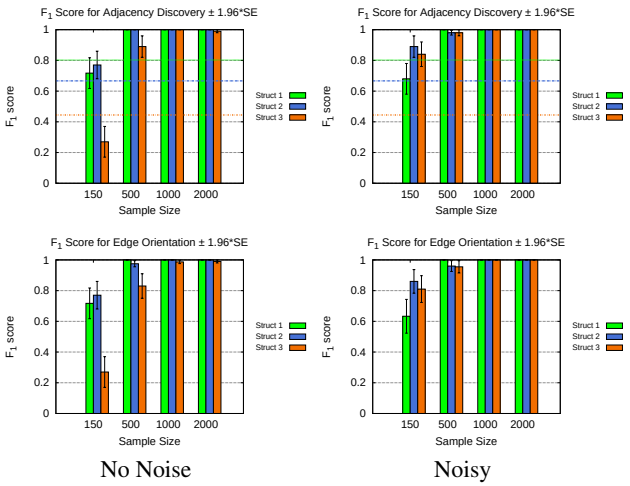


Figure 6: Results of adding systematic noise. Top: Comparison of F_1 scores for adjacency discovery. Horizontal lines are baseline F_1 scores computed for fully connected Bayesian networks. Bottom: Comparison of F_1 scores for edge orientation.

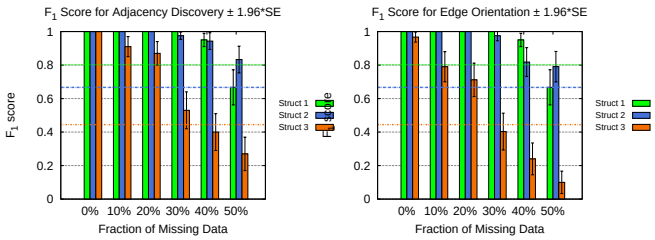


Figure 7: Results of learning with missing data. Left: Comparison of F_1 scores for adjacency discovery. Horizontal lines are baseline F_1 scores computed for fully connected Bayesian networks. Right: Comparison of F_1 scores for edge orientation.

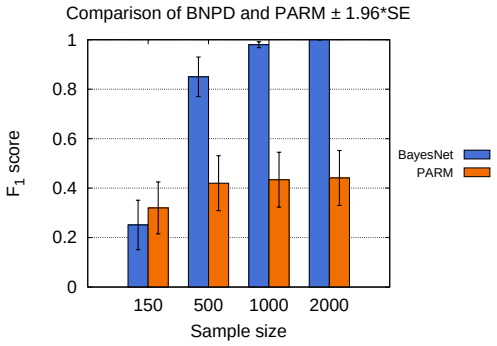


Figure 8: Comparison of BNPD and PARM for discovering prerequisite relationships.

by cyber-charter schools (we call it `virtual`). Both of our datasets systems use the same curriculum and textbook. The textbook items are classified in chapters, sections, and objectives; but for this paper, we only use the data from the first three chapters. In both systems, we use performance data while students solve homework and test items. We model the test and homework data using assessment models (§ 2.2.1) and learning methods (§ **??**), respectively.

2.2.1 Assessment Experiments

We now evaluate how REMIND discovers a remedial model in the assessment context using test data. For this, § 2.2.1 describes the data filtering and the Q-matrix we use; § 2.2.1 describes the prerequisite structure discovered by REMIND; § 2.2.1 describes a quantitative evaluation of the remediation model.

Q-matrix and preprocessing. We use an item-to-skill mapping (Q-matrix) that assigns each exercise to a skill solely as the book section in which the item appears. This Q-matrix does not convey any information on how or when to offer a remedial intervention to a learner. For this, we investigate the extent on which REMIND can discover a remediation model using student performance data. We process both of our datasets to find a subset of items and students that does not have missing data. This is, the dataset we use in REMIND has students responding to *all* of the items.

After filtering, each skill (book section) has two to four items, for a total of 33 items. The resulting dataset from the `traditional` implementation includes student test results for 5202 students; while the `virtual` implementation has the test results for 467 students. Our synthetic data experiments suggest that a large number of training data improves the learning quality of the prerequisite structure. For this reason, we use the larger `traditional` dataset to build the prerequisite model.

