Recovering Concept Prerequisite Relations from University Course Dependencies

Chen Liang[†] **Jianbo Ye**[†] **Zhaohui Wu**[‡] **Bart Pursel**[†] **C. Lee Giles**[†]
[†]Pennsylvania State University
[‡]Microsoft Corporation
{cul226, jxy198, bkp10, giles}@ist.psu.edu zhaowu@microsoft.com

Abstract

Prerequisite relations among concepts play an important role in many educational applications such as intelligent tutoring system and curriculum planning. With the increasing amount of educational data available, automatic discovery of concept prerequisite relations has become both an emerging research opportunity and an open challenge. Here, we investigate how to recover concept prerequisite relations from course dependencies and propose an optimization based framework to address the problem. We create the first real dataset for empirically studying this problem, which consists of the listings of computer science courses from 11 U.S. universities and their concept pairs with prerequisite labels. Experiment results on a synthetic dataset and the real course dataset both show that our method outperforms existing baselines.

Introduction

A *prerequisite* is a concept or skill that is necessary to learn before one can proceed to understand more advanced knowledge. Prerequisite relations exist as natural dependencies among concepts in cognitive processes when we learn, organize, apply, and generate knowledge (Laurence and Margolis 1999). Recently, the growth of available educational data has made a variety of emerging educational applications possible. Examples include intelligent tutoring systems (Aleven and Koedinger 2002), curriculum planning (Agrawal, Golshan, and Papalexakis 2015), automatic sequencing of learning materials (Changuel, Labroche, and Bouchon-Meunier 2015), etc. In these applications, obtaining prerequisite relations plays a crucial role.

While it can benefit both learners and instructional designers, discovering prerequisite relations among concepts is usually done manually by domain experts (Bergan and Jeska 1980). However, it is inefficient and expensive — and does not scale with large concept sets. A possible solution for scaling is to develop or integrate approaches that automatically infer such prerequisites from the increasing amount of digital educational data. Available sources include knowledge bases, student assessment data, text books, course materials, etc.

Developing data-driven methods for automatically discovering prerequisite relations is challenging. Earlier work in educational data mining has been devoted to analyzing student assessment data which records the performance of students on different items (e.g. units, sections, etc.) (Vuong, Nixon, and Towle 2011). Existing approaches aim to discover prerequisite relations among certain performance variables such as handcrafted knowledge components and skills. Since those discovered relations are highly structured and are often subject to a specific assessment pool, they can not be generalized for other educational purposes or applicable for processing a large concept set. To address these issues, Wikipedia data is exploited to find prerequisite relations among universally shared concepts (Talukdar and Cohen 2012; Liang et al. 2015), using both the Wikipedia article contents and their linkage structures. For the purpose of curriculum planning, Yang et al. (2015) proposed a supervised framework to infer course prerequisites by constructing a latent concept graph to support the prediction.

Here, we focus on the problem of recovering concept prerequisite relations from course dependencies (denoted as **CPR-Recover** for short). We utilize a similar data setting as that of (Yang et al. 2015) but instead focus on recovering an accurate and universally shared concept graph from the observed course dependencies rather than extrapolating the course prerequisites to unseen course pairs. We have information and dependencies of courses collected from different universities. As shown in Figure 1, courses #1-4 are from the curriculum of University A. Based on their descriptions, course dependencies (e.g. Course #2 depends on Course #1) are used to recover concept prerequisite relations in a shared concept graph. To address the CPR-Recover problem, we propose an unsupervised optimization-based method based on the following two assumptions:

1. **Causality**: The dependency among courses is caused by sufficient evidence provided by prerequisite relations among concepts representing the courses.
2. **Sparsity**: The prerequisites in a concept graph will be sparse, which means the number of prerequisite relations is much smaller than the total number of concept pairs.

