Prerequisite-Driven Deep Knowledge Tracing

Penghe Chen[1], Yu Lu[1,2,*], Vincent W. Zheng[3], Yang Pian[1]
[1]*Advanced Innovation Center for Future Education, Beijing Normal University, Beijing, China*
[2]*School of Educational Technology, Faculty of Education, Beijing Normal University, Beijing, China*
[3]*Advanced Digital Sciences Center, Singapore*
chenpenghe@bnu.edu.cn, vincent.zheng@adsc.com.sg, bianyang@bnu.edu.cn
[*]*Corresponding Author: Yu Lu, luyu@bnu.edu.cn*

***Abstract*—Knowledge tracing serves as the key technique in the computer supported education environment (e.g., intelligent tutoring systems) to model student's knowledge states. While the Bayesian knowledge tracing and deep knowledge tracing models have been developed, the sparseness of student's exercise data still limits knowledge tracing's performance and applications. In order to address this issue, we advocate for and propose to incorporate the knowledge structure information, especially the prerequisite relations between pedagogical concepts, into the knowledge tracing model. Specifically, by considering how students master pedagogical concepts and their prerequisites, we model prerequisite concept pairs as ordering pairs. With a proper mathematical formulation, this property can be utilized as constraints in designing knowledge tracing model. As a result, the obtained model can have a better performance on student concept mastery prediction. In order to evaluate this model, we test it on five different real world datasets, and the experimental results show that the proposed model achieves a significant performance improvement by comparing with three knowledge tracing models.**

***Index Terms*—Knowledge Tracing, Prerequisite Modeling, Deep Learning**

I. Introduction

Knowledge tracing aims to model the knowledge states of individual students through quantitatively diagnosing a student's mastery level on each concept [9]. Traditionally, knowledge tracing is one of the key components and techniques to enable both personalized learning and teaching, where students can choose what they need to practice more and teachers can decide which concepts to teach more to students [2], [3], [30]. The rapid developed computer supported education environment and online education platforms provide abundant students' exercise data for knowledge tracing. Yet, it also comes with a major challenge, which is the sparseness of student's behaviors. Specifically, students tend to use the system in an irregular time basis, and spend less time to practice for a comprehensive assessment of their concept mastery. Thus for each student, it is very likely that only a small set of concepts in each subject are practiced (by solving quiz problems) or learned (by browsing tutorial videos or interacting with teachers). The mastery levels of a large portion of concepts remain unknown, which makes the knowledge tracing suffer from accuracy issue and be restricted for further applications (e.g., student performance prediction [15], question difficulty prediction [14], curricula optimization [22]).

Our insight of solving the sparseness problem hinges upon the idea of fully exploring the interdependencies between concepts from the knowledge structure, especially the prerequisites between pedagogical concepts. The concepts usually do not exist alone: some concepts serve as the prerequisites of other concepts, thus if we know a student has mastered a target concept, then we are sure that she has also mastered its prerequisite concepts. Modeling prerequisites for knowledge tracing is greatly under-explored. Most existing work that studies prerequisites is mainly about mining prerequisites, rather than modeling them to assist knowledge tracing [16], [17], [20], [29]. Some other work tries to use prerequisites for concept map extraction [27], but it still does not consider how to use prerequisites for knowledge tracing. There is limited attempt to use prerequisites for student modeling [5], [7], but they do not specifically use deep learning for the knowledge tracing purpose.

In this paper, we propose a novel way to model prerequisites for knowledge tracing. In a nutshell, our intuition is that prerequisites actually represent concept learning dependencies, which naturally serves as **constraints** on the student concept mastery prediction. Let us consider an example. Suppose there are two concepts k_1 and k_2, and k_1 is the prerequisite of k_2. In general, it means that if a student has mastered k_2, then most likely she also masters k_1; besides, if the student does not master k_1, then she unlikely masters k_2 [4]. By utilizing this property, we propose a knowledge tracing model with prerequisites as constraints, which actually creates links between questions. As a result, the exercise data sparseness issue can be relieved and we can now achieve a better performance on estimating each student's mastery on the concepts.

However, even though modeling prerequisites as constraints for knowledge tracing model is promising for data sparseness issue, there still remains a main challenge on how to model prerequisites as constraints. There are many different ways to model constraints, but how to express and formulate prerequisites into a proper mathematical form is not straightforward. For example, one can model the constraints as logic [1], [23], or boolean functions [6], [11] in regularization, or a conditional probability [32] to maximize in the objective function. Each such way of constraint modeling, as we shall see later in detailed mathematical forms, is not trivial to formulate prerequisite relations between pedagogical concepts in a deep knowledge tracing setting.

978-1-5386-9159-5/18/$31.00 ©2018 IEEE
DOI 10.1109/ICDM.2018.00019

To address this challenge, by analyzing the characteristics of prerequisites, we propose to model each prerequisite as an *ordering pair* between a student's mastery levels on two concepts. That is, suppose concept k_1 is the prerequisite of concept k_2; then we enforce the probability of a student mastering k_1 as always larger than that of her mastering k_2 at any time later. As we shall see later in derivation details, such an ordering pair has a nice mathematical formulation, which is easily explainable and computationally tractable. Finally, we put forward an order pair regularized deep knowledge tracing framework to incorporate the prerequisite constraints, which will have a better performance on students' knowledge states estimation.

We can summarize our contributions in this work as follows:

- We study an important problem of how to incorporate prerequisites for assisting deep knowledge tracing, which essentially introduces the subject concepts and their structure information into the knowledge tracing domain;
- We explore a novel approach to model the prerequisites as constraints through a new knowledge tracing model;
- We evaluate our model on real-world datasets and the results show that our model improves the state-of-the-art baselines.

The rest of this paper is organized as follows: Section II reviews the existing work and how our model is different from those work. Section III gives a clear description on the problem definition and Section IV illustrates how the exercise sequence data can be modeled. Details of the proposed model are presented in Section V. Section VI presents experimental results on five real world datasets and the whole paper is concluded in Section VII.

II. Related Work

Researchers have explored different methods to do knowledge tracing. Earlier work focuses on Bayesian knowledge tracing models which define each student's knowledge state as a binary variable and utilize probabilistic models like Hidden Markov Model to estimate students' concept mastery [9]. Subsequently, researchers improve the model with considering cognitive factors like slipping and guessing [2]. Prior knowledge of individual students [30] and difficulty levels of questions [21] are also explored to improve those Bayesian knowledge tracing models. In order to handle the partially correct responses on subjective problems, a fuzzy cognitive diagnosis framework is proposed [28]. Recently, with the development of deep learning, deep knowledge tracing model is proposed to model the students' exercise sequences [22], and dynamic key-value memory network based model is also proposed to better represent question semantics [31]. In addition, with the advances of recommender system, matrix factorization based method was also proposed to model the students' exercise data [26]. Compared with the these knowledge tracing models, we are special in both using deep learning (especially prerequisite-driven recurrent neural networks) for knowledge tracing, and also exploring prerequisites in the deep knowledge tracing setting.

