Modeling and Leveraging Prerequisite Context in Recommendation

HENGCHANG HU, National University of Singapore, Singapore
LIANGMING PAN, National University of Singapore, Singapore
YIDING RAN, National University of Singapore, Singapore
MIN-YEN KAN, National University of Singapore, Singapore

Prerequisites can play a crucial role in users' decision-making yet recommendation systems have not fully utilized such contextual background knowledge. Traditional recommendation systems (RS) mostly enrich user–item interactions where the context consists of static user profiles and item descriptions, ignoring the contextual logic and constraints that underlie them. For example, a RS may recommend an item on the condition that the user has interacted with another item as its prerequisite. Modeling prerequisite context from conceptual side information can overcome this weakness. We propose Prerequisite Driven Recommendation (PDR), a generic context-aware framework where prerequisite context is explicitly modeled to facilitate recommendation. We first design a Prerequisite Knowledge Linking (PKL) algorithm, to curate datasets facilitating PDR research. Employing it, we build a 75k+ high-quality prerequisite concept dataset which spans three domain. We then contribute PDRS, a neural instantiation of PDR. By jointly optimizing both the prerequisite learning and recommendation tasks through multi-layer perceptrons, we find PDRS consistently outperforms baseline models in all three domains, by an average margin of 7.41%. Importantly, PDRS performs especially well in cold-start scenarios with improvements of up to 17.65%.

CCS Concepts: • **Information systems → Recommender systems**; *Content analysis and feature selection*; *Collaborative filtering*.

Additional Key Words and Phrases: Recommendation Systems, Prerequisites, Context-aware Recommendation

ACM Reference Format:
Hengchang Hu, Liangming Pan, Yiding Ran, and Min-Yen Kan. 2022. Modeling and Leveraging Prerequisite Context in Recommendation. In *Proceedings of Workshop on Context-Aware Recommender Systems (CARS@RecSys'22).* ACM, New York, NY, USA, 13 pages.

1 INTRODUCTION

Prerequisites are defined as the necessary contexts that enable downstream activity or state in human cognitive processes [16]. In certain domains — especially education [1, 24, 34] — such requisites are an important consideration that constrains item selection. Context-aware recommendation systems have integrated the use of collaborative filtering with auxiliary metadata about users' current background or state, such as time sand location [22, 32]. However, the role of prerequisite context (represented in the form of concepts [16] describing items) has been neglected in recommendation, where such dependent information is crucial for modeling users' interests. Take the educational domain example in Figure 1: The item's key concept *Probability Classifier* and user's prior knowledge are both prerequisite contexts for recommendation. By leveraging the implicit prerequisite relationships between them (represented in the form of a prerequisite graph), we can achieve a comprehensive and explainable recommender that connects the user prerequisite context with item prerequisite context. In our example, *Probability Classifier* is beyond the knowledge of what the user already knows

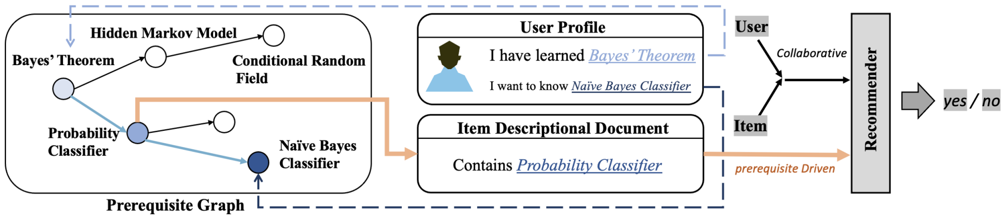


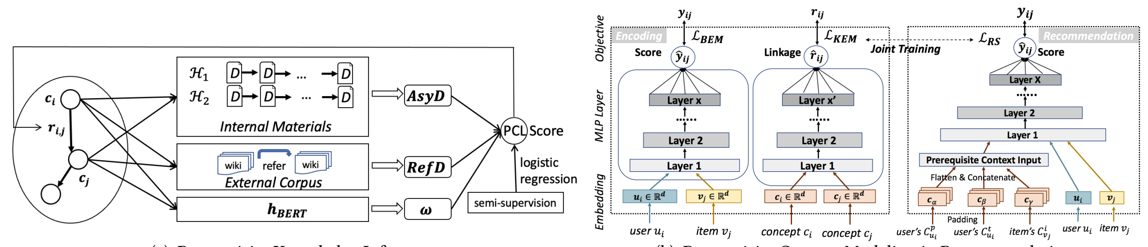
Fig. 1. An illustration of prerequisite driven recommendation: in which a recommender (right) incorporates prerequisite knowledge (left), distilled from both user and item prerequisite context. Dashed edges link users to the concepts they have mastered as prior concepts and the desired concepts to acquire as target knowledge.

(*Bayes' Theorem*) but on the path towards to the user's desired target knowledge (*Naive Bayes Classifier*), thus deserving a higher recommendation probability. Comparatively, items containing the concept of *Conditional Random Field* — although related to *Bayes' Theorem* — would be poor choices since the user lacks the prerequisite prior knowledge of *Hidden Markov Model*. Given the importance of user and item prerequisite context, we explore the possibility of modeling and leveraging them in recommendation.

Though previous work [12] focused on the sequential modeling between items, there is virtually no work investigating the next-step decision in light of users' conceptually-mastered knowledge. We fill this gap by capturing the user and item prerequisite contexts through the compilation of a prerequisite graph, and treating the user's knowledge as static context. Specifically, both the set of *prior concepts* that a user has already mastered, and the set of *target concepts* the user aims to acquire, directly influence the sequence of items to recommend. Based on this, we make a key observation that prerequisite driven recommendation requires two subtasks: (1) prerequisite modeling at the concept level, and (2) user modeling that identifies the prior and target concepts at the user–item level. More concretely, concept prerequisite modeling is the identification of prerequisite links among concepts; *cf* Fig. 1, prerequisite edges link the introductory concept *Probability Classifier* to the more advanced *Naïve Bayes Classifier*. Prior and target concept identification is thus the the process of inferring the state of knowledge for each user, with respect to the inventory of concepts in the prerequisite graph.

