Learning and Evaluating Prerequisite Graphs 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

Knowing the prerequisite structure of skills is crucial for designing curriculum, assessing mastery and for student modeling. These prerequisite structures are often hand-engineered by subject matter expert in a costly and time-consuming process. In this paper, we introduce *Bayesian Network based Prerequisite Discovery* (BNPD), a novel data-driven pipeline for inferring the prerequisite structure of skill from student performance on test items. By modeling the prerequisite relations as a Bayesian network, the pipeline estimates the causal structure and the probabilistic dependence among the skills via a two-stage learning process. In the first stage, the Structural Expectation Maximization (Structural EM) algorithm is used to select a class of Bayesian networks based on distribution fitting of student data; in the second stage, a single Bayesian network structure is determined by enforcing a constraint on the estimated conditional probability tables. We validate the proposed pipeline using simulations and by post-hoc analysis of student data. We show the discovered prerequisite structures can improve the student model in predicting student performance.

Keywords

Prerequisite discovery, Bayesian network, student modeling

1. INTRODUCTION

Students usually acquire skills in a meaningful sequence which starts from relatively simple concepts and gradually approaches more complex ones. 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 paper, 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.

Prerequisite structures of skills are often hand-engineered by subject matter experts in a costly and time-consuming process. Moreover, the prerequisite structures specified by the experts are seldom tested and might be unreliable in the sense that experts may hold "blind spots". Nowadays, large volume of educational data has been cumulated through the online tutoring systems. Thus, learning prerequisite structures from educational data has drawn substantial interest from both education and data mining communities [6, 18, 2, 14, 4, 13]. However, inferring the prerequisite relationships between skills is still challenging since a student's knowledge of a skill is a latent variable in the data. In this paper, we introduce *Bayesian Network based Prerequisite Discovery* (BNPD), a novel pipeline for discovering prerequisite structure of skills from data. BNPD models the prerequisite structure of the skills with a statistical model called Bayesian network. It then learns the Bayesian network through a two-stage process.

2. PREVIOUS WORK

Many works have investigated the discovery of prerequisite structures within domain models from data. Many of these approaches discover the relationship of items without using latent variables. That is, the prerequisites do not use skill mappings, and only find dependencies between items [6, 18, 13].

To account for hidden skills, a mapping of items to skills (often called a Q-matrix) is used. In this setting, students are tested on items-questions, problems, parts of questions- each of which requires one or multiple background skills. To estimate the relationship between two background skills, Brunskill proposed a method to compute and compare the log likelihoods between the prerequisite model and the flat model (two skills are independent) to determine which model better fits the data [2]. Chen et al. [4] used DINA (Deterministic Input Noisy AND) model to infer the knowledge states of students from their response data on test items and then used probabilistic association rules mining to determine the prerequisite relationships between two skills. However, these methods are only capable of estimating the pairwise relationships, instead of optimizing the full structure of the skills.

Bayesian network is a promising tool to model skill topologies as demonstrated in [10]. Bayesian network allows the modeling of full structure of skills and can encode conditional independence relationships between the skill. This enables more accurate modeling of the probabilistic dependence between skills. By modeling prerequisite relations as a Bayesian network, Scheines et al. converted the problem of prerequisite structure discovery to a Bayesian network structural learning problem, and proposed an algorithm to find Bayesian network structures that fit the data [14]. However, the output from their algorithm is a class of Bayesian networks, which represent different prerequisite structures. They did not offer any

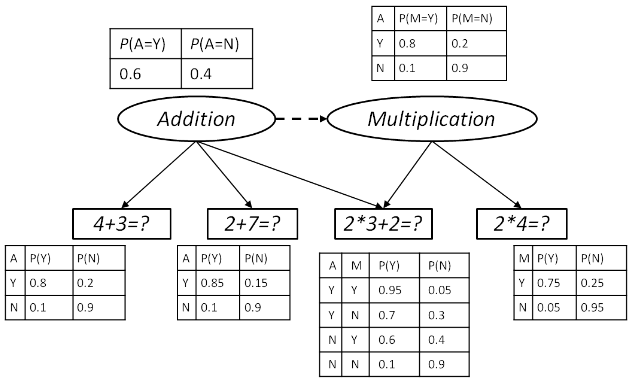


Figure 1: A hypothetical Bayesian network learned with Algorithm 1. Solid edges are given by the item to skill mapping, dashed edges between skill variables are to be discovered from data. The conditional probability tables are to be learned.

solution to discriminate between these Bayesian networks.