Our method is designed to recover the part of concept prerequisite relations that causes course dependencies. Take the example in Figure 1, the prerequisite relations (Matrix ⇒ Gaussian elimination, Matrix ⇒ QR decomposition) are evidences for supporting the dependency between Course

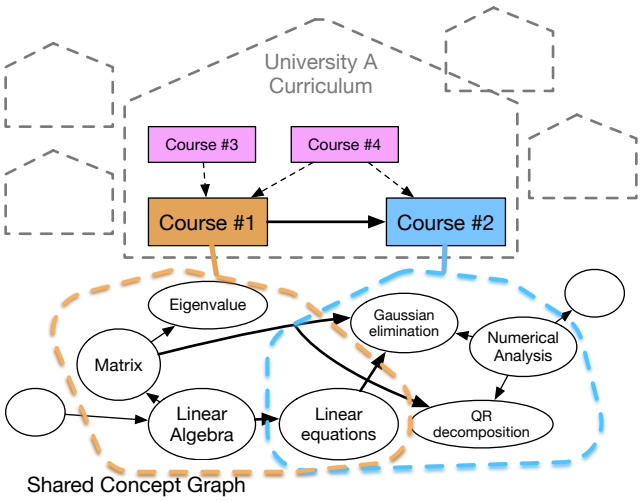


Figure 1: Recovering course prerequisite relations from course dependencies.

#1 and #2, thus can be recoverable from course data. Our method has been tested on both a synthetic dataset and a real computer science course dataset (including course names, descriptions, dependencies, etc.) collected from 11 U.S. universities. In order to make the discovered concept prerequisite relations explicit and interpretable, we represent these courses in terms of Wikipedia concepts. For the evaluation of concept prerequisites, we recruit a group of graduate students to assign the ground truth labels on a filtered subset of concept pairs. Experimental results on the two datasets both demonstrate the superior performance of our approach.

Our main contributions include:

1. A novel method to learn concept-level prerequisite relations from course dependencies that outperforms existing baselines on both a synthetic dataset and a real university course dataset.
2. The first real dataset for studying and evaluating the CPR-Recover problem, which consists of 654 unique computer science courses from 11 universities and 3544 concept pairs with their prerequisite labels.

Problem Setup

For convenience, we will use the following notations:

- $M = \{1, 2, ...m\}$ is the set of concepts where m is the number of concepts.
- $N = \{1, 2, ..., n\}$ is the set of all course IDs where n is the number of unique courses;
- $\mathbf{x}_i \in \mathbb{R}^m$ is the vector representation of the course i in concept space;
- $\{\mathbf{x}_i\}_{i \in N}$ is the set of courses;
- $i \mapsto j$ represents course i is a prerequisite of course j;
- $\Omega = \{i \mapsto j\}_{i,j \in N}$ is the set of prerequisite relations among courses;
- $\mathbf{A} = (a_{s,t})$ is a m-by-m matrix representing prerequisite relations among concepts, where $a_{s,t}$ is the weight quantifying how concept t depends on concept s.

CPR-Recover Problem Definition. Given a set of courses $\{\mathbf{x}_i\}_{i \in N}$ with a concept representation per course, and a set of observed course dependencies Ω, our goal is to recover the matrix $\mathbf{A}$ which quantifies the strength of prerequisite relations among concepts.

Our Approach

Course Representation

Since course descriptions are usually in the form of unstructured text, we first calculate a document representation for each course. In order to get an explicit and interpretable concept space, we choose to represent the text using Wikipedia concepts[1]. We represent each course as a bag-of-concepts model. Instead of using all words in the text, we first extract all Wikipedia concepts that are in the course description using Wikipedia Miner (Milne and Witten 2013). After concept extraction, the course vector x_i is calculated using term frequency-inverse document frequency (tf-idf).

CPR-Recover Formulation

Our approach to solve the CPR-Recover problem makes two assumptions: *causality assumption* and *sparsity assumption*. The former assumes that the prerequisite relation between two courses is caused by sufficient evidence provided by prerequisite relations among concepts which the two courses consist of. This serves as a bridge for making inferences between course dependencies and concept prerequisite relations. The latter assumption is that the prerequisite relations in a concept graph are sparse: the number of prerequisite relations is much smaller than the total number of concept pairs. Since concepts are usually linked with their prerequisites, we validate our sparsity assumption by estimating the sparsity of a concept prerequisite graph by sampling from the Wikipedia link graph. A depth-first search under the category "Computer science" returns a domain-specific sub-graph with about 10^5 nodes and 2.8×10^6 edges. The graph density is $\sim 2.8 \times 10^{-4}$, showing the sampled graph is quite sparse.