To demonstrate the effectiveness of leveraging prerequisite context for recommendation, we propose a Prerequisite-Driven Recommendation System (PDRS) embodying this formalism. The key challenge here is how to accurately acquire user- and item- forms of prerequisite context for use in recommendation. However, there is an important shortcoming. Explicit prerequisite knowledge is often sparse, requiring laborious effort to compile. It is often also brittle, as items and their relationships with underlying contextual concepts can evolve over time. Assuming manually-labeled prerequisites [33, 34] is often unrealistic due to the heavy cost of human annotation. An automatic means of inferring prerequisites is called for. We address this in two parts by contributing a) an automatic key knowledge concept extractor from item descriptive text, and b) a prerequisite relation constructor for concept pairs by inferring prerequisite weights from both internal and external domain features. Our encoding components finds a suitable representation of a user in terms of prior and target knowledge, leveraging pretrained language models. We then integrate such prerequisite context into the recommendation process by joint training of both the recommendation and prerequisite knowledge learning tasks.

We evaluate our PDRS system on three datasets representing different domains. We find that PDRS achieves performance gains in recommendation not only in domains where prerequisites exist as hard constraints — such as (educational) course recommendation — but also in domains where prerequisites are soft, personal preferences, as in movie and book recommendation. Importantly, we also show that such model makes an especially strong impact in sparse data cold-start scenarios, a pervasive problem in RS.



(a) Prerequisite Knowledge Inference (b) Prerequisite Context Modeling in Recommendation

Fig. 2. Our instantiated neural PDRS framework. It consists components of a) prerequisite constructor, b [left]) twin encoders pretrain embedding of users, items (BEM), and prerequisites (KEM), and b [right]) a fine-tuned neural recommendation system.

2 PROBLEM FORMULATION

Definition 1. Prerequisite Context. The context of each user u can be seen as a personal concept repository consisting of a set of mastered prior concepts and a set of target concepts to be acquired, represented as sets $\{C_u^p, C_u^t\} \subseteq C$, respectively. Items then contain these concepts: each item v's context are denoted as C_v^i. Importantly, items manifest concepts in a latent, implicit manner, such that the item concept inventory must be inferred.

Definition 2. Prerequisite Graph. We represent prerequisite context as a graph $\mathcal{G}$, having concepts C as nodes, and prerequisite relations $\mathcal{R}$ among them as edges, where $r \in \mathcal{R}$ represents the confidence towards prerequisite relation linking knowledge c_p to knowledge c_q. $\mathcal{G}$ can thus be represented as a series of edge tuples; for example, *(logic, 0.99, python)* means that *logic* is prior knowledge required for *python* with a confidence score of 0.99.

The task of Prerequisite-Driven Recommendation (PDR) aims to learn the latent factors not only from user–item interactions $\mathbf{Y}$ (where $y_{uv} = 1$ means u has interacted with v), but also from prerequisite context. We can view PDR as combining the knowledge linkage prediction task g and context-aware recommendation prediction task f. Specifically, it can be formalized as: 1) inferring latent prerequisites $\hat{r}_{c_i c_j} = g(c_i, c_j|\Phi, \mathcal{G})$, where $\hat{r}_{c_i c_j}$ represents the predicted prerequisite confidence from concept c_i to c_j; and 2) prerequisite-driven recommendation $\hat{y}_{uv} = f(u, v, \{c|c \in (C_u^p \cup C_u^t \cup C_v^i)\}|\Theta, \Phi, \mathrm{Y}, \mathcal{G})$. Here, Φ and Θ are the parameters for encoding knowledge concepts and users/items.

3 PDRS: PREREQUISITE-DRIVEN RS

Our instantiated framework (PDRS) is depicted in Figure 2. It consists of three components (§3.1, 3.2 and 3.3):

3.1 Prerequisite Knowledge Inference

To leverage prerequisite context, we first need to build a *prerequisite graph* with concept-level prerequisite links. We decompose this process into two subtasks: extracting concepts from item descriptions, and inferring prerequisite relation between concepts from the ordered item documents and general knowledge (topological relations in Wikipedia).

Concept Extraction. To extract key concepts from item documents (item description title, and item description content) as prerequisite context, we extend prior work [26] on graph propagation as a three-step subprocess. First, *seed concepts* are extracted from item v's document titles using TextRank [23] (with an empirically tuned threshold of 50%). Next, *candidate concepts* are gathered from v's document content, identifying all phrases that match the part-of-speech tag pattern for a noun phrase: $((A|N) + |(A|N)^*(NP)?(A|N)^*)N$ [13] (here, A, N, and P indicate adjectives, nouns, and prepositions). Lastly, we construct a fully-connected graph to include both seed concepts and candidate concepts, and expand the seed concept set to cover its relevant concepts. We implement this by iterative propagation, where concepts' confidence scores (where seed concepts are initially weighted with unit confidence) are propagated to their neighbors[1] regarding their semantic relatedness, as measured by cosine similarity $\omega(c_i, c_j) = cosine(h_{BERT}(c_i), h_{BERT}(c_j))$ of the textual meaning h from BERT [7].

Inferring Prerequisites. Knowledge concepts are then associated by prerequisite relations. As shown in Fig 2(a), we obtain the final linking scores by considering three features: i) asymmetric sequential interaction distance, ii) reference distance from general domain, and iii) semantic relatedness (as defined above). We explain the former two features:

(i) Asymmetric Sequential Interaction Distance. We observe that concepts covered in subsequently-consumed items are often dependent on those in previous ones; i.e., they are prerequisites. For example, courses that relate to *Bayes' Theorem* are more likely to appear in a user's interaction history *before* those related to *Hidden Markov Model.* We introduce Asymmetric Sequential Interaction Distance (AsyD) to model this, which captures the distributional pattern of knowledge concept pairs. Let's say $P(c_i, c_j) = \sum_u \sum_{(v_p, v_l) \in \mathcal{P}_u} tf(c_i, v_p) tf(c_j, v_l)$ indicates the probability of c_i preceding c_j in interaction histories, where $\mathcal{P}_u = \{(v_a, v_b) | v_a, v_b \in \mathcal{H}_u; ts_u(v_a) < ts_u(v_b)\}$ indicates item pairs ordered by timestamp ts in user historical interaction sequence $\mathcal{H}_u$ and $tf(c, v)$ denotes the term frequency of c in item v's documents (e.g., title and description). Specifically, if c_i is a prerequisite of c_j, the probability $P(c_i, c_j)$ should be greater than the converse association $P(c_j, c_i)$. Thus, our defined Asymmetric Sequential Interaction Distance is a normalized probability: $AsyD\left(c_i, c_j\right) = \sigma\{P(c_i, c_j)/P(c_j, c_i) - 1\}$. When c_i and c_j have arbitrary consumption order — *i.e.*, when they are independent of each other's distribution — then $AsyD = 0.5$.