Prerequisite Structure Discovery. We now describe how we use the REMIND algorithm to learn a remedial model. For simplicity we use binary variables to encode performance data (i.e, correct or incorrect) and skill variables (i.e., mastery or not mastery). This simplification is not necessary, as REMIND is able to use discrete variables with arbitrary number of states. We use an implementation of Structural EM available online called LibB[4]. We experiment with

[4] `http://compbio.cs.huji.ac.il/LibB/programs.html`

two conditions:

- We use domain knowledge to constrain the prerequisite structure. Since our skills correspond to the book sections, we use the table of contents as expert knowledge. We constraints skills so that a skill a can be prerequisite of a skill b, iff a appears before b in the table of contents of the textbook.
- Alternatively, we also experiment using a fully data-driven approach in which REMIND learns the prerequisite structure without any constraints.

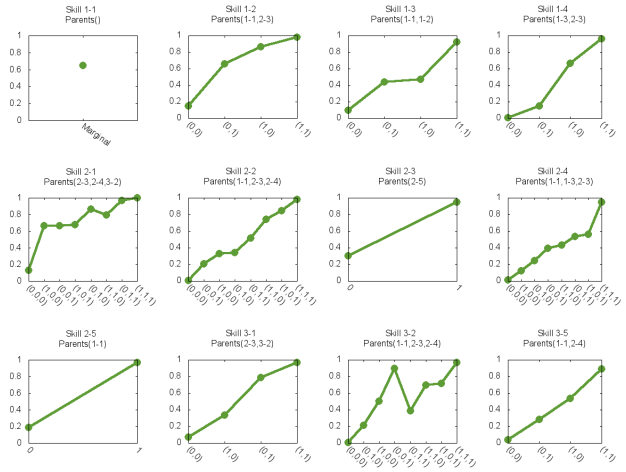


Figure 10: Visualization of the estimated conditional probability distributions for prerequisite structure model in Figure 9 (a). Each plot depicts the conditional probability of mastering a skill given the mastery status of its direct prerequisites. Parents(·) specifies the direct prerequisites of the skill. In X-axis, 0 means the corresponding prerequisite is not mastered and 1 otherwise.

Figure 9 illustrates Bayesian networks generated with the REMIND algorithm. Figure 9a represents a disconnected model generated by simply converting the input Q-matrix into a Bayesian network; this corresponds to the output of the initialization step of Algorithm 1. Figures 9b and 9c represent the prerequisite structures discovered by using domain knowledge or not. In the fully data-driven model, we draw with red the edges that contradict our text book ordering heuristic. Our observation is that the structures learned are not random, sections (skills) in the same chapter tend to form clusters, even in the structure learned without the ordering constraint. We should mention that the BIC score[5] of the structure learned without the ordering constraint is slightly better than the BIC score of that learned with the ordering constraint, indicating that the first model fits the data better than the second one does.

Our approach also outputs the conditional probabilities associated with each skill and its direct prerequisites. Figure 10 plots the conditional distribution table for each skill. Each sub-figure plots the conditional probability of student achieving the mastery of a skill against the mastery status of the skill's direct prerequisites. More specifically, X-axis specifies all possible mastery statuses of a skill's direct prerequisite, Y-axis is the probability of student achieving the mastery of the skill given the corresponding mastery status of its direct prerequisites. We can see a trend in all plots that the probability of student mastering a skill increases monotonically when the student has acquired more prerequisites of the skill. This is consistent with our intuition that mastering a skill's prerequisites will help student master the skill. We also observed a small contradiction in the case of skill 3-2, which exemplifies the limitation of fully data-driven methods.

[5]In our experiment, we use the Bayesian information criterion (BIC) score to measure the fitness of a Bayesian network to the data. The BIC score for a Bayesian network model is composed of a likelihood term and a term to penalize the model complexity.