Utilizing prerequisite information to solve the knowledge tracing problem is still limited in previous work. With modeling the knowledge tracing problem as a Bayesian network, [5] tries to introduce the prerequisites as a new layer in the Bayesian network. In addition, one recent work tries to extend the idea of Bayesian network modeling of prerequisites to jointly discover the prerequisite graph and estimate the student concept mastery [7]. Compared with these two methods, we consider knowledge tracing in a deep learning setting, and we systematically explore prerequisites.

Another closely related research is constraint learning, which has been a popular approach for incorporating domain knowledge in prediction. For example, Markov Logic Network (MLN) [23] models constraints in the form of first-order logic. Comparatively, our modeling the student concept mastery in an ordering pair form is more flexible than logic to formulate the prerequisites. There are also non-logic forms of modeling constraints, *e.g.*, Constraint-Driven Learning [6], Generalized Expectation [18], Posterior Regularization [11], and Robust RegBayes [19], but all these methods do not consider how to use constraints in deep learning models. In addition, some recent attempts try to use logic rules in knowledge graph embedding [12], deep neural networks [13], and text embedding for relation extraction [24]. However, they are designed for different applications other than deep knowledge tracing, and they have to model the constraints in a logic form.

III. Problem Formulation

Knowledge tracing aims to estimate students' knowledge states based on students' exercise data. As questions of exercise also link to different subject concepts, knowledge tracing models also consider concept information. Hence, the knowledge tracing problem concerns three main entities: student, question and concept. In this work, we denote a set of students as $\mathcal{U}$, a set of questions of exercise as $\mathcal{Q}$, and a set of concepts as $\mathcal{C}$. We further denote a result of student $i \in \mathcal{U}$ answering question $j \in \mathcal{Q}$ as $y_{i,j} \in \{0, 1\}$, where $y_{i,j} = 1$ indicates a correct answer and $y_{i,j} = 0$ indicates an incorrect answer. In addition, we use following definitions to denote some important terms.

Definition 1: An *exercise sequence* $\mathbf{s}_i$ for student u_i is a sequence of question-result pairs $\mathbf{s}_i = \{(\pi(i,t), y_{i,\pi(i,t),t}) | t = 1, ..., n_i\}$, where $\pi(i,t) : \mathcal{U} \times \mathcal{T} \rightarrow \mathcal{Q}$ returns the question that student i answers at time $t \in \mathcal{T}$, and $n_i > 0$ is the sequence length. This kind of data can be easily obtained through intelligent tutoring systems or online education platforms. This exercise sequence data is usually represented as a student-question matrix, and a toy example is shown by the "Student-Question Matrix" in Figure 1. In this matrix, 1 and 0 denote students answering questions correctly and wrongly respectively. It is not compulsory for all the students to answer all the questions, so some students may not answer certain questions, which is denoted by "-" in the matrix. This is also one of the reasons for the data sparseness issue discussed in the Introduction section.

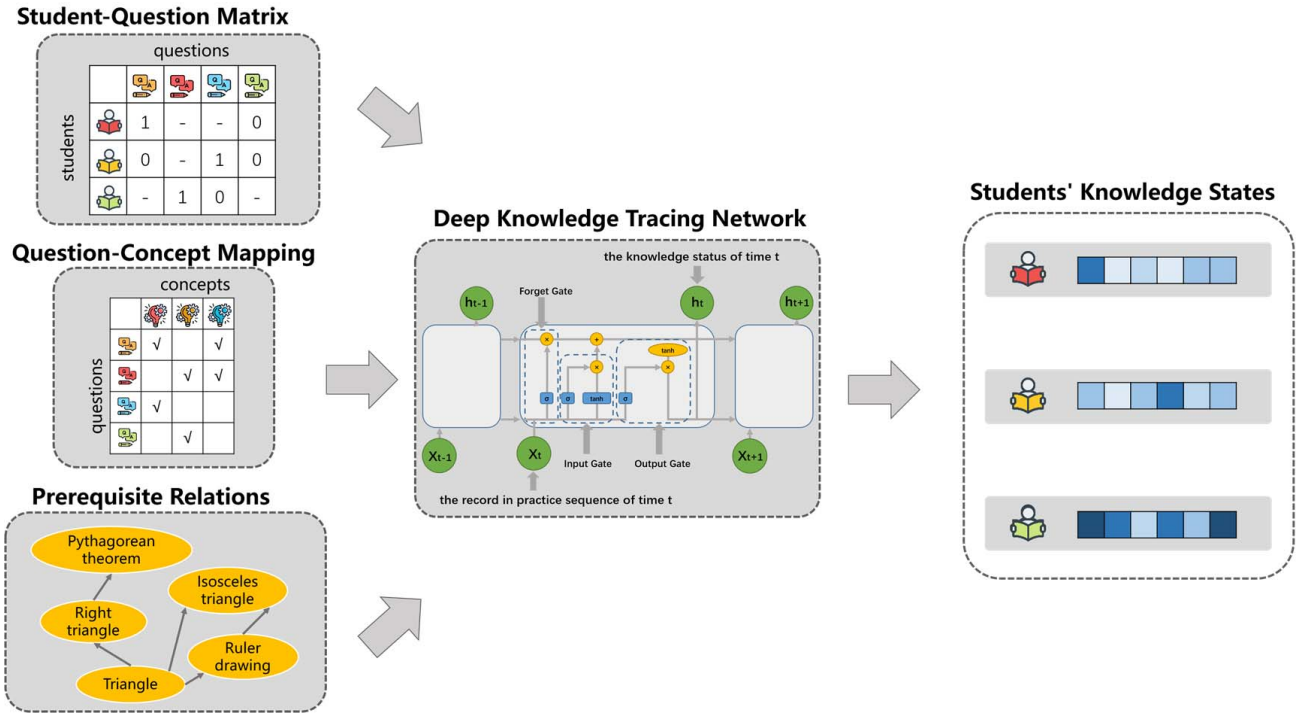


Fig. 1: Illustration of knowledge tracing problem.

Definition 2: A *question-concept matrix* $O \in \{0,1\}^{|\mathcal{Q}|\times|\mathcal{C}|}$ is a binary matrix, where $O_{j,k} = 1$ if question $j \in \mathcal{Q}$ contains concept $k \in \mathcal{C}$, and $O_{j,k} = 0$ otherwise. Obtaining this data is not trivial and there are two main approaches to do it: either automatically recognized by computer systems or manually labeled by experts. There is research work utilizing machine learning algorithms to identifying concepts contained in each question [31]. In this work, since the focus is knowledge tracing rather than identifying concept in questions, we utilize the manual approach to obtain this matrix data. A small example of the matrix is shown with the "Question-Concept Mapping" matrix in Figure 1. For the purpose of better visualization, this example drops all 0s and replace all 1s with the "✓" mark.