In this paper, we propose the *Bayesian Network based Prerequisite Discovery* (BNPD) pipeline to infer the prerequisite structures of skills from data. The pipeline, as named, also uses the promising Bayesian network model to represent the prerequisite relationships. The pipeline first uses a popular Bayesian network structure learning algorithm, called Structural EM to select a class of Bayesian network models based on the distribution fitting of the data. It then uses constraint on the conditional probability tables to select one Bayesian network, which gives a unique prerequisite structure. We demonstrate the robustness of proposed approach on noisy data that contains unaccounted hidden variables or missing values. Finally, we show the discovered prerequisite structures are useful for student modeling.

3. THE BNPD PIPELINE

BNPD 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 BNPD pipeline. Suppose we collect data from n students, answering p items. Then, the input of BNPD is a matrix $\mathbf{D}$ with $n \times p$ dimensions, an item to skill mapping.Each entry in $\mathbf{D}$ encodes the performance of a student (see Table 1 for an example).

Table 1: Example data matrix to use with BNPD. 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
		...		

BNPD 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. We now describe the steps of BNPD in detail.

Algorithm 1 The BNPD 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 $\mathbf{S}$).

Initialization
$G_0 \leftarrow$ Initialize$(\mathbf{S}, Q)$
$i \leftarrow 0$

Structural EM
do
  E-step:
    $\theta_i^* \leftarrow$ ParametricEM$(G_i, \mathbf{D})$
    $\mathbf{D}_i^* \leftarrow$ Inference$(G_i, \theta_i^*, \mathbf{D})$
  M-step:
    $\langle G_{i+1}, \theta_{i+1} \rangle \leftarrow$ BNLearning$(G_i, \mathbf{D}_i^*)$
    $i \leftarrow i+1$
while Stop criteria is not met

Orient edges
$RE \leftarrow FindReversibleEdges(G_i)$
$EC \leftarrow EnumEquivalentDAGs(G_i)$
$DE \leftarrow \{\}$
for every reversible edge $S_i - S_j$ in RE **do**
  $ratio \leftarrow \frac{P(S_i=1|S_j=1) \cdot P(S_j=0|S_i=0)}{P(S_j=1|S_i=1) \cdot P(S_i=0|S_j=0)}$ [1]
  if $ratio > 1$ **then**
    $ratio^* = ratio$
    $DE \leftarrow DE \cup S_i \rightarrow S_j$
  else
    $ratio^* = \frac{1}{ratio}$
    $DE \leftarrow DE \cup S_i \leftarrow S_j$
  end if
end for
$sort(DE)$ by $ratio^*$ in descending order
while DE is not empty **do**
  $e \leftarrow dequeue(DE)$
  $\forall G \in EC$, remove G from EC if $e \notin G$
end while
return EC

3.1 Initial Bayesian Network

BNPD 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

[1] $P(S_i = a|S_j = b)$ can be computed using any Bayesian network inference algorithm such as Junction tree algorithm[11].

answering items (e.g, correct or incorrect). We first create an initial Bayesian network that complies to the skill-to-item Q-matrix on the prerequisite structure That is, we create an arc to each item from each of its required skills and 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.

3.2 Structural EM

A common solution to learning a Bayesian network from data is the score-and-search approach [5, 9]. This approach uses a scoring function[2] 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. However, the number of possible Bayesian network structures is super-exponential with respect to the number of nodes. The Structural Expectation Maximization algorithm [7, 8] is an efficient alternative.

Structural EM is an iterative algorithm that inputs a matrix **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.[3] 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.

BNPD'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 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

[2]A commonly used scoring function is Bayesian information criterion (BIC), which is composed of a likelihood term and a term to penalize the model complexity

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

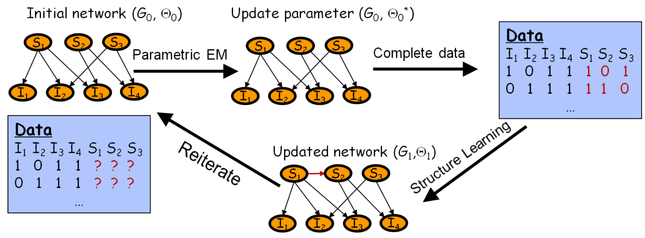


Figure 2: An illustration of the Structure EM algorithm to discover the structure of the latent variables.

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 prerequisite of content that appears later in the book.