Based on the two assumptions, we propose to solve the CPR-Recover problem by the following formulation:

$$\begin{aligned} \min_{\mathbf{A},\xi} \quad & \|\text{vec}(\mathbf{A})\|_1 + \lambda \cdot \sum_{i \mapsto j \in \Omega} \xi^2_{i \mapsto j} \\ \text{s.t.} \quad & \mathbf{x}_i^T \left[\mathbf{C}_{\mathbf{i} \mapsto \mathbf{j}} \odot \mathbf{A}\right] \mathbf{x}_j \geq \theta - \xi_{i \mapsto j}, \quad \forall i \mapsto j \in \Omega \\ & a_{s,t} + a_{t,s} = 0, \quad \forall s, t \in M \\ & -1 \leq a_{s,t} \leq 1, \quad \forall s, t \in M \\ & a_{s,t} = 0, \quad \forall (s,t) \notin \mathcal{K} \end{aligned} \tag{1}$$

where θ and λ are constant positive parameters; $\xi_{i \mapsto j}$ is a slack variable for the course pair (i, j); $\odot$ is element-wise product; $\mathbf{C}_{\mathbf{i} \mapsto \mathbf{j}} = (c_{s,t})$ is a design matrix where $c_{s,t} \in \{0, 1\}$ and $c_{s,t} = 0$ if $\mathbf{x}_i[t] > 0$ or $\mathbf{x}_j[s] > 0$, otherwise

[1]Each concept corresponds to a unique English Wikipedia article.

$c_{s,t} = 1$; $\mathcal{K}$ is the set of candidate concept prerequisite relations obtained from external prior knowledge.

Constraints result from the causality and sparsity assumptions, and also external knowledge. The first constraint is based on the causality assumption. Every course dependency $i \mapsto j$ is caused by the interaction among $\mathbf{x}_i$, $\mathbf{A}$, and $\mathbf{x}_j$. $\mathbf{C_{i \mapsto j}}$ is incorporated to remove the contribution from the common concepts between $\mathbf{x}_i$ and $\mathbf{x}_j$ to the course dependency $i \mapsto j$. If concepts occur in both course i and j, we assume they are not the cause of the course dependency. The second constraint specifies that $\mathbf{A}$ is a skew-symmetric matrix, which means if concept c_a is a prerequisite of c_b then c_b is not a prerequisite of c_a. The third constraint bounds the strength of prerequisite relation in $[-1, 1]$. The last constraint allows the method to incorporate external knowledge. Specifically, $\mathcal{K}$ here consists of all Wikipedia concept pairs $\{(c_a, c_b)\}$ where there is at least one hyperlink between c_a and c_b. Following Talukdar et al. (2012), we assume there is no prerequisite relation between two concepts which are not linked. Additional external knowledge can also be inserted. For example, if some of the concept prerequisite relations is already known from other resources such as manually-built concept graphs, this knowledge can also be incorporated into our method as constraints.

For the objective function, the first term is the regularization term and the second term is the empirical loss. In particular, the L1-norm on parameter $\mathbf{A}$ exploits the sparsity of computed prerequisite relations, and L2-norm of the slack variables is the loss in this setting. Two distinct variants of our approach come from mutating the choices of these two norms. By replacing the first term by L2-norm instead, one could show that a model proposed in (Yang et al. 2015) is a special case of this variant. And by replacing the first term by L2-norm and the second term by L1-norm, the resulting variant problem is the soft-margin SVM (Cortes and Vapnik 1995). For solving problems of moderate scale within minutes, it suffices to use standard quadratic programming solvers. Scalable optimization techniques, such as ADMM or dual coordinate descent, are available to handle large-scale problems. Experimental results below validate that this choice of norms which is coherent with our model assumption and achieves empirically good performance.