Definition 3: A *prerequisite matrix* $E \in \{0,1\}^{|\mathcal{C}|\times|\mathcal{C}|}$ is a binary matrix, where $E_{k_1,k_2} = 1$ if concept $k_1 \in \mathcal{C}$ is the prerequisite of concept $k_2 \in \mathcal{C}$, and $E_{k_1,k_2} = 0$ otherwise. Similar to the *question-concept matrix*, Identifying the prerequisite relations between pedagogical concepts can either be done automatically through algorithm like [27], or be manually labeled by experts. In this work, a manually labeled prerequisite matrix is utilized in model testing. Prerequisites are links between concepts, so this matrix naturally creates a graph of concepts. A small prerequisite graph is shown by the "Prerequisite Relations" diagram in Figure 1, in which five different prerequisite between five mathematical concepts are presented.

With the above definition, the problem *input* of knowledge tracing can be summarized as: 1) a student-question matrix, which includes a set of questions $\mathcal{Q}$, a set of students $\mathcal{U}$, and each student $i \in \mathcal{U}$ has an exercise sequence $\mathbf{s}_i$; 2) a question-concept matrix O; 3) a concept prerequisite matrix E. In addition, for the problem *output*, we aim to learn: a vector $\mathbf{h}_{i,t} \in \mathbb{R}^d$ indicating student i's knowledge states at time t. Based on the output, we expect to easily compute a student's level of mastering a concept.

Hence, the whole problem can be illustrated by Figure 1. It takes the three data matrices as input, utilizes a knowledge tracing model (e.g. deep learning network model) to process those data, and generates the knowledge states for different students. For easier checking, a list of notations used in this work is summarized in Table I.

IV. Modeling Exercise Sequence

We model each exercise sequence $\mathbf{s}_i$ with a Gated Recurrent Unit (GRU) [8]. The model input is an exercise sequence which is a sequence of question in one-hot vector. For the t-th question in $\mathbf{s}_i$, we denote its one-hot representation as $\mathbf{x}_{i,t}$, which is a $|\mathcal{Q}|$-dimensional vector with only the $\pi(i,t)$-th dimension as one and the other dimension as zeros. The model output is a sequence of student's hidden knowledge states. We denote student i's hidden knowledge states at time t as $\mathbf{h}_{i,t} \in \mathbb{R}^d$. To map the input of $\mathbf{x}_{i,t}$'s to the output of $\mathbf{h}_{i,t}$'s, we are inspired by [25] to take the question answering results into consideration. That is, we try to differentiate the question contribution to its corresponding knowledge hidden states, based on whether the question was answered correctly

TABLE I: A list of notations used in this work.

Var.	Description
$\mathcal{U}$	a set of students shown in the data
$\mathcal{Q}$	a set of questions appeared in the data
$\mathcal{C}$	a set of concepts included by all the questions
$\mathbf{s}_i$	an exercise sequence of student i
$\pi(i,t)$	the question student i answered at time t
$y_{i,j,t}$	binary value of whether student i solves question j at time t
$O_{j,k}$	binary value of whether question j has concept k
E_{k_1,k_2}	binary value of whether k_1 is prerequisite of k_2
$\mathbf{h}_{i,t}$	a d-dimensional vector as student i's embedding at time t
$\mathbf{x}_j$	a one-hot representation of question j
$\mathbf{c}_k$	a d-dimensional vector as concept k's embedding
$\hat{y}_{ij}$	a [0,1] value of how well student i solves question j
$m_{i,k,t}$	binary value of whether student i masters concept k at time t

or not. Denote $\oplus$ as a vector concatenation operator and $\mathbf{0}$ as a $|\mathcal{Q}|$-dimensional zero vector. Then, we introduce $\tilde{\mathbf{x}}_{i,t}$ as

$$\tilde{\mathbf{x}}_{i,t} = \begin{cases} \mathbf{x}_{i,t} \oplus \mathbf{0}, & \text{if } \ y_{i,\pi(i,t)} = 1; \\ \mathbf{0} \oplus \mathbf{x}_{i,t}, & \text{otherwise.} \end{cases} \tag{1}$$

Then, we use $\tilde{\mathbf{x}}_{i,t}$'s as the new input to GRU for estimating the student knowledge hidden states $\mathbf{h}_{i,t}$'s.

• *Reset gate*: at time t of $\mathbf{s}_i$, the reset gate is defined as

$$\mathbf{r}_{i,t} = \sigma(W_r \tilde{\mathbf{x}}_{i,t} + U_r \mathbf{h}_{i,t-1} + \mathbf{b}_r) \tag{2}$$

where $\sigma(\cdot)$ is a sigmoid function; $W_r \in \mathbb{R}^{d\times d}$, $U_r \in \mathbb{R}^{d\times d}$ and $\mathbf{b}_r \in \mathbb{R}^d$ are parameters. Besides, $\mathbf{h}_{i,0}$'s are also parameters, indicating each student i's initial knowledge states.

• *Update gate*: at time t of $\mathbf{s}_i$, the update gate is defined as

$$\mathbf{z}_{i,t} = \sigma(W_z \tilde{\mathbf{x}}_{i,t} + U_z \mathbf{h}_{i,t-1} + \mathbf{b}_z) \tag{3}$$

where $W_z \in \mathbb{R}^{d\times d}$, $U_z \in \mathbb{R}^{d\times d}$ and $\mathbf{b}_z \in \mathbb{R}^d$ are parameters.

• *Temporary hidden state*: at time t of $\mathbf{s}_i$, it is defined as

$$\tilde{\mathbf{h}}_{i,t} = tanh(W_h \tilde{\mathbf{x}}_{i,t} + U_h[\mathbf{r}_{i,t} \odot \mathbf{h}_{i,t-1}] + \mathbf{b}_h) \tag{4}$$

where $W_h \in \mathbb{R}^{d\times d}$, $U_h \in \mathbb{R}^{d\times d}$ and $\mathbf{b}_h \in \mathbb{R}^d$ are parameters; $\odot$ is an element-wise multiplication.

• *Output hidden state*: at time t of $\mathbf{s}_i$, it is defined as

$$\mathbf{h}_{i,t} = (1 - \mathbf{z}_{i,t}) \odot \mathbf{h}_{i,t-1} + \mathbf{z}_{i,t} \odot \tilde{\mathbf{h}}_{i,t}. \tag{5}$$

The set of parameters in GRU can be denoted as $\Theta_{GRU} = \{W_r, U_r, \mathbf{b}_r, W_z, U_z, \mathbf{b}_z, W_h, U_h, \mathbf{b}_h\}$. With this set of parameters, the GRU model can estimate one student's knowledge state according to her exercise sequence data.

Besides modeling students' exercise sequences, another important part is how to compute the level of mastery on a specific concept for a student according to her hidden knowledge states, which is described next.

Concept Mastery. We first define the probability that student i masters concept k at time t as

$$P(m_{i,k,t} = 1 | \mathbf{s}_i, \Theta_{GRU}, \mathbf{c}_k) = \sigma(\mathbf{h}_{i,t-1}^T \mathbf{c}_k + b_m), \tag{6}$$

where $\mathbf{h}_{i,t-1}$ denotes student's knowledge states at time $t-1$, $\mathbf{c}_k$ denotes concept k's embedding, and b_m denotes the bias.