(ii) Wikipedia Reference Distance. Textual evidence of prerequisites in domain documents can be sparse, resulting in noisy learned prerequisites. Contextual knowledge from general information sources, such as Wikipedia, can aid the identification of prerequisite relations by providing supplemental evidence. We observe that related general information that refer to domain concepts can also provide statistical evidence of prerequisites. To capture this, we propose a domain-adaptive Reference Distance ($RefD$) which builds on Liang's work [20] on Wikipedia. For example, the in-domain concept c – *"probability classifier"* may not occur in Wikipedia, we measure its reference imbalance through its related concepts t – *"bayes classifier"* in Wiki corpus. Specifically, when a related concept t is present in Wikipedia that refers to concept c, we capture its reference ratio, similar to $AsyD$, by calculating its normalized distance score. $RefD(c_i, c_j)$ measures the imbalance between each pair of in-domain concepts c_i and c_j via its associated reference imbalance τ to general concepts t. This process transforms the concept linking task to one of locating related concepts in Wikipedia. The reference imbalance $\tau = 1$ if and only if t_i refers to t_j in any Wikipedia article but where t_j never refers to t_i. As such, τ analogously ranges $[-1, +1]$. Thus, $RefD$ can smoothly incorporate information from general sources (τ), as well as in-domain sources (represented by $AsyD$); i.e., $RefD(c_i, c_j) = \sum_{i,j} \tau(t_i, t_j) \cdot S_{i,j} / \sum_{i,j} S_{i,j}$, where $S_{i,j} = \omega(t_i, c_i)\omega(t_j, c_j)$ is used to represent how closely c_i and c_j are linked to their related Wiki concepts, and the denominator allows $RefD$ to also be interpreted as a normalized probability.

Prerequisite Learning. To obtain the final prerequisite knowledge linkage scores $PKL(c_i, c_j)$, we train logistic regression over seed annotated samples ($n = 300$ per dataset), as shown on the right part of Fig. 2a. The regression is learned by adjusting the contribution weights for the three features $\omega, AsyDandRefD$ on manually-labeled concept pairs. We then run the regression to yield output for all knowledge concept pairs. This process reduces noise from the parameters, providing more accurate PKL scores for downstream recommendation (Fig. 2b). Our constructed prerequisite data are available at https://github.com/HoldenHu/PDRS/.

[1]For computational efficiency, we only propagate when edge scores are above a tunable parameter λ.

3.2 User/item and Prerequisite Context Encoding

The Encoding modules take the sparse input representations of users, items and concepts identified by PKL and encode them into dense representations (Fig. 2b [left]), employing a multi-layer perceptron (MLP) for both *Knowledge Encoding Module (KEM)* and *Behavior Encoding Module (BEM)*.

KEM learns the knowledge concept embedding by training pairs of (c_i, c_j) to approximate their previously-assigned prerequisite PKL scores. Our target in this step is to obtain every concept c's embedding $\mathbf{c} \in \mathbb{R}^{\mathbf{1 \times d}}$ by tuning the learnable parameters Φ to achieve the minimized $\mathcal{L}_{KEM}(r_{i,j}, \hat{r}_{i,j})$, where $r_{i,j}$ is expected prerequisite weight $PKL(k_i, k_j)$, and $\hat{r}_{i,j}$ is the estimated output from a MLP. BEM works similarly, learning user/item embeddings by minimizing the difference $\mathcal{L}_{BEM}$ between predicted $\hat{y}_{i,j} = MLP(u_i, v_j, \Theta)$ and the truth $y_{i,j}$. We select a mean-squared loss for $\mathcal{L}_{KEM}$ to best model the real-valued prerequisite scores, and a binary cross entropy loss for $\mathcal{L}_{BEM}$.

3.3 Recommendation Module

As shown in Fig. 2b [right], PDRS combines the embedded representation of user context (user prior knowledge concepts C_u^p and target knowledge concepts C_u^t) and item context (item concepts C_v^i) to output the final recommendation prediction $\hat{y}_{ij}$. We apply the average pooling for the concept embedding, while a user/item corresponds to multiple knowledge concepts. Formally, $\hat{y}_{uv} = MLP\{\mathbf{u}, \mathbf{v}, avg(\mathbf{C}_u^p), avg(\mathbf{C}_u^t), avg(\mathbf{C}_v^i)\}$, where BEM and KEM provide the pre-trained embeddings $\mathbf{u}$, $\mathbf{v}$, and $\mathbf{c}$ for the user, item and prerequisite contexts, respectively. We further tune the embedding parameters through joint training that alternately applies the objective of $\mathcal{L}_{KEM}$ and the final recommendation $\mathcal{L}_{Rec}$. $\mathcal{L}_{Rec}$ is also a binary cross entropy loss between model prediction $\hat{y}$ and the ground truth interaction y.

We empirically observe that the joint training of losses $\mathcal{L}_{KEM}$ and $\mathcal{L}_{Rec}$ enables the model to focus on prerequisite information. It also reduces the time complexity to a linear $O(|Y| + |R|)$ from the total quadratic complexity $O(\frac{|Y| \times |R| \times (dY + dR)}{dY \times dR})$ of alternatively executing the two objectives (where dY and dR represent the batch sizes of interaction and prerequisite pairs, respectively).

4 EXPERIMENTS

To the best of our knowledge, no existing datasets are specifically designed for *prerequisite context* modeling. Hence, we modify the existing datasets SSG-Data, MovieLens, and Amazon Books[2] which all contain textual descriptions of items. SSG-Data (**Course**) is an exhaustive anonymized listing of life-long course-taking history of citizen participants in 9 month period in *SkillsFuture Singapore (SSG)*, which is not publicly available yet. Course *content* and *objectives* are used as an item's documents. We also use the public MovieLens 100K and Amazon Books as **Movie** and **Book** datasets, where the crawled textual description from IMDB and TMDB[3] serve as movies' documents, and the crawled textual overview from Goodreads and Google Books[4] serve as books' documents. To align the task with (binary) recommendation, where only implicit feedback is available, we deem interactions with $ratings \geq 3$ as positive feedback in the movie and book scenarios. We provide detailed dataset statistics in Appendix A.