Experiments

For evaluation we conduct experiments on two datasets created by us: a synthetic dataset and a real course dataset. To the best of our knowledge, there is no existing dataset that contains both the course dependency data and prerequisite labels for the underlying concept graph. All experiments are done on a Red Hat Enterprise Linux server with 24 Intel Xeon processors @ 2.67GHz and 32GB of RAM. Mosek[2] is used to solve the optimization problem.

Synthetic Dataset

To simulate the CPR-Recover problem, we generate a concept prerequisite graph, pairs of course with dependencies, and documents representing the courses.

[2]Available online at https://www.mosek.com/

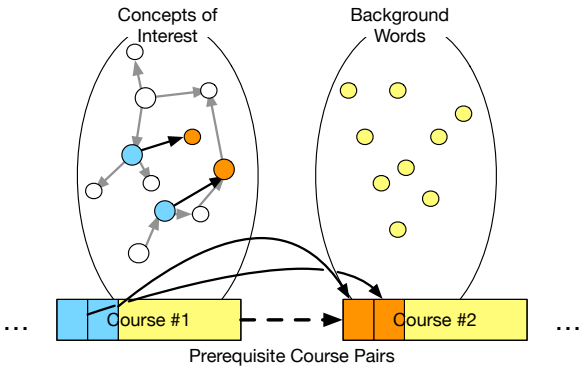


Figure 2: Process of synthetic data generation. After a concept prerequisite graph is created, prerequisite course pairs are generated by blending prerequisite concept pairs with background words.

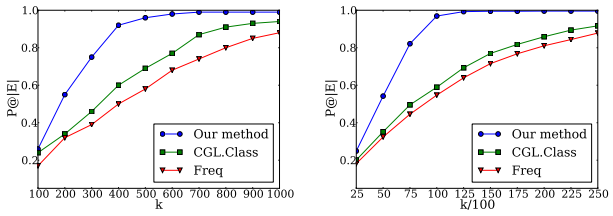
G	m	$\lvert E\rvert$	$\lvert V\rvert$	p	l	l_c
G_S	100	100	100	0.01	10	2
G_L	500	2463	500			

Table 1: Statistics of two synthetic datasets.

Process of Data Generation The process of data generation illustrated in Figure 2 consists of the following two steps:

1. (*Concept prerequisite graph generation*) We follow the Erdős-Rényi model to generate the underlying concept prerequisite graph, i.e., generating a random graph given the number of concepts (nodes) m and the probability p for creating a prerequisite relation (edge) between two concepts. Note every edge is set to be directed from a vertex to another with a larger concept ID. This step results in a directed graph G with m nodes and $|E|$ edges.
2. (*Prerequisite course pairs generation*) Next we generate k pairs of courses with prerequisite relations. Two assumptions are made: (i) The document representing the course consists of both concepts and background words. Prerequisite relations that only exist among concepts and background words are considered as noise. (ii) The prerequisite relation between two documents is consistent with the prerequisite relation between concepts in the two documents. Specifically, given the document length l, the number of concepts in a document l_c, the vocabulary of background words V, we generate a pair of documents (d_i, d_j) representing $i \mapsto j$ with the following three steps:
   (a) Randomly sample l_c edges from the concept prerequisite graph G generated from previous step.
   (b) For every edge $c_s \mapsto c_t$ from the sampled l_c edges, add c_s to d_i, c_t to d_j.
   (c) Randomly sample $l - l_c$ background words from V and add them to d_i. Conduct a similar random sampling to add background words to d_j.

For our experiment, two concept prerequisite graphs, G_S and G_L, with different sizes are generated to test perfor-



(a) Results for G_S. (b) Results for G_L.

Figure 3: Results on two synthetic datasets.

mance at different scales. A detailed description of the two experiment settings is shown in Table 1. The vocabulary size of background words is set equal to the number of nodes in G. The edge creation probability p is set to be 0.01 to enforce the sparsity on the concept graph. The document length l is set to 10 and each document consists of 2 concepts. We now explore the relation between the method performance and the number of course prerequisite pairs k. Specifically consider the following questions: will the concept prerequisite matrix $\mathbf{A}$ be better recovered if we obtain more pairs of course prerequisites? How will our method perform compared to other baselines?