In general, student i knows how to solve question j if she masters all the concepts in j. That is, student i must master every concept k with $O_{j,k} = 1$. Thus, we derive the probability that student i knows how to solve question j as

$$\Delta_{i,j,t} = \textstyle\prod_{k:O_{j,k}=1} P(m_{i,k,t} = 1 | \mathbf{s}_i, \Theta_{GRU}, \mathbf{c}_k). \tag{7}$$

Denote $\Theta = \{\mathbf{c}_k | k \in \mathcal{C}\} \cup \Theta_{GRU}$ as the overall set of parameters. Then, we can derive the probability of observing a correct answer from student i to a question j as

$$P(y_{i,j,t} = 1 | \mathbf{s}_i, \Theta) = \Delta_{i,j,t}. \tag{8}$$

V. Modeling Prerequisites as Constraints

Given a concept k_1 is another concept k_2's prerequisite, we can define two requirements on one student's mastery of concepts as following:

R_1: if student i has mastered k_2 at time t_2, then we can easily see she also masters k_1 thereafter. We can formalize this requirement as $(m_{i,k_2,t_2} = 1) \Rightarrow (m_{i,k_1,t_1} = 1)$, where $t_1 \geq t_2$ and "$\Rightarrow$" is an operator of *implication* similarly in first-order logic.

R_2: if student i does not master k_1 at time t_1, then most likely she does not master k_2 either. We can formalize this requirement as $(m_{i,k_1,t_1} = 0) \Rightarrow (m_{i,k_2,t_2} = 0)$, where $t_2 \geq t_1$.

Our intuition is to model prerequisite relations as constraint based on R_1 and R_2, but it is actually not a trivial task. There are several existing choices to model prerequisite relations as constraints, but none of them seems feasible.

- *Modeled as logic*. Both MLN [23] and PSL [1] require formulating the whole problem as logic reasoning, which is not easy for the task of knowledge tracing with complex student learning behaviors and psychological factors.
- *Modeled as boolean functions*. Both CODL [6] and PR [11] can be adapted to model a prerequisite constraint as a boolean function indicating whether R_1 and R_2 are satisfied. For example, given a threshold ϵ, we let $m_{i,k,t} = 1$ if $P(m_{i,k,t} = 1) \geq \epsilon$. Based on the values of m_{i,k_1,t_1} and m_{i,k_2,t_2}, the boolean function returns one if R_1 and R_2 are satisfied, or zero otherwise. The challenging part here is to choose the threshold ϵ; moreover, it is not clear whether one single threshold ϵ works for all different $m_{i,k,t}$'s.
- *Modeled as conditional probability*. CPR [32] can be adapted to model a prerequisite as a conditional probability to maximize. For example, we can formulate R_1 as $P((m_{i,k_2,t_2} = 1) \Rightarrow (m_{i,k_1,t_1} = 1)) = \frac{P(m_{i,k_1,t_1}=1, m_{i,k_2,t_2}=1)}{P(m_{i,k_2,t_2}=1)}$. In Eq. 6, we have defined $P(m_{i,k_2,t_2} = 1)$. But it is not trivial to define $P(m_{i,k_1,t_1} = 1, m_{i,k_2,t_2} = 1)$, such that: 1) it depends on $\{\mathbf{h}_{i,t}, \mathbf{c}_{k_1}, \mathbf{c}_{k_2}\}$; 2) it meets a probability definition

(*e.g.*, smaller than $P(m_{i,k_2,t'}=1)$, sum as one, and so on). Similar challenge also exists in formulating R_2 as a conditional probability.

Modeling Prerequisite as an Ordering Pair. With above analysis, we propose to consider a soft version of R_1 and R_2.

$\tilde{R}_1$. If $P(m_{i,k_2,t_2}=1)$ is large, then $P(m_{i,k_1,t_1}=1)$ is larger for $t_1 \geq t_2$, when both $y_{i,\pi(i,t_1)t_1}$ and $y_{i,\pi(i,t_2),t_2}$ are 1.

$\tilde{R}_2$. If $P(m_{i,k_1,t_1}=1)$ is small (i.e., $P(m_{i,k_1,t_1}=0)$ is large), then $P(m_{i,k_2,t_2}=1)$ is smaller for $t_2 \geq t_1$, when both $y_{i,\pi(i,t_1),t_1}$ and $y_{i,\pi(i,t_2),t_2}$ are 0.

We propose an ordering pair to formulate $\tilde{R}_1$ and $\tilde{R}_2$ together as:

$$P(m_{i,k_2,t_2}=1) \leq P(m_{i,k_1,t_1}=1), \quad \forall y_{i,\pi(i,t_1),t_1} = y_{i,\pi(i,t_2),t_2} \tag{9}$$

Let us analyze the advantages of the above ordering pair formulation: 1) **Mathematically**, this ordering pair can satisfy both $\tilde{R}_1$ and $\tilde{R}_2$ at the same time. If $P(m_{i,k_2,t_2}=1)$ is large, then according to Eq. 9, $P(m_{i,k_1,t_1}=1)$ is even larger. If $P(m_{i,k_1,t_1}=1)$ is small, then according to Eq. 9, $P(m_{i,k_2,t_2}=1)$ is even smaller. 2) **Semantically**, this ordering pair naturally encodes the fact that more advanced an concept is, more difficult it is to learn. That is, since k_2 (*e.g.*, addition within 1000) is a more advanced concept than k_1 (*e.g.*, addition within 10), the chance a student can master k_2 is lower than that she can master k_1. But if the student does master k_2, she must have mastered k_1. 3) **Computationally**, this ordering pair makes prerequisite modeling tractable for optimization. The constraints of R_1 and R_2 are hard to optimize due to the discrete assignments of $m_{i,k,t}$'s. Recall that, each $P(m_{i,k,t}=1)$ is parameterized with $\mathbf{h}_{i,t}$, $\mathbf{c}_k$ and b_m according to Eq. 6. Therefore, to enforce the ordering pair in Eq. 9, we can either use it as a hard constraint or a regularization term for maximizing $P(y_{i,j,t}=1)$ in Eq. 8. In either way, we solve a continuous optimization problem w.r.t. $\mathbf{h}_{i,t}$, $\mathbf{c}_k$ and b_m, which is much easier than directly optimizing with R_1 and R_2.