Quantitative evaluation. We now evaluate REMIND using data that has already been collected (*post-hoc* analysis). We evaluate how well the discovered Bayesian networks can predict student performance on a on a test item given performance on other items. In particular, we compute the posterior probability of a student's response to an item I_i given his performance on all other items $\mathbf{I}_{-i} = \mathbf{I} \setminus \{I_i\}$, by marginalizing:

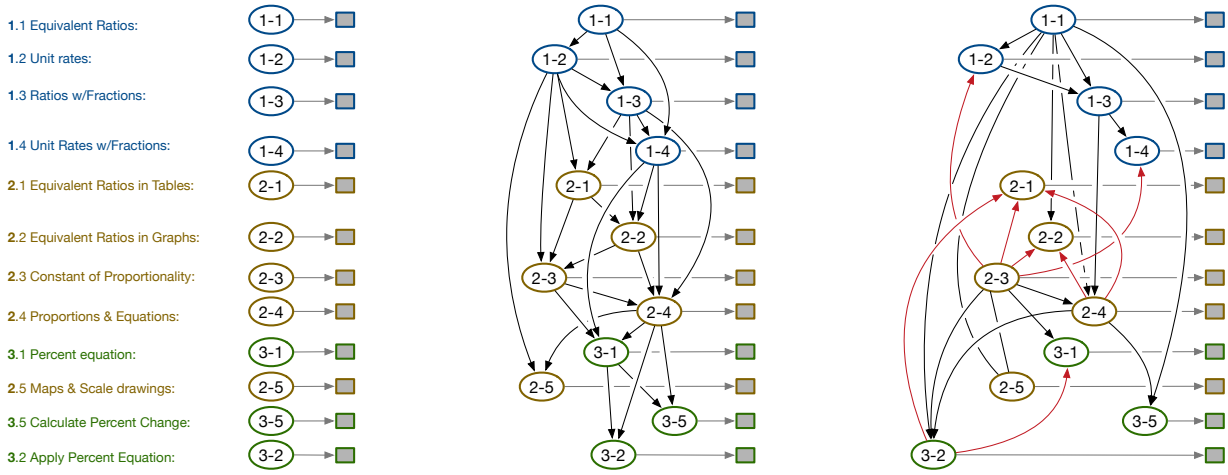
$$P(I_i = 1 | \mathbf{I}_{-i} = \mathbf{i}_{-i}) = \sum_{\mathbf{S}} P(I_i, \mathbf{S} | \mathbf{I}_{-i} = \mathbf{i}_{-i}), \qquad (1)$$

This can be computed efficiently using the Junction tree algorithm [11]. We then do binary classification based on the posterior probability.

We compare REMIND with four baseline predictors:

- A *majority* classifier which always classifies an instance to the majority class. For example, if majority of the students get an item wrong, other students would likely get it wrong.
- A Bayesian network model in which the skill variables are *disconnected*. This corresponds to using the Q-matrix Bayesian network of Figure 9a. This model assumes that the skill variables are marginally independent of each other.
- A Bayesian network model in which the skill variables are connected in a *chain* structure, i.e., 1-1→1-2→1-3→ … This assumes that a section (skill) only depends on the previous section. In other words, a first-order Markov chain dependency structure.
- A *fully connected* Bayesian network where skill variables are fully connected with each other. This model assumes no conditional independence between skill variables and can encode any joint distribution over the skill variables. However, it has exponential number of free parameters and thus can easily overfit the data.

The parameters of these baseline Bayesian network predictors are estimated from the `traditional` data set. We first did cross-validation experiments to evaluate these classifiers using the `traditional` data. Figure 11a evaluates the model predictions using the *Area Under the Curve* (AUC) of the Receiver Operating Characteristic (ROC) curve metric. The error bars show the 95% confidence intervals calculated from the 10-fold cross-validation. The best performing models are REMIND with ordering constraint with an AUC of 0.804 ± 0.007 and REMIND with no ordering constraint with an AUC of 0.807 ± 0.006. These two REMIND models outperform the other four models. The *chain* model performs the best among the four baseline predictors with an AUC of 0.796 ± 0.006. A paired t-test reveals that both of the REMIND models significantly outperform the *chain* model (the p-values are 0.0064 and 0.003). However,



(a) Q-matrix represented as a Bayes network (b) Network constrained by domain knowledge (c) Network with no constraints

Figure 9: Prerequisite structures constructed by Structural EM. Nodes have been augmented with color information to emphasize the chapter association. Reversible edges are depicted as undirected. Squares represent the exercise items.

the two REMIND prerequisite models are not statistically different from each other. We note that the *fully connected* model was outperformed by the two prerequisite models and the *chain* model, suggesting overfitting.