Baselines and Evaluation Metrics – Synthetic Data For comparison we use the two following baselines, one a previous method (**CGL.Class** (Yang et al. 2015)) and the other a naive method — **Freq** which for the concept pair (c_s, c_t) calculates $a_{s,t}$ as the number of times the pair "co-occurs" in course prerequisite pairs. For $i \mapsto j$, (c_s, c_t) "co-occurs" if $\mathbf{x}_i[s] > 0$ and $\mathbf{x}_j[t] > 0$.

Precision at K (*P@K*) is calculated for the evaluation of $\mathbf{A}$. We first sort the elements of $\mathbf{A}$ in a descending order and get an ordered list $l_{\mathbf{A}} = \{a_{s,t}\}$. $P@K$ is calculated as $\frac{\sum_{i=1}^{K} rel(i)}{K}$, where $rel(\cdot)$ is binary indicator function; $rel(i) = 1$ if the concept pair (c_s, c_t) corresponding to the i-th element in $l_{\mathbf{A}}$ belongs to E. In our experiment, we set $K = |E|$ because ideally the top $|E|$ elements of $\mathbf{A}$ should correspond to all $|E|$ edges in the generated concept prerequisite graph.

Results – Synthetic Data Experimental results on the two synthetic datasets are shown in Figure 3. For each method $P@|E|$ is calculated with a given number of course prerequisite pairs k. For the experiment setting G_S, k is chosen from $[100, 200, ..., 1000]$. And for G_L, k is chosen from $[2500, 5000, ..., 25000]$. We can see that results for G_S and G_L are consistent and show that: (i) As k increases, the performance of different methods keeps improving; (ii) Our method outperforms both *CGL.Class* and *Freq* for every choice of k. A closer look at the performances of the three methods shows that the difference is decreasing when k becomes small enough ($\approx |E|$) or large enough ($\approx 10|E|$). While for the former case there are not enough observations to make correct inferences, for the latter case all three methods are able to recover most of the concept prerequisites from the sufficient observed course prerequisites. In addition, we can see that when $k \approx 6|E|$ our method can recover almost all edges in G while the other two methods perform significantly worse. We explain this by noting that our method is designed to exploit the sparsity assumption for the concept prerequisite graph G. In contrast, *CGL.Class* and *Freq* do not.

University Course Dataset

In addition to the synthetic dataset, we also create a real course dataset based on data collected from 11 U.S universities, all with a focus on computer science or computer science like departments (CS). We developed a Web scraper to extract the course information from the online course catalogs of these universities. Besides the basic information such as course ID, name, and description, the course catalogs also provide course prerequisite information. For example, "CS 311 Data Structures and Algorithms" is a prerequisite for "CS 422 Data mining". Our focus on courses in the computer science is two-fold. We have domain expertise in that area and focusing on only one domain will make the ground truth acquisition easier and more realistic. After collecting the course descriptions, we apply Wikipedia Miner (Milne and Witten 2013) to extract Wikipedia concepts.

In total from the 11 universities there were 654 unique CS courses (both undergrad and graduate level), 639 pairs of courses with prerequisite relations, and 569 Wikipedia concepts. And the average number of concepts per course was 4.73. The dataset is available upon request.

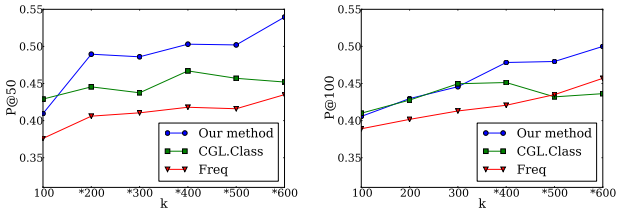
Data Labeling To accurately label for a given concept pair (c_s, c_t) whether c_t is an actual prerequisite of c_s requires domain knowledge. As such for labeling we recruited 13 graduate students with CS backgrounds. Instead of labeling all pairs of concepts in the concept space, they only needed to label the set of candidate concept pairs $P = \{(c_s, c_t) | \exists i \mapsto j \in \Omega, \text{ s.t. } \mathbf{x}_i[s] > 0 \text{ and } \mathbf{x}_j[t] > 0\}$, i.e., only the concept pairs that "co-occur" in course prerequisite pairs. Note that for a concept pair $(c_s, c_t) \notin P$, $a_{s,t}$ would be unchanged and stay at zero during the optimization process.