Based on this ordering pair formulation, we propose our new model named **Prerequisite-driven Deep Knowledge Tracing with Constraint modeling (PDKT-C)**, as illustrated by Figure 2. Its objective function can be defined as

$$\max_{\Theta} \log \prod_i \prod_t P(y_{i,\pi(i,t),t}|\mathbf{s}_i,\Theta) \tag{10}$$
$$s.t., P(m_{i,k_2,t_2}=1) \leq P(m_{i,k_1,t_1}=1), \quad \forall (k_1,k_2) \in E \;\; \& \;\; y_{i,\pi(i,t_1),t_1} = y_{i,\pi(i,t_2),t_2}$$

Note that $P(y_{i,\pi(i,t),t}=0) = 1 - P(y_{i,\pi(i,t),t}=1)$, and we shall maximize the probability $P(y_{i,\pi(i,t),t})$ according to the value of $y_{i,\pi(i,t),t}$. We relax the above hard constraint as soft

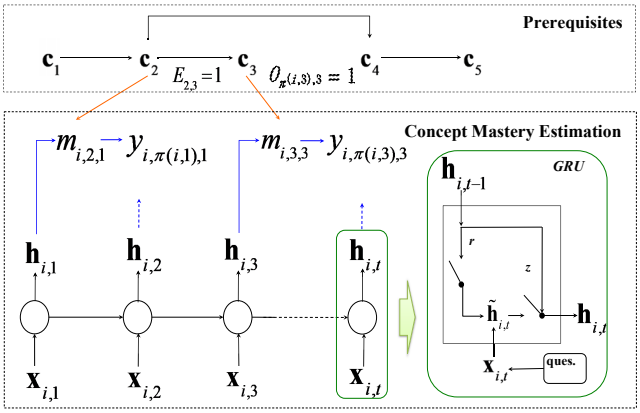


Fig. 2: Illustration of PDKT-C model.

regularization, and also introduce logarithm operator to ensure the scale consistency with the loss term:

$$\max_{\Theta} \sum_i \sum_t \log P(y_{i,\pi(i,t),t}|\mathbf{s}_i,\Theta) + \lambda \sum_i \sum_{k_1,k_2} \sum_{t_1} \sum_{t_2} \delta(*) \left[\log P(m_{i,k_1,t_1}) - \log P(m_{i,k_2,t_2})\right], \tag{11}$$

where k_1, k_2 are concept pairs having $E_{k_1,k_2}=1$; $\delta(*) = \delta(y_{i,\pi(i,t_1),t_1} = y_{i,\pi(i,t_2),t_2})$, and $\delta(*)=1$ if $y_{i,\pi(i,t_1),t_1}$ has same value with $y_{i,\pi(i,t_2),t_2}$, $\delta(*)=0$ otherwise; $\lambda > 0$ is a trade-off parameter. During model training and parameter learning, we use stochastic gradient descent to optimize Θ.

Discussions. One prerequisite may be stronger than another prerequisite. How shall we enforce such an intensity difference in our model design? Recall that in Eq. 10, we use a trade-off parameter λ to regularize the ordering pair constraint in Eq. 9. The larger λ is, the stronger the ordering pair constraint is, because $P(m_{ik_1}=1)$ has to be even larger than $P(m_{ik_2}=1)$. In other words, if student i does not master concept k_1 (*i.e.*, $P(m_{ik_1}=1)$ is small), then she is unlikely to master concept k_2 (*i.e.*, $P(m_{ik_2}=1)$ should be much smaller). Therefore, this λ value effectively encodes the prerequisite intensity. Ideally we want to assign different λ values for different prerequisites. There are several straightforward approaches for this λ value assignment: 1) provided by domain experts, which is labor intensive; 2) fined tuned according to the model evaluation w.r.t. Eq. 10, which is computationally expensive for such a combinatorial search. A possible idea is to take a data-driven approach. For example, we may use the association rule mining to identify *confidence* and *support* for each prerequisite, and use these scores as the priors to guide the learning of λ's for different prerequisites.

To accurately estimate a student's knowledge states, most knowledge tracing problems essentially make an implicit assumption that the sequence of exercises made by a learner usually occurs in a short period. Accordingly, the student's knowledge states would not be influenced by other factors during the interval of exercises.

TABLE II: Data Statistics

Statistics	ASSIST	AICFE-CM	AICFE-GE	AICFE-MA	AICFE-PH
# of records	88,123	125,726	44,662	66,568	67,874
# of students	1610	2648	1451	2445	964
Average records/student	54.7	47.5	30.8	27.2	70.4
# of questions	683	655	815	617	483
Question coverage/student	8%	7%	4%	4%	15%
# of concepts	45	21	23	25	19
# of relations	50	27	38	27	13

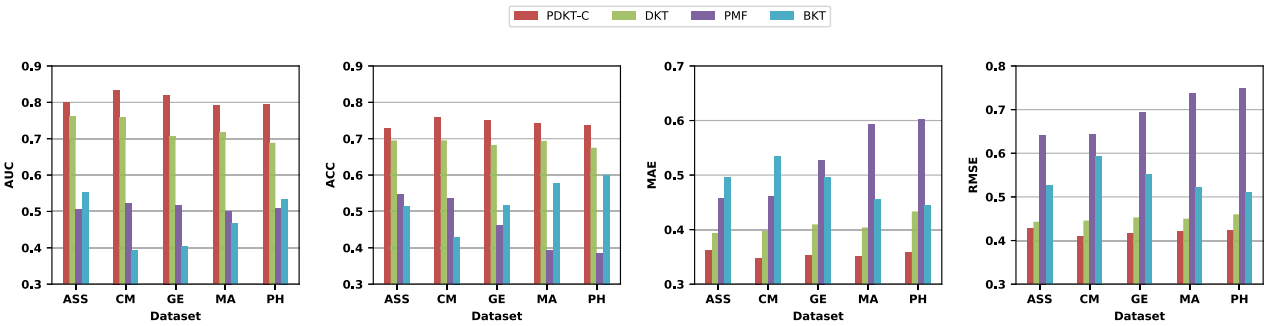
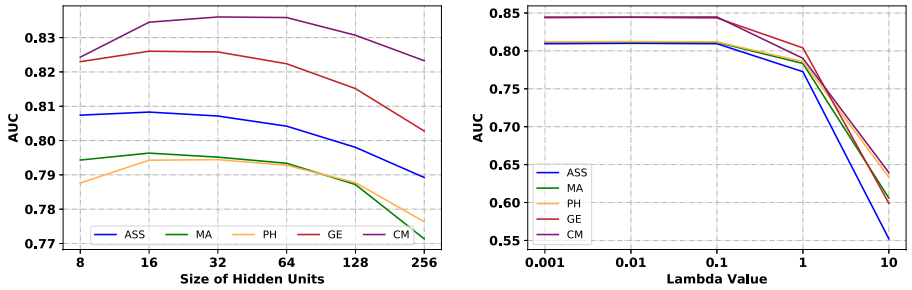


Fig. 3: Results compared to baselines.



(a) PDKT-C: # of Hidden Units.

(b) PDKT-C: λ.

Fig. 4: Impact of Parameters.

VI. Experiments

In this section, we first introduce the datasets and experimental settings. Then we evaluate our models by comparing with other baseline models. Subsequently, we conduct experiments to elaborate how important parameters affect the final performance.

A. Experimental Setup

Datasets. In this paper, we conduct our experiments with five real world datasets: ASSIST, AICFE-CM, AICFE-GE, AICFE-MA, AICFE-PH. ASSIST represents the public dataset ASSISTment 2009-2010 [1] "Non-skill builder" [10], which has been widely used by a lot of knowledge tracing work. In this work, we use the "Non-skill builder" dataset as it can better represent students' normal exercise sequences. Certain operations are applied to preprocess this dataset. Firstly, we use sequences generated with "random order" rather than the "linear order" because "random order" sequences are more natural and can avoid possible bias caused by order. Secondly, only sequences longer than 30 are used to avoid short sequences. Thirdly, questions of each sequences are further filtered by a set of concepts which is a subset selected by two experts from the whole set of concepts in ASSISTment data. These two experts manually define the prerequisite relations between the chosen concepts with the Kappa value as 0.83. Detailed data statistics of ASSIST are shown in Table II.