Another advantage of Structural EM is that it can be applied when there are missing data in the student performance matrix **D**. That is, some students do not answer all the items thus some responses are missing. This "missing values" problem is very common in real-world situations. Structural EM can be applied to this type of data since it was originally developed to solve the "missing values" problem [7]. The general idea is, in the *E-step*, the algorithm also computes the expected values for missing data points, in addition for latent variables.

3.3 Discriminate Between Equivalent BNs

Structural EM selects Bayesian network model based on how well it explains the distribution of the data. Bayesian network theory states that some Bayesian networks are statistically equivalent in representing the data. Thus, the output from Structure EM is an equivalence class that may contain many Bayesian network structures. These equivalent Bayesian networks have the same skeleton and the same v-structures[4]. For instance, Figure 3 gives an example of two simple Bayesian networks that are not distinguishable by Structural EM algorithm. They share the skeleton but differ in the orientation of the edge. They apparently represent two different prerequisite structures.

3.3.1 Use Domain Knowledge

To determine a unique structure, we shall determine the orientation of each reversible edge. For this purpose, we can use the following domain knowledge.

Knowledge 1. If S_1 is a prerequisite of S_2, i.e., $S_1 \to S_2$, then $P(S_1 = 1|S_2 = 1) = 1$ and $P(S_2 = 0|S_1 = 0) = 1$.

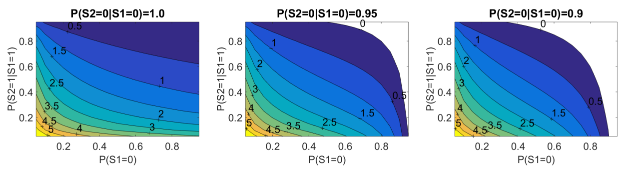
Knowledge 1 says if a skill S_1 is the prerequisite of a skill S_2, a student must master skill S_1 before he masters S_2 and it is impossible for him to master S_2 if he has not mastered S_1. In other words, S_1 is not a prerequisite of S_2 if at least one of the conditions does not hold. This puts a constraint on the joint distribution encoded by the Bayesian network to be learned. We can check for any reversible edge $S_1 - S_2$ in the Bayesian network, either $P(S_1 = 1|S_2 = 1) = 1$ and $P(S_2 = 0|S_1 = 0) = 1$, or $P(S_2 = 1|S_1 = 1) = 1$ and $P(S_1 = 0|S_2 = 0) = 1$. If the former holds, $S_1 \to S_2$; otherwise, $S_1 \leftarrow S_2$.

[4] 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].

However, in real situations, it is possible for a student to master the post-requisite even he does not know the prerequisite. Further, there is always noise in the data. Thus, the two conditional probabilities $P(S_1=1|S_2=1)$ and $P(S_2=0|S_1=0)$ are not exactly but close to 1. Thus, we use the following empirical rule: if $P(S_1=1|S_2=1)\cdot P(S_2=0|S_1=0) \geq P(S_2=1|S_1=1)\cdot P(S_1=0|S_2=0)$, we determine $S_1 \rightarrow S_2$; otherwise, we determine $S_1 \leftarrow S_2$. Note that these conditional probabilities can be computed easily from the Bayesian network model output from Structural EM. The intuition behind this is that the two probabilities $P(S_1=1|S_2=1)$ and $P(S_2=0|S_1=0)$ are certificates of the prerequisite structure $S_1 \rightarrow S_2$. The larger of these two probabilities, the more likely the relationship $S_1 \rightarrow S_2$ holds. Since here we are concerned with which direction the edge goes, we simply compare the products of two probabilities and select the direction that is more probable.

(a) (b)

Figure 3: Two equivalent Bayesian networks representing different prerequisite structures.



(a) (b) (c)

Figure 4: Contour plot of $log(ratio)$ for various values of $P(S_2=0|S_1=0)$.