In the course dataset, from 861 course prerequisites we get $|P| = 3544$. For each candidate pair (c_s, c_t), each student annotator decides whether c_t is a prerequisite of c_s and gives a binary label. The labeling task is assigned to the 13 annotators in a way that each concept pair can get labels from three annotators. The majority vote for the three labels is treated as the ground truth. Based on our setting of the labeling process, Fleiss' kappa κ (Fleiss 1971) is used to assess the reliability of agreement between annotators. Note that $\kappa = 1$ if annotators are in complete agreement and $\kappa \leq 0$ if the observed agreement among the raters is no more than what would be expected by chance. For the labels we collected, $\kappa = 0.42$, which shows a level of moderate agreement. With no existing labeled concept prerequisite dataset available, such data, though not perfect, is necessary for evaluation. Finally, from the 3544 candidate concept pairs we get 1008 pairs of concepts with prerequisite relations, denoted as P_{true}.

Baselines and Evaluation Metrics – University Course Data Besides *CGL.Class* and *Freq*, our method is also compared with the following two baselines. **RefD** (Liang et

Methods	$P@50$	$P@100$	$AP@50$	$AP@100$
Our method	**0.54**	0.50	**0.60**	**0.57**
CGL.Class	0.46	0.42	0.56	0.51
Freq	0.44	0.46	0.37	0.41
RefD	0.52	**0.55**	0.42	0.49
Random	0.28			

Table 2: Results on the course dataset.



(a) Results for $P@50$. (b) Results for $P@100$.

Figure 4: Results using different numbers of pairs (k) of course dependencies. $*$ indicates that with a given k our method significantly outperforms other two methods ($p < 0.05$).

al. 2015): A link-based metric for measuring prerequisite relations among Wikipedia concepts. For each $(c_s, c_t) \in P$, set $a_{s,t} = RefD(c_s, c_t)$. Note that instead of utilizing course dependencies this method requires the knowledge of the whole Wikipedia graph. We compare with this method in order to explore the performance difference between our method and methods based on external knowledge bases. **Random**: For each $(c_s, c_t) \in P$, randomly choose $a_{s,t}$ from $\{0, 1\}$ with equal probability. Each time the probability of discovering a true prerequisite pair is equal to the ratio of prerequisite pairs P_{true} to all candidate pairs P, which is $|P_{true}|/|P|$.

Given the size of concept graph and the limited number of observed course prerequisites in the dataset, the method is expected to recover only a part of all prerequisite concept pairs. Thus we focus on the Top-K precision performance. Specifically, for the evaluation of $\mathbf{A}$, we use precision at K ($P@K$) and average precision at K ($AP@K$). Given the ranked list of scores for each candidate pair, $l_P = \{a_{s,t}\}_{(c_s,c_t)\in P}$, $P@K$ and $AP@K$ are calculated by $P@K = \frac{\sum_{i=1}^{K} rel(i)}{K}$ and $AP@K = \frac{\sum_{i=1}^{K} P@i \cdot rel(i)}{K}$, where $rel(\cdot)$ is binary indicator function; $rel(i) = 1$ if the concept pair (c_s, c_t) corresponding to the i-th element in l_P belongs to P_{true}. In our experiments, we compare different methods with K equal to 50 and 100.