All the other four AICFE-* datasets represent the data collected from our online educational platform - Smart Learner Platform (SLP), which is developed by Advanced Innovation Center for Future Education (AICFE) at Beijing Normal University. This platform serves more than 4000 students from

[1] https://sites.google.com/site/assistmentsdata/home/assistment-2009-2010-data

31 local schools in Beijing. More specifically, AICFE-CM, AICFE-GE, AICFE-MA and AICFE-PH are four datasets for chemistry, geography, mathematics and physics respectively, and they are all online available for the research purpose[2]. All the questions on SLP are carefully labeled with the corresponding concepts by experienced teachers in corresponding subject. The data collection process lasts around two years since 2016. However SLP does not provides the prerequisite relation data, which is done manually by experts. For each subject, we find two experts to label the prerequisite relations between concepts. The Kappa values are 0.85, 0.90, 0.93, 0.87 for chemistry, geography, mathematics, physics respectively. The detailed data statistics of these datasets are shown in Table II.

In Table II, the "Average records/student" row shows the average number of question each student answers, which is between 27 and 71 for all the five dataset. Another row "Question coverage/student" describes the average percentage of questions each student answers, which is defined as "Average records/students" divides "# of questions". As shown in the table, this values varies from 4% to 14%. This low coverage percentage reflects the data sparseness issue on knowledge tracing.

Setting. In each experiment, we randomly select X% of sequences from the entire dataset as training data. We then randomly select 10% of sequences from the rest data as validation data. The remaining (100-10-X)% of sequences is used as testing data. Each experiment is repeated 10 times to calculate the average performance.

We evaluate PDKT-C from both classification and regression perspectives. On one hand, with considering it as a classification problem, exercise result is defined as a binary value, in which 0 represents incorrect answer as negative sample and 1 represents correct answer as positive sample. Hence, we use two popular classification metrics: Area Under ROC Curve (AUC) and Predication Accuracy (ACC) to measure our model performance. On the other hand, with considering it as a regression problem, questions result is defined as a continuous value between 0 and 1. Hence, we use two popular regression metrics: Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) to evaluate our model performance.

For our model PDKT-C, the number of hidden units is set to 32 for GRU network. In addition, for PDKT-C, the λ is set as 0.01. Furthermore, for the Adam algorithm utilized during model training, the initial learning rate and iteration number are set to 0.01 and 500 respectively with parameters $\beta 1 = 0.9$, $\beta 2 = 0.999$ and $\epsilon = 1 \times 10^{-8}$. All models are implemented with Tensorflow using Python on a Linux server that equips with 128GB RAM, two Intel Xeon Processor E5-2683 v3 CPUs and one GeForce GTX TITAN X GPU.

Baselines. To demonstrate the effectiveness of PDKT-C, we compare with other models solving knowledge tracing problem. More specifically, three models are selected as baselines: Bayesian Knowledge Tracing (BKT) [2], Probabilistic Matrix Factorization (PMF) [26] and Deep Knowledge Tracing (DKT) [22].

BKT: BKT utilizes a binary variable to indicate students' mastery of a concept. Based on the exercise sequences on a specific concept, BKT uses HMM to update the probabilities of this binary variable. BKT model inherently assumes that mastered knowledge will not be forgotten, but factors such as guessing and slipping are considered [2]. BKT is a classic knowledge tracing model and used by a lot of knowledge tracing work, so it is chosen as a baseline for our model evaluation.

PMF: PMF is originally designed for recommender system, but is adopted to estimate students' knowledge states recently [26]. PMF transforms the question answering problem into a similarity comparison problem, namely how well a student can answer a specific question depends on how similar her knowledge states and the question are. For this purpose, PMF projects both students' knowledge states and questions into vectors of the same space, and then calculate the similarities accordingly. Our PDKT-C has similar computation, so we also select PMF as one of our baseline models.

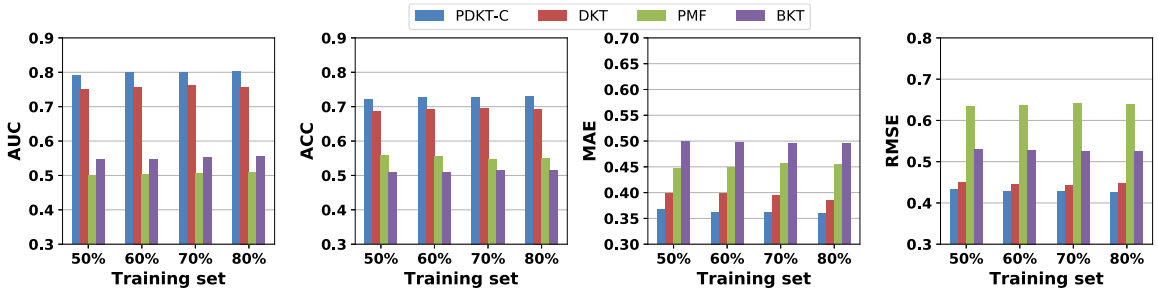
DKT: DKT applies the recurrent neural network model on exercise sequences to estimate students' mastery of concepts. These exercise sequences are a mixture of questions linking to different concepts, so DKT can estimate students' knowledge states of multiple concepts simultaneously. More specifically, DKT takes the one-hot vector of each question's concept as input, and outputs a vector between 0 and 1 that represents students' mastery level on all concepts [22]. DKT has become an important knowledge tracing model, so we also select it as a main baseline model. In this word, to be consistent with our model, we develop the DKT model with GRU network instead of the LSTM network.

B. Model Evaluation

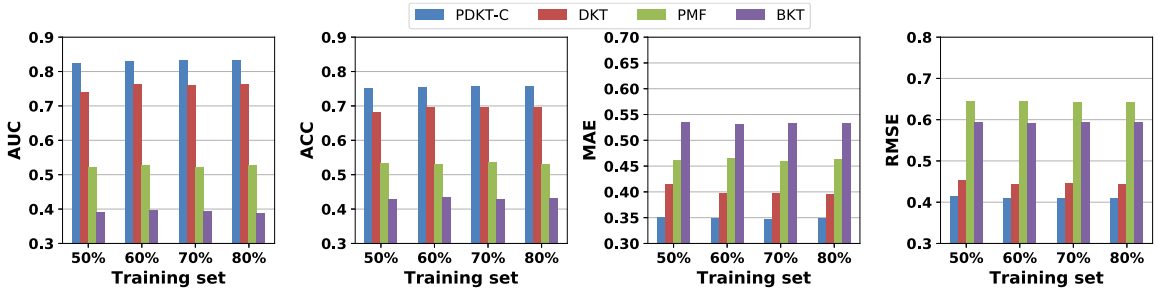
Comparison with Baselines. we first compare the overall performance of PDKT-C with baseline models. In this experiment, for each dataset, 70% of the data is randomly selected as training data, 10% is randomly selected as validation data, and the remaining 20% is used as testing data. Results of the five datasets on the four performance metrics are presented in Figure 3.