3.3.2 Theoretical Justification

The empirical rule proposed in section 3.3.1 needs theoretical justification. We consider the simple Bayesian network in Figure 3a, which has only one reversible edge $S_1 - S_2$. This model has three free conditional probability parameters: $P(S_1=0)=p$, $P(S_2=0|S_1=0)=q$, $P(S_2=1|S_1=1)=r$. If we observe data generated from this model, can we use the rule to determine $S_1 \rightarrow S_2$? Equivalently, we use rule $ratio = \frac{P(S_1=1|S_2=1)\cdot P(S_2=0|S_1=0)}{P(S_2=1|S_1=1)\cdot P(S_1=0|S_2=0)} \geq 1$. Using Bayes rule and rules of probability, the rule becomes

$$ratio = \frac{(1-p)(qp+1-r-p+pr)}{p[r(1-p)+p(1-q)]} \geq 1, \tag{1}$$

which can be simplified to

$$\frac{(1-p)(1-r)-p(1-q)}{p[r(1-p)+p(1-q)]} \geq 0. \tag{2}$$

The rule holds only if $(1-p)(1-r)-p(1-q) \geq 0$. Thus, we can use the rule to determine $S_1 \rightarrow S_2$ if $(1-p)(1-r)-p(1-q) \geq 0$. Since $ratio$ depends on p, q and r, we study how $ratio$ changes with these parameters. Figure 4 shows the contour plots of $log(ratio)$ against $P(S_1=0)$ and $P(S_2=1|S_1=1)$ for three different values of $P(S_2=0|S_1=0)$. The white region in each contour plot is the region where the rule is not applicable. That is, in this region, $ratio < 1$ even $S_1 \rightarrow S_2$ is the true model. As it is showed, if $P(S_2=0|S_1=0)=q=1$, the rule is always applicable no matter what values $P(S_1=0)$ and $P(S_2=1|S_1=1)$ are. With $P(S_2=0|S_1=0)$ decreasing, the white region becomes larger and the rule becomes more inapplicable. $P(S_2=0|S_1=0)$ can be interpreted as the strength of the prerequisite relationship. When the prerequisite relationship is weaker, the rule is less applicable and using the rule to determine the prerequisite relationship is less accurate.

3.3.3 Orient All Reversible Edges

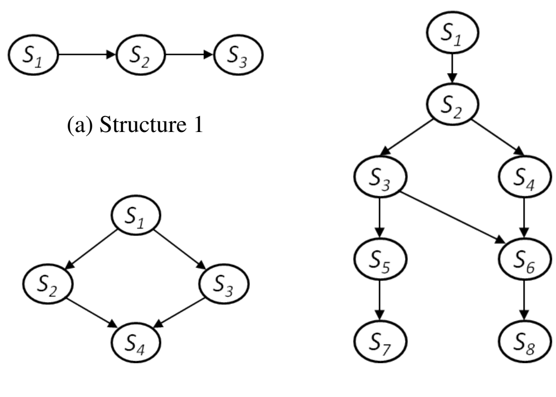
Using the empirical rule, we can orient every reversible edge in the network structure. However, orienting each reversible edge is not independent and may conflict each other. Having oriented one edge would constrain the orientation of other reversible edges because we have to ensure the graph is a DAG and the equivalence property is not violated. For example, in Figure 5a, if we have determined $S_1 \rightarrow S_2$, the edge $S_2 \rightarrow S_3$ is enforced. In this paper, we take an ad-hoc strategy to determine the orientation for all reversible edges. For each reversible edge $S_i - S_j$, we let $ratio^* = ratio$ if $ratio > 1$ and $ratio^* = \frac{1}{ratio}$ otherwise. The larger the $ratio^*$ is, the more confidently when we decide the orientation. We sort the list of reversible edges by $ratio^*$ in descending order. We then orient the edges by this ordering. When one edge is oriented, the constraint is propagated to other reversible edges. In practical implementation, we use a the following strategy: we first enumerate all equivalent Bayesian networks and make them a list of candidates; when an edge is oriented to $S_i \rightarrow S_j$, we remove all contradicting Bayesian networks from the list. Eventually only one Bayesian network structure stands. This procedure is detailed in the *Orient edges* section of Algorithm 1. The $EnumEquivalentDAGs(G_i)$ implements the algorithm of enumerating equivalent DAGs in [3].

4. EVALUATION

In § 4.1, we evaluate BNPD with simulated data to assess the quality of the discovered prerequisite structures. Then, in § 4.2 we use data collected from real students. In all our experiment, we use the Bayesian information criterion (BIC) as the scoring function in Structural EM .

4.1 Simulated Data

Synthetic data allows us to study how BNPD compares to ground truth. For this, we engineered three prerequisite structures (DAGs), shown in Figure 5. Here, each figure represents different causal relations between the simulated latent skill variables.



(a) Structure 1

(b) Structure 2

(c) Structure 3

Figure 5: Three different DAGs between latent skill variables. Item nodes are omitted.