We now investigate the extent of which a REMIND model can be generalized to a different implementation of the curriculum. For this, we use the models constructed from students using the `traditional` curriculum to make prediction on the `virtual` curriculum. The comparison of AUCs is illustrated in Figure 11b. The error bars show the 95% confidence intervals calculated with an implementation of the Logit method[6], which corrects for the non-independence of the points of the ROC curve. Again, the two REMIND prerequisite models clearly outperform other four models. The AUCs of both REMIND models are 0.784 ± 0.010. Curiously, in this experiment the difference between REMIND and the chain baseline becomes more prominent.

3. RELATION TO PRIOR WORK

We believe we are the first ones to propose a pipeline of learning the prerequisite dependencies from data and use it for student modeling. Our work builds on prior work that discovers prerequisite from data [5, 18, 2, 14, 3, 13]. However, these approaches do not attempt to validate the prerequisite discovered using real student performance data. Our approach differs from prior work in several ways. The approaches discover the relationship of items without using latent variables. This is, the prerequisites do not use skill mappings, and only find dependencies between items [5, 18, 13]. For the approaches that can account for latent variables [2, 3], they focus on estimating the pairwise prerequisite relationships. By contrast, we try to optimize the full structure of the model.

A contribution of our work is introducing the Structural EM algorithm to the educational community. This approach has many advantages over prior work in that it does not require tuning of many parameters. For example, prior work [2] assumes domain parameters called *guess probability* and *slip probability* are provided for each pair of item and skill. Similarly, more recent approaches [3] require manually specified thresholds to determine the existence of a prerequisite relationship. The determination of these thresholds requires experts' intervention. By contrast, the only required input of our algorithm is the observed student performance.

A limitation of our work is that we do not address in this paper is comparing Structural EM with these approaches. Future work may address a comprehensive comparison of different prerequisite structure methods.

4. CONCLUSION

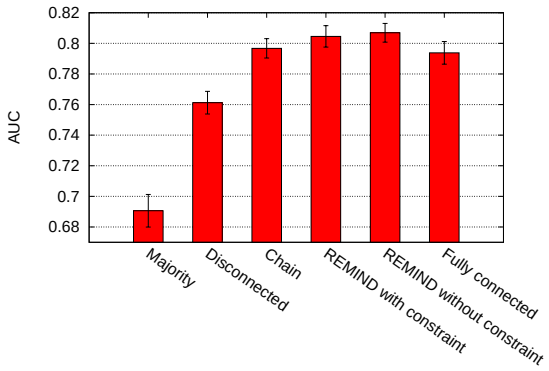
Although in some educational paradigms [10] students are not supposed to move to subsequent lessons until they have mastered all of the prerequisites, these is not always attainable in practice. We propose and evaluate the REMIND pipeline, a simple but effective novel algorithm that detects when a student needs remediation.

A limitation of our study is that we only evaluated REMIND using simulations and posthoc analyses. Future work may evaluate the REMIND algorithm in a randomized control trial. Further, we evaluated our models using traditional statistical metrics, but recent work suggest that tailored evaluation metrics for tutoring system may be superior [8]. Thus, another piece of future work is to evaluate REMIND using these evaluation metrics.

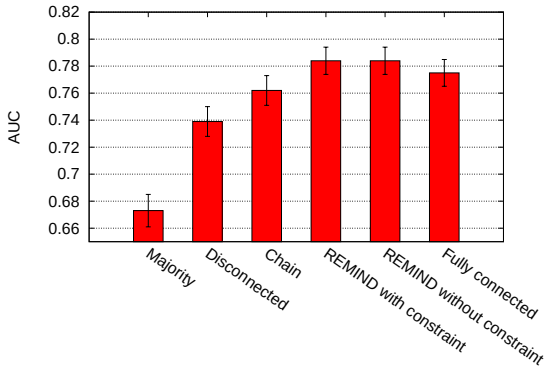
The main contributions of our work are: a novel data-driven algorithm that allows domain knowledge for designing remediation triggers; a novel methodology to evaluate prerequisite graphs using student data; and suggesting the Structural EM algorithm for educational applications.