From Figure 3, we can have two main observations. Firstly, compared to baseline models, we can see PDKT-C obtains higher AUC and ACC values and lower MAE and RMSE values, which means a better performance. More specifically, compared to DKT, PDKT-C obtains 10% improvement on average. This proves that prerequisite relations do provide additional useful information for estimating students' knowledge states. Secondly, compared to PMF and BKT, PDKT-C and DKT obtain more consistent performances on all five datasets, which implies that deep learning based models are more robust.

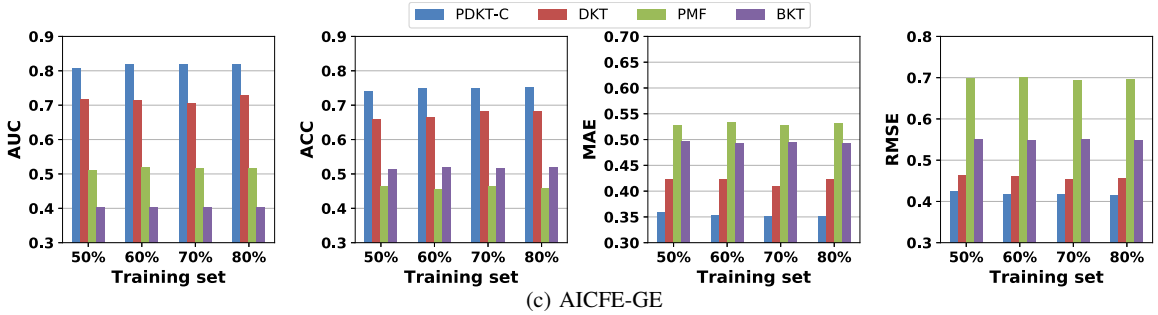
[2]http://www.bnu-ai.cn/download-unit



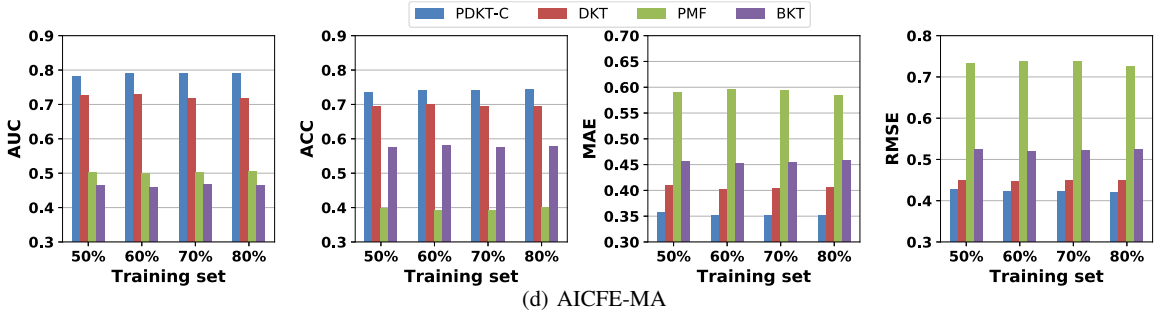
(a) ASSIST



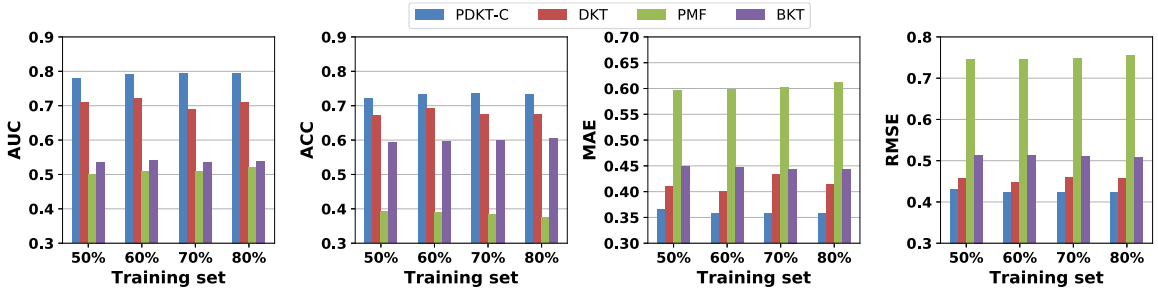
(b) AICFE-CM



(c) AICFE-GE

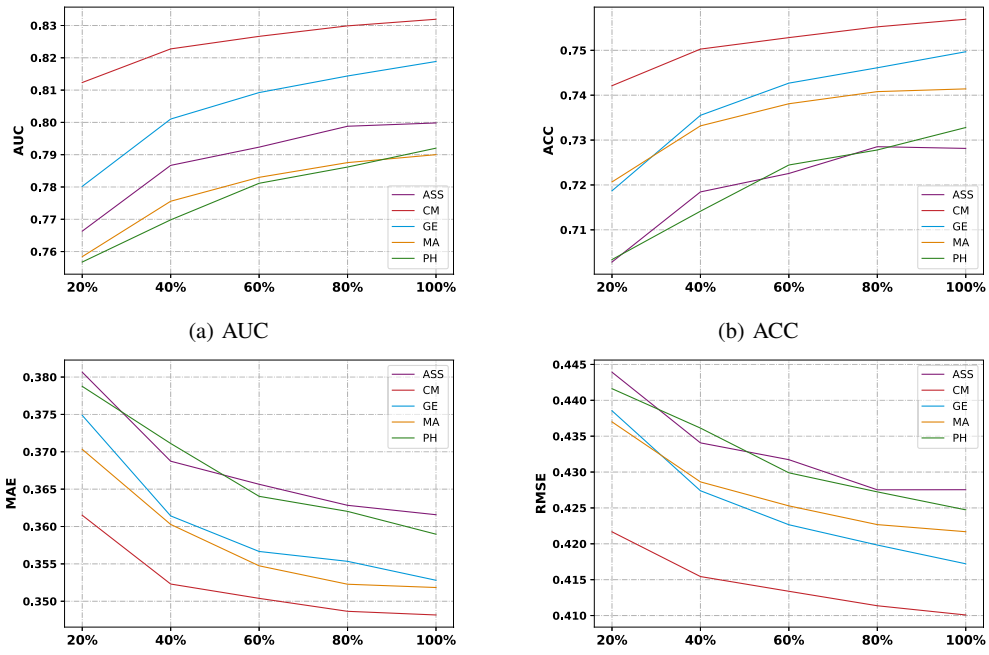


(d) AICFE-MA



(e) AICFE-PH

Fig. 5: Impact of training data size.



(a) AUC (b) ACC

(c) MAE (d) RMSE

Fig. 6: Impact of prerequisite number.