To obtain the user's state of knowledge C_u^p, C_u^t, we assume that users have mastered the knowledge contained in items that they have previously interacted with, and take concepts from the documents of first 30% and the last 20% of items they interacted with as their prior and target knowledge, respectively. To maintain strict training and testing separation, we only use the remaining 50% of knowledge concepts for training and testing our PDRS. We also follow the

[2](S) https://www.skillsfuture.gov.sg/, (M) https://grouplens.org/datasets/movielens/, and (A) https://jmcauley.ucsd.edu/data/amazon/
[3]https://www.imdb.com, and https://themoviedb.org
[4]https://www.goodreads.com/ and https://books.google.com/

Model		Course				Movie				Book			
		H@2	H@10	N@2	N@10	H@2	H@10	N@2	N@10	H@2	H@10	N@2	N@10
Without Prereq. Knowl.	ItemPop	0.4541	0.7813	0.3973	0.5300	0.2504	0.5905	0.2152	0.3496	0.2305	0.4740	0.2003	0.2865
	ItemKNN	**0.7122**	0.8076	**0.6551**	0.6785	0.2631	0.4613	0.2299	0.3092	0.0890	0.1178	0.0818	0.0944
	SVD	0.5934	0.8256	0.5863	0.6672	0.2720	0.6552	0.2288	0.3789	0.2025	0.4002	0.1791	0.2527
	BPR	0.6650	0.8396	0.6317	0.6794	0.2882	0.6673	0.2469	0.3958	0.2048	0.4043	0.1820	0.2712
	NeuMF	0.6859	0.8455	0.6257	0.6811	0.2899	0.6641	0.2467	0.3936	0.2178	0.4404	0.1876	0.2748
	GCMC	0.4081	0.7661	0.3471	0.4928	0.2518	0.5904	0.2165	0.3511	0.2234	0.4734	0.2234	0.2929
	DeepFM	0.5329	0.8154	0.4630	0.5799	0.2695	0.6457	0.2341	0.3821	0.2236	0.4431	0.1915	0.2751
With Prereq. Knowl.	GCMC+KEM	0.4512	0.7996	0.3821	0.5241	0.2618	0.6066	0.2266	0.350	0.2319	0.4885	**0.2251**	0.2992
	DeepFM+KEM	0.6786	0.8723	0.6101	0.6895	0.3034	0.6737	0.2125	0.4100	0.2365	0.4910	0.2026	0.3003
	PDRS	0.6894	**0.8789**	0.6390	**0.7142**	**0.3290**	**0.6993**	**0.2825**	**0.4281**	**0.2427**	**0.5150**	0.2110	**0.3172**

Table 1. Hit Ratio (H) and NDCG (N) @K on the Course, Movie, and Book datasets. Bold figures highlight best performers.

C^p	C^t	C^i	*Course*		*Movie*		*Book*	
			H@10	N@10	H@10	N@10	H@10	N@10
			0.8192	0.6117	0.6444	0.3796	0.4460	0.2749
✓			0.8626	0.6944	0.6598	0.4003	0.4579	0.2810
	✓		0.8556	0.6929	0.6714	0.4026	0.4669	0.2859
		✓	0.8422	0.6539	0.6731	0.3991	0.4684	0.2826
✓	✓		*0.8682*	*0.7099*	0.6705	0.4043	0.4644	0.2809
✓		✓	0.8633	0.6968	*0.6940*	*0.4208*	0.4898	0.3053
	✓	✓	0.8639	0.6978	0.6851	0.4134	*0.4966*	*0.3062*
✓	✓	✓	**0.8789**	**0.7142**	**0.6993**	**0.4281**	**0.5150**	**0.3172**

Table 2. Ablation study on PDRS. C^p, C^t and C^i represent user prior concepts, user target concepts, and concepts contained by item. Bold figures indicated leading performers; italicized figures, second-best.

	User Cold Start		*Item Cold Start*	
	H@10	N@10	H@10	N@10
ItemPop	0.7013	0.5010	-	-
SVD	0.347	0.2259	0.0661	0.0234
NeuMF	0.7097	0.4314	0.0036	0.0012
BPR	0.3787	0.2690	0.0915	0.0414
PDRS	**0.8337**	**0.6005**	**0.1989**	**0.1110**
PDRS (w/o $C_{u/v}$)	0.6381	0.3977	0.0729	0.0271

Table 3. Cold start evaluation on Course recommendation. PDRS uses knowledge from both user side and item side. PDRS w/o $C_{u/v}$ denotes the ablation of user/item context in user/item cold start, respectively.

common practice [18] and only retain user records with more than three interactions, to ensure each user has at least one item for evaluating recommendation performance and one item each for modeling prior and target knowledge.

We assess our PDRS method against traditional recommendation models (ItemPop, ItemKNN [28], and NeuMF [11]) and lightweight yet effective feature-incorporated models (GCMC [2] with optimized dropout rate 0.5, and DeepFM [10] with optimized dropout rate 0.2). We also compare against two-widely accepted models of GCMC and DeepFM to validate whether our prerequisite context representation is effective by replacing their user and item IDs feature inputs and modifying them to accept our KEM-derived 64-dimension feature embedding (denoted as GCMC+KEM and DeepFM+KEM). Specifically, the feature embedding dimensions, and node dropout rates are tuned for optimal performance, set as $\{8, 8, 16\}$, and 0.5, respectively. The optimal knowledge concept embedding dimension is set empirically at 64. We also employ two weak baselines: BPR [27] and SVD [14] for a comprehensive comparison.

We split our interaction data into (80%, 10%, 10%) to serve as training, validation and testing, respectively. For studying warm-start, we leave one item out per user. For the user (item) cold-start (also called new item) scenarios, users (items) in validation/testing set do not appear in the training set. For each test case, we follow the common practice [11], ranking 100 items — 99 negative samples and 1 positive sample. We use Hit Ratio@k (HR@k) and Normalized Discounted Cumulative Gain@k (NDCG@k) [30] as top-k ranking-based accuracy measures.