Results – University Course Data Note that there are two parameters θ and λ for our method. We perform a grid search on $\theta \in \{4, 8, 12, 16, 20, 24\}$ and $\lambda \in \{2^{-13}, 2^{-12}, ..., 2^{-6}, 2^{-5}\}$ to find the best parameters. We find that the parameter combination $(\theta = 8, \lambda = 2^{-8})$ yields the best $AP@100$. Table 2 lists the experiment results of different methods, from which we have the following findings: Our method outperforms all baselines in terms of $P@50$ and $AP@K$. As for $P@100$, our method performs better than *CGL.Class*, *Freq*, and *Random*, only worse than *RefD*. Such results are encouraging because, as previously mentioned, *RefD* utilizes the link structure of the entire Wikipedia, which contains much more information than the collected course prerequisite dataset. In other words, if we only compare methods which are strictly based on the CPR-Recover problem setting, our method is the best one w.r.t. both $P@K$ and $AP@K$.

	(c_s, c_t)
TPP	(System programming, Computer programming)
	(Computer graphics, Algorithm)
	(Machine learning , Probability)
	(Machine learning, Mathematical optimization)
	(Parallel algorithm, Parallel computing)
	(Computer animation, Computer graphics)
	(Programming language, Computer science)
	(Operating system, Computer programming)
	(Computer network, Computer)
	(Software engineering, Data structure)
FPP	(Mathematical optimization, Computer science)
	(Analysis of algorithms, Data structure)
	(Algorithm, Dynamic programming)
	(Database, Computer software)
	(Computer vision, Graph theory)

Table 3: Examples of prerequisite concept pairs we recovered, which are categorized as true prerequisite pairs (TPP) and false prerequisite pairs (FPP).

Our method is also evaluated in a similar setting to the one used for the synthetic dataset, where our method is compared with *CGL.Class* and *Freq* using different number of course dependencies (k). Since in total only $|\Omega| = 639$ course dependencies are available, k is chosen from $[100, 200, ..., 600]$. For each k, we randomly sample k dependencies from Ω and compare different methods. We repeat such process 40 times and calculate the average performance. Results using different k are shown in Figure 4. While different methods perform similarly when k is small, the advantage of our method to other baselines becomes statistically significant ($p < 0.05$) given sufficient k, specifically when $k \geq 200$ for $P@50$ and $k \geq 400$ for $P@100$.

To analyze the human performance for identifying concept prerequisites, we evaluate each of the 13 student annotators based on the ground truth P_{true} (i.e., the consensus). The precision and F-score for identifying concept prerequisites by student annotators are 0.75 ± 0.13 ($mean \pm SD$) and 0.75 ± 0.08. Because they have previously acquired background knowledge, it is not surprising that the precision is much higher than the top-K precision of all methods in Table 2. From experimental results on synthetic dataset, we have noticed that our method benefits more from the increasing number of course dependencies than baselines do. If a sufficient number of course dependencies is given, the difference between the performances of our method and that of humans is expected to become smaller.

Case Study We further investigate our method by studying examples of prerequisite concept pairs recovered. Ta-

ble 3 lists examples of both true prerequisite pairs (TPP) and false prerequisite pairs (FPP), based on the ground truth labels collected. While we can see from the TPP that we can recover some of the concept prerequisites based on course dependencies, the FPP illustrate errors. Looking closely at the FPP, we hypothesize the errors are due to: (i) The performance of concept extraction is not that good. For example, the Wikipedia Miner extracts *Mathematical optimization* from the course description "...basic program analysis and optimization..." rather than find the correct concept *Program optimization*. *Program optimization* requires *Computer science*, but *Mathematical optimization* does not. (ii) A Wikipedia concept can have different levels of interpretation, which will affect the choice for data labeling. For example, both *Database* and *DBMS* correspond to the same Wikipedia concept *Database*. *Computer software* is a prerequisite for *DBMS* but not for the general *Database*. (iii) For our dataset, concept pairs such as (*Algorithm*, *Dynamic programming*) and (*Analysis of algorithms*, *Data structure*) co-occur many times in course prerequisites. Such pairs are more likely to be recovered by data-driven methods including ours and the compared baselines.

Related Work

To the best of our knowledge, the problem of recovering concept prerequisite relations from course dependencies has not been systematically studied. However, our work is closely related to the following research areas.