For clarity, Figure 5 omits the item nodes; but each skill node is parent of six item variables and each item variable has 1-3 skill nodes as parents. 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 BNPD 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. 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.[5] For completeness, we list these formulas in tables 2 and 3, 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*	$\frac{2 \cdot TPOR \cdot TDOR}{TPOR + TDOR}$

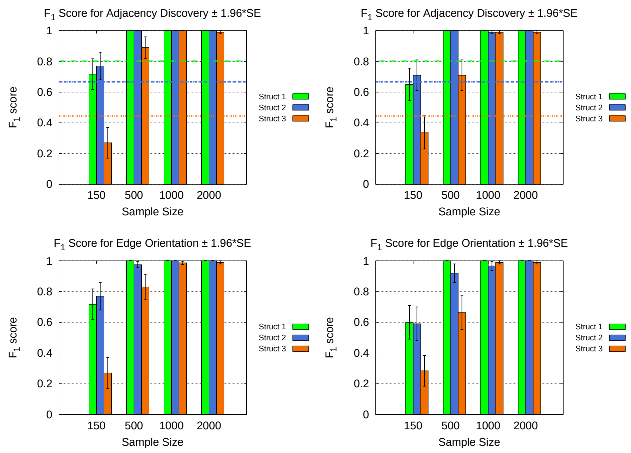
We use these metrics to evaluate the effect of varying the number of observations of the training set (sample size) on the quality of learning the prerequisite structure. We designed experiments to specifically answer the following four questions:

1. How does the type of items affect BNPD's ability to recover the prerequisite structure? We consider the situation where in the model each item requires only one skill and the situation where each item requires multiple skills.
2. How well does BNPD perform when there is noise in the data? We focus on studying noise due to the presence of unaccounted hidden variables.
3. How well does BNPD perform when the student performance data has missing values?
4. How is BNPD compared with other prerequisite discovery methods? In particular, we compare BNPD to the Probabilistic Association Rules Mining (PARM) method [4].

We now investigate these questions.

4.1.1 *Single-skill vs Multi-skill Items*

We consider two situations where different types of Q-matrix are used. In the first situation, each item node maps to only one skill node. In the second one, each item loads 1-3 skills. Figure 6 compares the F_1 of adjacency discovery and edge orientation result under two types of Q-matrix. 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, when the model contains multiple-skill items, the accuracy is slightly lower than that where all items in the model are single-skilled items, probably because there are more parameters to be estimated in the case of multi-skill items.



Single Skill Multiple Skill

Figure 6: 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.

4.1.2 *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 BNPD 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. For simplicity, all items in the setting are single-skilled items. 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 7 illustrates the procedure of this sensitivity analysis experiment.

[5]We do not compute F_1 orientation score for fully connected DAGs because all edges in a fully connected DAG are reversible.

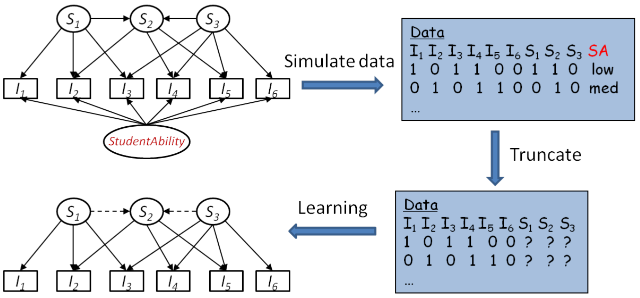


Figure 7: Evaluation of BNPD with noisy data

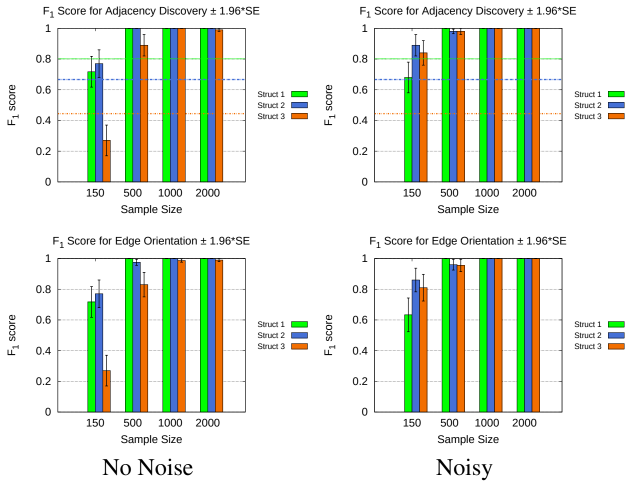


Figure 8: Results of adding systematic noise. Top: Comparison of F_1 scores for adjacency discovery. Horizontal lines are baseline F_1 scores computed for fully connected Bayesian networks. Bottom: Comparison of F_1 scores for edge orientation.