4.1 Main Results

Table 1 shows recommendation performance. In general, PDRS outperforms the baselines across all three datasets, in terms of both HR and NDCG, demonstrating the effectiveness of prerequisite context. One exception is that PDRS performs worse than ItemKNN on course recommendation when $k = 2$. We believe this is due to the differing levels of sparsity on the item side: there are $\frac{89K}{4.3K} = 20.7$ records per item for courses, but only $\frac{409K}{70K} = 5.8$ records per item for books. This makes it easier to find similar items in the former, but more difficult in the latter. This gain evaporates when k increases to 10, as ItemKNN only recommends accurately for users who choose courses similar to previous ones;

however we see that users do manifest individualized learning pathways in courses, which ItemKNN does not generalize to. GCMC and DeepFM perform well, outperforming other baselines, but they improve dramatically with addition of encoded prerequisite information from KEM, validating that prerequisites play an important role in recommendation tasks. The variants utilizing KEM pretraining (GCMC+KEM, DeepFM+KEM, PDRS) all improve over their corresponding base models. Among these three models, PDRS performs best.

To verify which form of side information in PDRS is most responsible for performance gains (i.e., user prior knowledge, user target knowledge, and item concept in Table 2), we conduct an ablation study (Table 2). When no side information is used, the model functions just as matrix factorisation. For all three recommendation scenarios, the results are best when all forms of side information are used, and worst when none are leveraged. The result of introducing a single form of side information shows that user context plays the most important role in course recommendation (+$\underline{5.3}$% with C^p, +4.4% with C^t, and +2.8% with C^i), whereas item context is more helpful for movie and book recommendation (+{2.4, 4.1, $\underline{4.4}$}% for movies and +{2.7, 4.7, $\underline{5.0}$}% for books on {C^p,C^t,C^i}, respectively. Interestingly, the three datasets exhibit different optimal combinations of two forms of side information. For courses, combining user prior and target knowledge is best, likely because learners do choose necessary bridging courses based on their target course. In contrast, in movie recommendation, the optimal combination is C^p and C^i. Watchers may choose based more on their experience with relatable plots and characters. For book recommendation, C^t and C^i is the best combination. We surmise that readers choose from their preferred book category (correlated in our PDRS by target knowledge).

4.2 Discussion: Is prerequisite context beneficial for Cold-start Problem?

User cold-start (also termed as new-user) and item cold-start problems [15, 29] are major recommendation system concerns. Table 3 compares PDRS in the course recommendation against baselines in user and item cold-start scenarios.

From the table, item cold start is more serious than user cold start, as evidenced by the drastic drop in performance in the former case. Missing item interactions — in addition to high user sparsity — makes item and user representations inaccurate. Note that as ItemKNN locates items similar to users' previous interactions and ItemPop uses item popularity, both baseline models do not apply to this item cold start scenario. The performance of latent factor based approaches (SVD, BPR) severely drops in such cold start scenarios. NeuMF, being able to learn nonlinear relationships, is aware of more complex information from interaction data, yielding comparable HR@10. PDRS significantly outperforms baselines in both cold start scenarios, validating the value added by prerequisite knowledge modeling in recommendation accuracy. By comparing the use of user side information in user cold start problem, we observe the advantage of user information. The same applies to items with item information linking items to users through knowledge linkage.

4.3 Discussion: Is our induced prerequisite knowledge accurate?

PDRS relies on accurate and comprehensive prerequisite inference from documents. To verify the reliability of our automatically compiled prerequisites, we also conduct a quality evaluation of our induced prerequisite graphs.

Recall that during relation inferring in prerequisite graphs (§ 3.1), we train a logistic regression model to predict the strength of prerequisite edges between concepts. We validate our PKL score fitted from the use of features, and evaluate using precision, recall and F1. We take 80% as training samples and the remaining 20% as test. We compare against two baselines in this knowledge learning prediction task: 1) Hyponym Pattern Method (HPM) [36]: detects whether prerequisites among noun phrases pairs in sentences fulfill 10 lexico-syntactic patterns (e.g., "*Python* (NP1), one of the *Programming Language* (NP2)"). 2) Reference Distance (RD) [20]: where we modify RD's feature extraction

Approach	Precision	Recall	F1
HPM	66.68	16.09	25.92
RD	59.37	47.44	52.79
AsyD (Ours)	**77.08**	60.66	67.89
PKL (Ours)	76.00	**62.29**	**68.47**

Table 4. Prerequisite extraction performance (%) on our Course dataset. AsyD is an ablated form of our prerequisite inferring component by testing only the Asymmetric Distance (AsyD) module. Bold figures highlight the best performer.

Domain	Knowledge	PKL	Knowledge	PKL	Knowledge	PKL
Course	Python	0.38	Code	0.50	Feature Learning	0.73
	Machine Principle	0.34	Program	0.50	Deep Learning	0.78
	Computer Basic	0.34	Database	0.50	Algorithm Analysis	0.73
Movie	Voldemort	0.38	Policeman	0.56	Handsome Boy	0.75
	Triwizard Tournament	0.34	Battle	0.50	Europe	0.78
	Evil Dragon	0.31	Merchant	0.50	Fate of Human	0.93
Book	Scientist	0.39	Atom Bomb	0.50	Space Travel	0.91
	21^{th} Centuries	0.35	Flash	0.50	Ethologist	0.80
	World War	0.35	Earthquake	0.50	Star Trek	0.87

Table 5. Example Prerequisite Knowledge Linkage (PKL) scores for items PKL(*'Machine Learning'*, knowledge) from Course, PKL(*'Harry Potter'*, knowledge) from Movie, and PKL(*'Alien'*, knowledge) from Book.

methodology to be able to apply it to our task (*cf* 3.1) by empirically tuning a symmetric threshold $\theta = 0.15$, which is used for an $RD(k_i, k_j)$ of $[-1, -\theta)$, $[-\theta, \theta]$, $(\theta, 1]$ denoting a posterior, neutral, or prior prerequisite relationship.

Table 4 shows the macroscopic comparative results against the Course ground truth. Both AsyD and the final PKL outperform the other two baselines by large margins (+15 F1). The performance improvement is mostly attributed to domain-specific features (AsyD is much better than RD), whereas general domain features brought in by the final PKL bring a minor boost. We zoom in on a typical microscopic case study of the *machine learning* knowledge concept from the Course dataset. Table 5 lists knowledge concepts that frequently co-occur with *machine learning*. Large, positive PKL scores indicate that *machine learning* is a prerequisite of the knowledge concept. We see that concepts on the right, high-PKL column have higher probability of being recommended to users who master *machine learning*, compared to concepts on the left, low-PKL side. As an example, people taking up a course on *machine learning* usually have learned *python*. Prerequisite linking in our other two datasets is less intuitive, but meaningful nonetheless.