Design of data-driven methods for automatically discovering prerequisite relations has been explored in multiple works. Established methods in educational data mining have been developed based on the automatic analysis of the assessment data acquired by students' performance (Vuong, Nixon, and Towle 2011). In addition, Liu et al. (2011) proposed a classification method for mining learning dependencies between knowledge units in text books. As the largest open knowledge base, Wikipedia has also been studied to find prerequisite relations among concepts (Agrawal, Golshan, and Papalexakis 2015; Liang et al. 2015; Talukdar and Cohen 2012), where both Wikipedia article contents and their link structures are utilized. Using Wikipedia, Wang et al. (2016) designed a joint framework for key concept extraction and prerequisite identification to extract concept maps from textbooks. For automatic curriculum planing, Yang et al. (2015) proposed a supervised framework to infer course prerequisites by constructing a latent concept graph to support prediction. Such a data setting is also utilized in our work. In comparison, their empirical efforts focused more on extrapolating the observed course prerequisites to unseen pairs, while we focus on recovering a universally shared concept graph. In their paper, the latent concept graph based on their approach — not yet formally evaluated as remarked in (Yang et al. 2015) — is also empirically evaluated with our method in the two experiments. Gordon et al. (2016) proposed a information-theoretic metric to capture concept dependencies in a scientific corpus. Their method relies on topic modeling techniques and requires human annotaions of latent topics to make the result interpretable.

Conclusion

We proposed an effective data-driven method for recovering concept prerequisite relations from university course dependencies, the CPR-Recover problem. Our method was evaluated on a synthetic dataset and real course datasets derived from computer science or related course listings at 11 US universities and significatly outperformed existing baselines. To our knowledge, this is the first real dataset for the CPR-Recover problem.

References

Agrawal, R.; Golshan, B.; and Papalexakis, E. 2015. Data-driven synthesis of study plans. Technical Report TR-2015-003, Data Insights Laboratories.

Aleven, V. A., and Koedinger, K. R. 2002. An effective metacognitive strategy: Learning by doing and explaining with a computer-based cognitive tutor. *Cognitive science* 26(2):147–179.

Bergan, J. R., and Jeska, P. 1980. An examination of prerequisite relations, positive transfer among learning tasks, and variations in instruction for a seriation hierarchy. *Contemporary Educational Psychology* 5(3):203–215.

Changuel, S.; Labroche, N.; and Bouchon-Meunier, B. 2015. Resources sequencing using automatic prerequisite–outcome annotation. *TIST* 6(1):6.

Cortes, C., and Vapnik, V. 1995. Support-vector networks. *Machine learning* 20(3):273–297.

Fleiss, J. L. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin* 76(5):378.

Gordon, J.; Zhu, L.; Galstyan, A.; Natarajan, P.; and Burns, G. 2016. Modeling concept dependencies in a scientific corpus. In *Proc. ACL*.

Laurence, S., and Margolis, E. 1999. Concepts and cognitive science. *Concepts: core readings* 3–81.

Liang, C.; Wu, Z.; Huang, W.; and Giles, C. L. 2015. Measuring prerequisite relations among concepts. In *Proc. EMNLP*, 1668–1674. ACL.

Liu, J.; Jiang, L.; Wu, Z.; Zheng, Q.; and Qian, Y. 2011. Mining learning-dependency between knowledge units from text. *The VLDB Journal* 20(3):335–345.

Milne, D., and Witten, I. H. 2013. An open-source toolkit for mining wikipedia. *Artificial Intelligence* 194:222–239.

Talukdar, P. P., and Cohen, W. W. 2012. Crowdsourced comprehension: predicting prerequisite structure in wikipedia. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, 307–315. ACL.

Vuong, A.; Nixon, T.; and Towle, B. 2011. A method for finding prerequisites within a curriculum. In *Proc. EDM*, 211–216.

Wang, S.; Ororbia, A.; Wu, Z.; Williams, K.; Liang, C.; Pursel, B.; and Giles, C. L. 2016. Using prerequisites to extract concept maps fromtextbooks. In *Proc. CIKM*, 317–326. ACM.

Yang, Y.; Liu, H.; Carbonell, J.; and Ma, W. 2015. Concept graph learning from educational data. In *Proc. WSDM*, 159–168. ACM.