Figure 8 compare the results where noise was introduced or not. Interestingly, the noise does not harm the learning accuracy at all, and actually improves the accuracy. We hypothesize that adding this variable is actually removing noise by separating students by ability, which can help the structure learning.

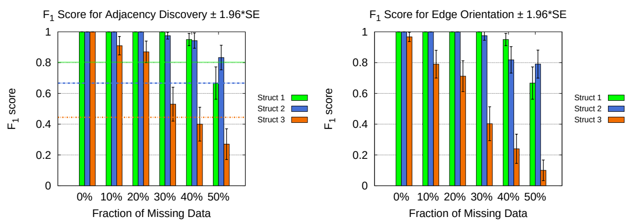


Figure 9: Results of learning with missing data. Left: Comparison of F_1 scores for adjacency discovery. Horizontal lines are baseline F_1 scores computed for fully connected Bayesian networks. Right: Comparison of F_1 scores for edge orientation.

4.1.3 In Presence of Missing Values

Real-world student data often has missing values, i.e., students do not respond to all items. To evaluate how BNPD performs on data with missing values, we generated data sets of size 1000 with varying fraction of randomly missing values (10%, 20%, 30%, 40%, 50%) and used BNPD to recover the structures from these data sets. Again, the models only contains single-skilled items. The results are presented in Figure 9. It is seen that the accuracy decreases when the fraction of missing values increases. However, BNPD is still able to recover the true structures for Structure 1 and 2 even the data contains up to 30% missing values.

4.1.4 Comparison With PARM

Chen and his colleagues proposed to use Probabilistic Association Rules Mining (PARM) for discovering the prerequisite relationships between skills [4]. Since PARM discovers pair-wise prerequisite relationship, instead of constructing the full structure, we only compare BNPD with PARM for discovering the pair-wise relationships. That is, we derived pair-wise prerequisite relationships from the Bayesian network structure and see how the two approaches discover these relationships. Here we simulated data from Structure 3 (Figure 5(c)) (with single-skilled items), which has 21 pair-wise prerequisite relationships. When experimenting with PARM, the following cutoff values were used: $minsup = 0.125$, $minconf = 0.76$, $minprob = 0.9$. These values have been used in the simulation experiments in [4]. The evaluation metric used is $F_1 = \frac{2*TPR*TDR}{TPR+TDR}$, where $TPR = \frac{\text{\# of correct relationships learned}}{\text{\# of relationships in true model}}$ and $TDR = \frac{\text{\# of correct relationships learned}}{\text{\# of relationships in learned model}}$. The result is presented in Figure 10. It shows that BNPD significantly outperforms PARM for sample size $n \geq 500$. The low F_1 score by PARM is caused by low TPR, i.e., PARM failed to discover many prerequisite relationships (data not shown). It can be speculated that selecting a different set of cutoff values for PARM might improve the results. However, determining these thresholds is not trivial and requires experts' intervention.

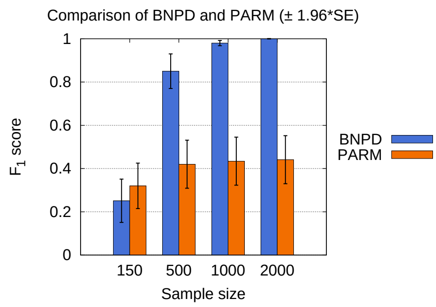


Figure 10: Comparison of BNPD and PARM for discovering prerequisite relationships in Structure 3. The cutoff values used for PARM experiments are: $minsup = 0.125$, $minconf = 0.76$, $minprob = 0.9$.

4.2 Real Student Performance Data

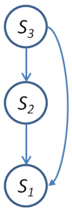
We now evaluate BNPD using two real-world student data sets.