5 RELATED WORK

Context-aware recommender systems handle static metadata as auxiliary information, such as user profiles [17, 38] and item attributes [4]. Textual content is a typical form with rich contextual information to facilitate RS accuracy. Some works treat item content as raw features by hidden vectors [31, 40], while others select important text pieces, such as item tags [8, 19], and semantic clues [35]. However, few works focus on establishing context causality between user and item, where our approach uses prerequisite context.

Proper **Prerequisite Relation Identification** is thus crucial for both intrinsic prerequisite representation task and our ultimate extrinsic task of recommendation. Many works rely on statistical methods to determine prerequisites. An early study by Vuong et al. [34] examined the effect of learning curriculum units in various orders. Chen et al. [6] treated prerequisite relations as a Bayesian network, which requires a mapping of courses to fine-grained skill and relevant student performance data. Chen et al. [5] apply probabilistic association rule mining to infer student knowledge from performance data. To make the prerequisite relation identification more feasible, others — including ourselves — tap into generic information sources. Pan et al. [25] utilize a Wikipedia corpus to learn semantic representation of concepts for detecting prerequisites in MOOC. Talukdar and Cohen [33] study how prerequisites can be inferred between Wikipedia entities. Wang et al. [36] use Wikipedia articles and categories for Concept Graph Learning that uses observed prerequisite relation to learn unobserved ones. Although this task has mostly been applied to education [39], our findings emphasize that prerequisites indeed generalize and do not need to be restricted to a particular context.

6 CONCLUSION AND FUTURE WORK

To the best of our knowledge, we are the first to explore the use of prerequisites — an overlooked but crucial form of context — for recommendation. We introduce Prerequisite Knowledge Linking (PKL) method to induce a prerequisite graph automatically, through semi-supervised learning over both general and domain-specific features. We instantiate our formalism in the form of a Prerequisite Driven Recommendation System (PDRS; §3) embodied as a modern neural architecture, which adopts joint training to optimise the model for the twin objectives of knowledge linking prediction and recommendation. We demonstrate that prerequisite context is a functional booster to solve cold-start problem, and can benefit recommenders universally through our experiments on our Course, Movie, and Book datasets (§4).

While our PDRS is a simple instantiation of a prerequisite driven recommendation, its elegance leads to synergistic performance gains. Designing more sophisticated models to leverage captured prerequisite knowledge is open future work. As our prerequisite graph is a structured format of knowledge, future work may seek more complex encoding methods, such as typical translation-based methods (e.g., TransE [3], TransH [37]). Moreover, our work opens the door for studying user's state of knowledge in dynamic scenarios, such as conversational recommendation and long-term sequential recommendation.

REFERENCES

[1] Rakesh Agrawal, Behzad Golshan, and Evangelos Papalexakis. 2016. Toward data-driven design of educational courses: a feasibility study. *Journal of Educational Data Mining* 8, 1 (2016), 1–21.

[2] Rianne van den Berg, Thomas N Kipf, and Max Welling. 2017. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263* (2017).

[3] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems* 26 (2013).

[4] Xu Chen, Hanxiong Chen, Hongteng Xu, Yongfeng Zhang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2019. Personalized fashion recommendation with visual explanations based on multimodal attention network: Towards visually explainable recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 765–774.

[5] Yetian Chen, José P González-Brenes, and Jin Tian. 2016. Joint discovery of skill prerequisite graphs and student Models. *International Educational Data Mining Society* (2016).

[6] Yang Chen, Pierre-Henr Wuillemin, and Jean-Marc Labat. 2015. Discovering Prerequisite Structure of Skills through Probabilistic Association Rules Mining. *International Educational Data Mining Society* (2015).

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[8] Yuyun Gong and Qi Zhang. 2016. Hashtag recommendation using attention-based convolutional neural network.. In *IJCAI*. 2782–2788.

[9] Jonathan Gordon, Linhong Zhu, Aram Galstyan, Prem Natarajan, and Gully Burns. 2016. Modeling concept dependencies in a scientific corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 866–875.

[10] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. *arXiv preprint arXiv:1703.04247* (2017).

[11] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*. 173–182.

[12] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[13] John S Justeson and Slava M Katz. 1995. Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural language engineering* 1, 1 (1995), 9–27.

[14] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[15] Xuan Nhat Lam, Thuc Vu, Trong Duc Le, and Anh Duc Duong. 2008. Addressing cold-start problem in recommendation systems. In *Proceedings of the 2nd international conference on Ubiquitous information management and communication*. 208–211.

[16] Stephen Laurence and Eric Margolis. 1999. Concepts and cognitive science. *Concepts: core readings* 3 (1999), 81.

[17] Chenyi Lei, Dong Liu, Weiping Li, Zheng-Jun Zha, and Houqiang Li. 2016. Comparative deep learning of hybrid representations for image recommendations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2545–2553.

[18] Wenqiang Lei, Gangyi Zhang, Xiangnan He, Yisong Miao, Xiang Wang, Liang Chen, and Tat-Seng Chua. 2020. Interactive path reasoning on graph for conversational recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2073–2083.

[19] Yang Li, Ting Liu, Jing Jiang, and Liang Zhang. 2016. Hashtag recommendation with topical attention-based LSTM. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 3019–3029.

[20] Chen Liang, Zhaohui Wu, Wenyi Huang, and C Lee Giles. 2015. Measuring prerequisite relations among concepts. In *Proceedings of the 2015 conference on empirical methods in natural language processing*. 1668–1674.

[21] Chen Liang, Jianbo Ye, Zhaohui Wu, Bart Pursel, and C Giles. 2017. Recovering concept prerequisite relations from university course dependencies. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 31.

[22] Amit Livne, Moshe Unger, Bracha Shapira, and Lior Rokach. 2019. Deep context-aware recommender system utilizing sequential latent context. *arXiv preprint arXiv:1909.03999* (2019).

[23] Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*. 404–411.