The advantage of REMIND are both qualitative and quantitative. Qualitatively, REMIND allows us to understand the organization of the skills in the curriculum as it builds a prerequisite network of the skills in a Q-matrix. When used in a learning model, REMIND builds a hypothesis of how practice affects the knowledge of the student. Sometimes the practice may affect the skill directly, but sometimes it may affect a prerequisite. Future work may validate these hypothesis in a controlled experiment. Additionally, our

[6] `http://www.subcortex.net/research/code/area_under_roc_curve`



(a) `traditional` AUC results. The results are from 10 fold cross-validation.



(b) `virtual` AUC results. Models are trained using `traditional` data set.

Figure 11: Comparison of AUC of six models to predict student performance on exercise items

quantitative results suggest that REMIND can be used to improved student modeling. Overall, we believe that REMIND is promising technology to detect when a student needs help.

5. REFERENCES

[1] Russell G Almond, Robert J Mislevy, Linda Steinberg, Duanli Yan, and David Williamson. 2015. *Bayesian networks in educational assessment*. Springer.

[2] Emma Brunskill. 2010. Estimating prerequisite structure from noisy data. In *Educational Data Mining 2011*.

[3] Yang Chen, Pierre-Henri Wuillemin, and Jean-Marc Labat. 2015. Discovering Prerequisite Structure of Skills through Probabilistic Association Rules Mining. In *Proceedings of the 8th International Conference on Educational Data Mining*. 117–124.

[4] Gregory F Cooper and Edward Herskovits. 1992. A Bayesian method for the induction of probabilistic networks from data. *Machine learning* 9, 4 (1992), 309–347.

[5] Michel C Desmarais, Peyman Meshkinfam, and Michel Gagnon. 2006. Learned student models with item to item knowledge structures. *User Modeling and User-Adapted Interaction* 16, 5 (2006), 403–434.

[6] Nir Friedman. 1997. Learning belief networks in the presence of missing values and hidden variables. In *ICML*, Vol. 97. 125–133.

[7] Nir Friedman. 1998. The Bayesian structural EM algorithm. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*. 129–138.

[8] José P. González-Brenes and Yun Huang. 2015. Your model is predictive— but is it useful? Theoretical and Empirical Considerations of a New Paradigm for Adaptive Tutoring Evaluation. In *Proceedings of the 8th International Conference on Educational Data Mining*.

[9] David Heckerman, Christopher Meek, and Gregory Cooper. 1997. *A Bayesian approach to causal discovery*. Technical Report. MSR-TR-97-05, Microsoft Research.

[10] Kenneth R Koedinger, Albert T Corbett, and Charles Perfetti. 2010. The knowledge-learning-instruction (KLI) framework: Toward bridging the science-practice chasm to enhance robust student learning. *Cognitive Science* (2010).

[11] Daphne Koller and Nir Friedman. 2009. *Probabilistic graphical models: principles and techniques*. MIT press.

[12] Judea Pearl. 2000. *Causality: models, reasoning and inference*. Vol. 29. Cambridge Univ Press.

[13] Chris Piech, Jonathan Spencer, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas Guibas, and Jascha Sohl-Dickstein. 2015. Deep Knowledge Tracing. *arXiv preprint arXiv:1506.05908* (2015).

[14] Richard Scheines, Elizabeth Silver, and Ilya Goldin. 2014. Discovering prerequisite relationships among knowledge components. In *Educational Data Mining 2014*.

[15] Peter Spirtes, Clark Glymour, and Richard Scheines. 2001. *Causation, prediction, and search*. MIT Press.

[16] Kikumi K Tatsuoka. 1983. Rule space: An approach for dealing with misconceptions based on item response theory. *Journal of educational measurement* 20, 4 (1983), 345–354.

[17] Thomas Verma and Judea Pearl. 1990. Equivalence and synthesis of causal models. In *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*. 255–270.

[18] Annalies Vuong, Tristan Nixon, and Brendon Towle. 2010. A method for finding prerequisites within a curriculum. In *Educational Data Mining 2011*.