4.2.1 ECPE Data Set

The ECPE (Examination for the Certification of Proficiency in English) data set was collected from a test by the English language Institute of the University of Michigan to examine individual cognitive skills required for understanding English language grammar [16]. It contains a sample of 2922 examinees who is tested by 28 items on 3 skills, i.e., *morphosyntactic rules* (S_1), *cohesive rules* (S_2) and *lexical rules* (S_3). Each item requires either one or two of the three skills. The prerequisite structure output from BNPD are depicted in Figure 11. The estimated conditional probability tables (CPT) are showed correspondingly. The discovered structure says *lexical rules* is a prerequisite of *cohesive rules* and *morphosyntactic rules*, and *cohesive rules* is a prerequisite of *morphosyntactic rules*. This totally agrees with the findings in [16] and by the PARM method in [4]. Further, BNPD also outputs the conditional probabilities associated with each skill and its direct prerequisite. And we clearly see that the probability of student mastering a skill increases when the student has acquired more prerequisites of the skill.

$P(S_3{=}0)$	$P(S_3{=}1)$
0.33	0.67

S_3	$P(S_2{=}0 \mid S_3)$	$P(S_2{=}1 \mid S_3)$
0	0.90	0.10
1	0.20	0.80



S_2	S_3	$P(S_1{=}0 \mid S_2,S_3)$	$P(S_1{=}1 \mid S_2,S_3)$
0	0	0.97	0.03
1	0	0.57	0.43
0	1	0.87	0.13
1	1	0.33	0.67

Figure 11: Result on ECPE data set. S_1: Morphosyntactic rules; S_2: Cohesive rules; S_3: Lexical rules. The estimated conditional probability tables (CPTs) are shown correspondingly.

4.2.2 HED Data Set

We now evaluate BNPD using data collected from a commercial non-adaptive tutoring system. The data set, named `HED`, is from anonymized students interacting with an implementation of a seventh grade math curriculum. The textbook items are classified in chapters, sections, and objectives; but for this paper, we only use the data from Chapter 2 and Chapter 3 of the book. We use performance data while students solve test items. That is, students are tested on the items after they have been taught all relevant skills.

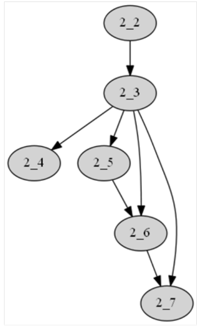
Q-matrix and preprocessing. We use an item-to-skill mapping (Q-matrix) that assigns each exercise to a skill solely as the book section in which the item appears. For each chapter, We process the data set to find a subset of items and students that does not have missing values. This is, the dataset we use in BNPD has students responding to *all* of the items.

After filtering, we obtained two data sets, `HED-chap2` and `HED-chap3` for Chapter 2 and 3 respectively. In `HED-chap2`, six skills (book sections) are included and each skill is tested on three to eight items, for a total of 30 items. In `HED-chap3`, seven skills are included and each skill has three to seven items, for a total of 33 items. `HED-chap2` includes student test results for 1720 students, while the `HED-chap3` has test results for 1245 students.

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 BNPD is able to use discrete variables with arbitrary number of states.

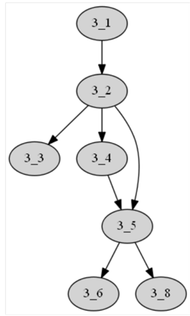
Prerequisite Structure Discovery. The Bayesian networks generated with the BNPD algorithm are illustrated in Figure 12. Our observation is that the topological order of the sections in both structures are fully consistent with the book ordering heuristic, This shows an agreement between our fully data-driven method and human experts. We also ran PARM approach to learn pair-wise prerequisite relationships from these data sets. Given $minsup =$

Skill ID	Skill Name
2_2	Symbols and Sets of Numbers
2_3	Fractions and Mixed Numbers
2_4	Exponents, Order of Operations, Variable Expressions, and Equations
2_5	Adding Real Numbers
2_6	Subtracting Real Numbers
2_7	Multiplying and Dividing Real Numbers



(a) Prerequisite structure learned for `HED-chap2`.

Skill ID	Skill Name
3_1	Simplifying Algebraic Expressions
3_2	The Addition and Multiplication Properties of Equality
3_3	Solving Linear Equations
3_4	An Introduction to Problem Solving
3_5	Formulas and Problem Solving
3_6	Percent and Mixture Problem Solving
3_8	Solving Linear Inequalities



(b) Prerequisite structure learned for `HED-chap3`.

Figure 12: Prerequisite structures constructed by BNPD for HED data sets.

0.1, $minconf = 0.75$ and $minprob = 0.9$, $2\_5 \rightarrow 2\_6$ and $2\_5 \rightarrow 2\_7$ are discovered for `HED-chap2`, $3\_1 \rightarrow 3\_3$ and $3\_2 \rightarrow 3\_3$ are discovered for `HED-chap3`. These relationships are subset of the set of relationships discovered by BNPD.