[24] Matthew W Ohland, Amy G Yuhasz, and Benjamin L Sill. 2004. Identifying and removing a calculus prerequisite as a bottleneck in Clemson's General Engineering Curriculum. *Journal of Engineering Education* 93, 3 (2004), 253–257.

[25] Liangming Pan, Chengjiang Li, Juanzi Li, and Jie Tang. 2017. Prerequisite relation learning for concepts in moocs. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1447–1456.

[26] Liangming Pan, Xiaochen Wang, Chengjiang Li, Juanzi Li, and Jie Tang. 2017. Course concept extraction in moocs via embedding-based graph propagation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 875–884.

[27] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).

[28] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. 285–295.

[29] Andrew I Schein, Alexandrin Popescul, Lyle H Ungar, and David M Pennock. 2002. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*. 253–260.

[30] Guy Shani and Asela Gunawardana. 2011. Evaluating recommendation systems. In *Recommender systems handbook*. Springer, 257–297.

[31] Peijie Sun, Le Wu, Kun Zhang, Yanjie Fu, Richang Hong, and Meng Wang. 2020. Dual learning for explainable recommendation: Towards unifying user preference prediction and review generation. In *Proceedings of The Web Conference 2020*. 837–847.

[32] Zhu Sun, Qing Guo, Jie Yang, Hui Fang, Guibing Guo, Jie Zhang, and Robin Burke. 2019. Research commentary on recommendations with side information: A survey and research directions. *Electronic Commerce Research and Applications* 37 (2019), 100879.

[33] Partha Talukdar and William Cohen. 2012. Crowdsourced comprehension: predicting prerequisite structure in wikipedia. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*. 307–315.

[34] Annalies Vuong, Tristan Nixon, and Brendon Towle. 2011. A method for finding prerequisites within a curriculum.. In *EDM*. 211–216.

[35] Heyuan Wang, Fangzhao Wu, Zheng Liu, and Xing Xie. 2020. Fine-grained interest matching for neural news recommendation. In *Proceedings of the 58th annual meeting of the association for computational linguistics*. 836–845.

[36] Shuting Wang, Alexander Ororbia, Zhaohui Wu, Kyle Williams, Chen Liang, Bart Pursel, and C Lee Giles. 2016. Using prerequisites to extract concept maps fromtextbooks. In *Proceedings of the 25th acm international on conference on information and knowledge management*. 317–326.

[37] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 28.

[38] Qidi Xu, Fumin Shen, Li Liu, and Heng Tao Shen. 2018. Graphcar: Content-aware multimedia recommendation with graph autoencoder. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 981–984.

[39] Yiming Yang, Hanxiao Liu, Jaime Carbonell, and Wanli Ma. 2015. Concept graph learning from educational data. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*. 159–168.

[40] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the tenth ACM international conference on web search and data mining*. 425–434.

Appendix A DATASET STATISTICS

Dataset	# Users	# Items	# U–I	# Concepts	# C–C Pairs
Course	34,235	4,322	89,565	5,131	227,515
Movie	608	7,354	49,738	12,503	8,402,493
Book	92,971	70,695	409,616	58,492	15,889,953

Table 6. Statistics of the datasets used in our experiments. U–I: user–item interactions; C–C pairs: concept-concept pairs.

Detailed dataset statistics are shown in Table 6. Our Course dataset is sparser per user, validating observations that educational activities are hard to recommend without any side information. We follow the commonly-used annotation method [9, 21, 25] in inferring prerequisite relations: manually labeling a small number of concept pairs to train a logistic regression to label the rest. We annotated 300 pairs (c_i, c_j) for each task with +1 if c_i is c_j's prerequisite, −1 if c_j is c_i's prerequisite, or 0 if no obvious dependency relation exists. Of the 300 labeled samples 80% = 240 are used for model training, and the remaining 20% = 60 are held out for testing. The main purpose of this process is to *de-noise* the PKL scores by considering three indicators, discussed in § 4.3.

Appendix B DETAILED MICROSCOPIC CASE STUDY OF GENERATED PREREQUISITE GRAPHS

As shown in Table 5, prerequisite linking in our other two datasets feels less intuitive, but is meaningful nonetheless. Taking selected concepts from our Movie dataset for the film *'Harry Potter'* as an example, we see that the film is learned as a prerequisite for the knowledge concept *'Fate of Human'*. This means that the source IMDB and TMDB documents mentioning *'Harry Potter'* mention *'Fate of Human'* but not vice versa, leading to a large PKL score, hence a prerequisite. Casual inspection of the general Web confirms that *Harry Potter* is a foil for subsequent discussions about fate and free will, which makes sense. While the prerequisite does not constitute a strictly sequential viewing order among *'Harry Potter'* and other movies featuring fate and free will as a central theme, these soft constraints do model inspiration; in Books, people who have read *'Alien'*-themed books may go on to choose similarly themed *'Space Travel'* books, including *'Star Trek'* novels.

We also find that prerequisite graphs have different structural properties. Course prerequisite graphs contain the richest dependency relations (i.e., longer average path and larger average node degree), while graphs for the Movie and Book datasets are sparser and shallower. Our casual observation of the graphs also reveal that longer prerequisite paths correlate with more specific terms (e.g., *'Voldemort'*), while the shorter paths refer to more general knowledge (*'Love'*). Both studies show consistency, in that movies and books have fewer dependencies on required prerequisites, as compared to the formal knowledge acquired in courses, but that such soft constraints still aid recommendation accuracy.

Appendix C THE ROLE OF PREREQUISITES IN RECOMMENDATION

C.1 What embedding size is most suited for prerequisite representation?

Here, we use Root Mean Squared Error (RMSE) and R Squared (R2) to measure the fidelity of the KEM prediction; that is, after encoding. In the figure, KEM (Knowledge Encoding Module) refers to the results from knowledge encoding module alone (ablating recommendation), RM (Recommendation Module) refers to the training of recommendation target without constraints from KEM, and PDRS applies both sets of constraints.

Both plots in Figure 3 show a consistent trend: adding more latent dimensions improves knowledge representation and prerequisite capture. Furthermore, the joint training in PDRS of both $\mathcal{L}_{PKL}$ and $\mathcal{L}_{Rec}$ shows a slight but consistent benefit. Both exhibit large improvements compared against optimizing RM alone. Seen this way, good prerequisite capture through PKL is its own reward, but has the free side effect of benefitting downstream recommendation as well.