Predictive Performance of Prerequisite Models. BNPD not only discovers the topological relationships among skills, but also outputs a statistical model in the form of Bayesian network. This statistical model can be used for inference and predictive modeling. For example, given a student's response to one or to a set of items, we can infer his knowledge status of any skill. This is very useful for remedial intervention, i.e., discovering when a student is lacking background knowledge. In this paper, we evaluate these models by evaluating how well the generated Bayesian networks predict student performance on a test item given response 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:

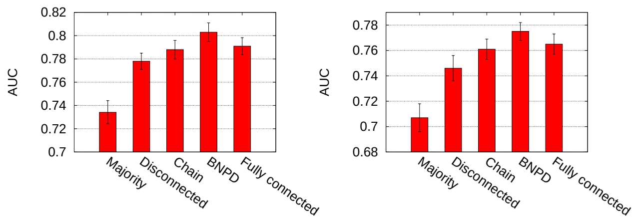
$$P(I_i = 1 | \mathbf{I}_{-i} = \mathbf{i}_{-i}) = \sum_{\mathbf{S}} P(I_i, \mathbf{S} | \mathbf{I}_{-i} = \mathbf{i}_{-i}), \tag{3}$$

This can be computed efficiently using the Junction tree algorithm [11]. We then do binary classification based on the posterior probability. We compare the Bayesian network models generated from BNPD 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 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., 2-2→2-3→2-4→ ... 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.



(a) HED-chap2 AUC results. (b) HED-chap3 AUC results.

Figure 13: Comparison of AUC of five models to predict student performance on test items. The results are from 10 fold cross-validation.

The parameters of these baseline Bayesian network predictors are estimated from the data. The model predictions were evaluated using the *Area Under the Curve* (AUC) of the Receiver Operating Characteristic (ROC) curve metric calculated from 10-fold cross-validation. Results are presented in Figure 13. The error bars show the 95% confidence intervals calculated from the cross-validation. On both HED-chap2 and HED-chap3 data sets, the BNPD models outperform the other four models. The *fully connected* models are the second best performing models. On HED-chap2, BNPD model has an AUC of 0.803 ± 0.008 and the *fully-connected* model has an AUC of 0.791 ± 0.007 (Figure 13a). A paired t-test reveals that the AUCs of two models are statistically different with a p-value of 0.0022. On HED-chap3, BNPD model has an AUC of 0.775 ± 0.007 and the *fully-connected* model has an AUC of 0.765 ± 0.008 (Figure 13b). The AUCs of two models are also statistically different with a p-value of 0.01. The *fully connected* models was outperformed by the prerequisite models, which is much simpler, suggesting overfitting.

5. CONCLUSION AND DISCUSSION

Discovering prerequisite structure of skills from data is challenging since skills are latent variables in the data. In this paper, we proposed BNPD, a novel pipeline for learning the prerequisite dependencies of hidden skill variables from students' response data on test items. We demonstrated our approach using both synthetic and real-world data under different conditions such as data contains missing values and noises. We show the discovered prerequisite models are useful for student modeling and remedial intervention by demonstrating its predictive performance on unseen data.

Studying prerequisite structure discovery in the context of Bayesian network structure learning is not new [14]. However, existing approach does not discriminate between equivalent Bayesian network structures which represent different prerequisite structures. In our approach, we address this problem by using an ad-hoc strategy to refine the prerequisite model with a constraint on conditional distributions tables. This strategy is quite empirical. In future, it would be wiser to combine Structural EM learning with model discrimination. For example, in Structural EM search, we can use a scoring function with an additional term that penalizes the structures that violate the CPT constraint. In such way, learning may be more accurate.

6. 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] Yetian Chen and Jin Tian. 2014. Finding the k-best Equivalence Classes of Bayesian Network Structures for Model Averaging.. In *AAAI*. 2431–2438.

[4] 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.

[5] 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.

[6] 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.

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

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

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

[10] Tanja Käser, Severin Klingler, Alexander Gerhard Schwing, and Markus Gross. 2014. Beyond knowledge tracing: Modeling skill topologies with bayesian networks. In *Intelligent Tutoring Systems*. Springer, 188–198.

[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] Jonathan Templin and Laine Bradshaw. 2014. Hierarchical diagnostic classification models: A family of models for estimating and testing attribute hierarchies. *Psychometrika* 79, 2 (2014), 317–339.

[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*.