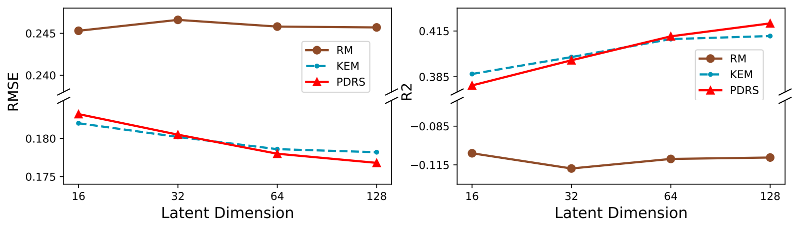


Fig. 3. Effect of varying the dimensionality of the knowledge concept encoding (x-axes) on downstream recommendation performance (y-axes, as measured in (l) RMSE (lower better) and (r) R2 (higher better).

C.2 Do prerequisites complement user–item interaction?

Unlike collaborative filtering, prerequisites play a decisive and logical factor in driving users' sequential interactions with items. While collaborative filtering may find quality recommendations, such recommendations may not account for the immediate needs of the user.

Take the first row of Table 7 as in example of the role of such knowledge. Here, the left portion of the table gives the input user's state of knowledge and the right portion shows the output recommendation from both the interaction-only BEM and our full PDRS. We see this user has handled basic office skills, and thus she may be interested in other related skills such as team work, or soft social skills, which may be recommended by collaborative filtering as is done in BEM. But perhaps their priority is in building on existing credentials: here, *'Microsoft Word'* can be seen as a more efficient means of *'text'* entry.

User Context	Model	First Recommended Item's Knowledge Concepts
PriorK: 'sort', 'text', 'ribbon', 'page orientation', 'manipulation'	*PDRS*	'read', 'document', 'page', 'undo', 'microsoft word'
TargetK: 'file', 'combo box', 'footnote', 'layout', 'form', 'controller'	*BEM*	'team', 'development', 'plan', 'social response', 'application'
PriorK: 'muscle', 'tissues', 'bone', 'mobile', 'neck'	*PDRS*	'autonomic nervous system', 'profile', 'nervous system', 'area'
TargetK: 'privacies', 'skill', 'certifier', 'participant'	*BEM*	'compliance', 'requirement', 'response', 'investigation'
PriorK: 'protein', 'sugar', 'food product', 'selection'	*PDRS*	'bread', 'baking'
TargetK: 'tour', 'participation', 'competition', 'visitor'	*BEM*	'preventive act', 'report', 'unit'
PriorK: 'conversation', 'spot', 'paper','detection'	*PDRS*	'data analysis', 'skill', 'sql', 'in demand', 'database'
TargetK: 'database', 'product', 'real world', 'code', 'data feature', 'universe'	*BEM*	'facial', 'neck', 'eye', 'client', 'facial car'

Table 7. Course examples where PDRS behaves differently. Underlined concepts are strongly induced by PKL.

Can prerequisite knowledge be derived from user–item interaction directly? In short, yes, but much less effectively as we have done in PDRS. To illustrate the fine-grained level of useful prerequisite knowledge captured (item or concept), we also benchmark PDRS with coarser, obvious prerequisites. We build another item dependency graph directly from the interaction sequence order in our Course scenario. We set an aggressive threshold, keeping only the top ~250 item pairs satisfying a minimal number of occurrences and prerequisite strength. We run these identified, salient prerequisites through BEM but find that performance suffers significantly, leading to a 3.6% drop in performance. This finding direct attributes the power of the fine-grained prerequisite capture, and the effectiveness of PDRS in incorporating general features enhance the signal coming from sequential interaction histories.

Appendix D DISCUSSION: IS PDRS SENSITIVE TO HYPERPARAMETER SETTING?

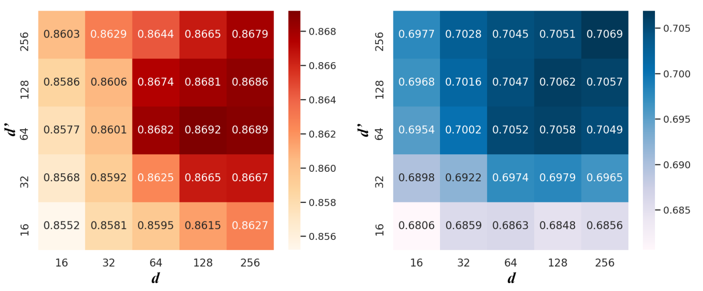
We examine how PDRS handles both prerequisite context and user/item encodings as the model complexity (in terms of hidden layers) is varied. As shown in Table 8, models with or without pretraining both perform best with $L = 4$. Fewer layers are insufficient to learn the complex relationship between embeddings (especially for PDRS to learn the relation between knowledge embedding from both user and item), whereas larger numbers suggest overfitting.

PDRS with pretraining achieves better performance with more hidden layers, but is worse than the case without pretraining, when number of layers is small (i.e., $L = 1, 2, 3$). It can be seen that using pretraining may yield more accurate user/item and knowledge embeddings as initial values for PDRS. Again, too few layers may be insufficiently rich to model embeddings for recommendation.

	With Pre-training		**Without Pre-training**	
# Layers	**HR@10**	**NDCG@10**	**HR@10**	**NDCG@10**
1	0.7965	0.5884	**0.8042**	**0.5961**
2	**0.8543**	0.6717	0.8542	**0.6723**
3	0.8592	0.6937	**0.8613**	**0.6986**
4	**0.8703**	**0.7051**	0.8682	0.7028
5	**0.8641**	**0.6993**	0.8633	0.6984
6	**0.8640**	**0.6989**	0.8638	0.6932

Table 8. Performance of PDRS with varying numbers of layers, with and without pretraining. Bold figures indicate the better strategy w.r.t. each number of layers.

We also examine the impact of the combination of the dimension d used for BEM and the dimension d' used for KEM on Course recommendation. As can be seen from the heat map in Figure 4, NDCG@10 improves as d and d' increases. This supports using more factors to store the latent signals and thus improving the model capacity. The best HR@10 is achieved when $d = 128, d' = 64$, and larger models tend to overfit.



(a) HR@10 (b) NDCG@10

Fig. 4. Top-10 recommendation Hit Ratio and NDCG scores w.r.t. various # latent factor d in BEM and d' in KEM (# layer = 4).