C. Impact of Parameters

Impact of trade-off parameters. For PDKT-C, there are two important parameters affecting the final performance. One is the # of hidden units of GRU network, and the other is the λ in Eq. 11. In this subsection, we examine how different values of these two parameters influence the results.

# of Hidden Units: Figure 4a illustrates how the performance varies with the number of hidden units of GRU network for PDKT-C . From Figure 4a, we can find that PDKT-C has better results when the number of hidden units is set to 16 or 32. Hence, we set it as 32 in our experiments.

Parameter λ: Figure 4b illustrates how the performance of PDKT-C varies with different values of λ. The results show that PDKT-C achieves similar performance when λ is 0.001 or 0.01 or 0.1. By checking the exact value, we find that PDKT-C performs slightly better when λ equals 0.01, so λ is set as 0.01.

Impact of training data size. The size of training data usually has a great influence on model performance. In this subsection, we elaborate how performance changes w.r.t. different numbers of training samples. Experiments are conducted with different size of training data, namely 50%, 60%, 70%, 80%. Experimental results of the five datasets on the four performance metrics are shown in Figure 5.

From the results, we can get two main observations. Firstly, for all datasets, we can see that PDKT-C outperforms baseline models on all sizes of training data. Secondly, the performance difference between different training sizes are relatively small. Even only 50% of the data is used for training, PDKT-C can obtain good performance.

Impact of prerequisite number. Besides the the size of training data, the number of prerequisites is another influential factor. In this subsection, we further check the performance change w.r.t. different number of prerequisites in each dataset. More specifically, we conduct experiments with 20%, 40%, 60%, 80%, 100% of all prerequisites.

Figure 6 shows how the performance of different metrics vary with the number of prerequisites used. With all the four metrics, we can find that the performance of PDKT-C improves with more prerequisites used, which verifies the effectiveness of prerequisites in estimating students' knowledge states.

VII. Conclusion and Future Work

In this paper, we developed a new knowledge tracing model PDKT-C to better estimate students' knowledge states. The intuition is to utilize prerequisite relations between concepts to regularize knowledge tracing model. More specifically, by considering how prerequisite relations affect students' mastering concepts, we defined ordering pairs with the probability of concept mastery. With a proper mathematical formulation, this ordering pair formed mathematical constraints in knowledge tracing model to better predict students' knowledge states. Finally, extensive experiments are conducted on five real world datasets and the results show that the proposed PDKT-C model generates a significant performance improvement comparing to other baseline models.

For the future work, we would like to explore other ways to better express those concept interdependencies, for example, graph embedding. In addition, we also would like to jointly optimize the prerequisite relation extraction process and knowledge tracing process to better identify prerequisite relations and estimate students' knowledge states.

ACKNOWLEDGMENT

This research is partially supported by the National Natural Science Foundation of China (No. 61702039), the Humanities and Social Sciences Foundation of the Ministry of Education of China (No. 17YJCZH116) and the Fundamental Research Funds for the Central Universities.

REFERENCES

[1] Stephen H. Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. Hinge-loss markov random fields and probabilistic soft logic. arXiv:1505.04406 [cs.LG], 2015.

[2] Ryan SJD Baker, Albert T Corbett, and Vincent Aleven. More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing. In *International Conference on Intelligent Tutoring Systems*, pages 406–415. Springer, 2008.

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

[4] John R Bergan and Patrick Jeska. 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, 1980.

[5] Cristina Carmona, Eva Millán, José-Luis Pérez-de-la-Cruz, Mónica Trella, and Ricardo Conejo. Introducing prerequisite relations in a multi-layered bayesian student model. In *10th International Conference of User Modeling*, pages 347–356, 2005.

[6] Ming-Wei Chang, Lev Ratinov, and Dan Roth. Structured learning with constrained conditional models. *Mach. Learn.*, 88(3):399–431, September 2012.

[7] Yetian Chen, José P. González-Brenes, and Jin Tian. Joint discovery of skill prerequisite graphs and student models. In *9th International Conference on Educational Data Mining*, pages 46–53, 2016.

[8] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[9] Albert T Corbett and John R Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.

[10] Mingyu Feng, Neil Heffernan, and Kenneth Koedinger. Addressing the assessment challenge with an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction*, 19(3):243–266, 2009.

[11] Kuzman Ganchev, João Graça, Jennifer Gillenwater, and Ben Taskar. Posterior regularization for structured latent variable models. *Journal of Mach. Learn. Research*, 11:2001–2049, aug 2010.

[12] Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. Knowledge graph embedding with iterative guidance from soft rules. In *AAAI*, 2018.

[13] Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard H. Hovy, and Eric P. Xing. Harnessing deep neural networks with logic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*, 2016.

[14] Zhenya Huang, Qi Liu, Enhong Chen, Hongke Zhao, Mingyong Gao, Si Wei, Yu Su, and Guoping Hu. Question difficulty prediction for reading problems in standard tests. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[15] Mohammad Khajah, Rowan Wing, Robert Lindsey, and Michael Mozer. Integrating latent-factor and knowledge-tracing models to predict individual differences in learning. In *Educational Data Mining 2014*, 2014.

[16] Chen Liang, Zhaohui Wu, Wenyi Huang, and C. Lee Giles. Measuring prerequisite relations among concepts. In *EMNLP*, pages 1668–1674, 2015.

[17] Chen Liang, Jianbo Ye, Zhaohui Wu, Bart Pursel, and C. Lee Giles. Recovering concept prerequisite relations from university course dependencies. In *AAAI*, pages 4786–4791, 2017.

[18] Gideon S. Mann and Andrew McCallum. Generalized expectation criteria for semi-supervised learning with weakly labeled data. *Journal of Mach. Learn. Research*, 11:955–984, 2010.

[19] Shike Mei, Jun Zhu, and Jerry Zhu. Robust regbayes: Selectively incorporating first-order logic domain knowledge into bayesian models. In *Proc. of ICML*, 2014.

[20] Liangming Pan, Chengjiang Li, Juanzi Li, and Jie Tang. Prerequisite relation learning for concepts in moocs. In *ACL*, pages 1447–1456, 2017.

[21] Zachary A Pardos and Neil T Heffernan. Kt-idem: introducing item difficulty to the knowledge tracing model. In *International Conference on User Modeling, Adaptation, and Personalization*, pages 243–254. Springer, 2011.

[22] Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein. Deep knowledge tracing. In *Advances in Neural Information Processing Systems*, pages 505–513, 2015.

[23] Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, 2006.

[24] Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. Injecting Logical Background Knowledge into Embeddings for Relation Extraction. In *Conference of the North American Chapter of the Association for Computational Linguistics Human Language Technologies (NAACL HLT)*, 2015.

[25] Yu Su, Qingwen Liu, Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Chris Ding, Si Wei, and Guoping Hu. Exercise-enhanced sequential modeling for student performance prediction. In *The 32nd AAAI Conference on Artificial Intelligence*, 2018.

[26] Nguyen Thai-Nghe, Lucas Drumond, Tomáš Horváth, Artus Krohn-Grimberghe, Alexandros Nanopoulos, and Lars Schmidt-Thieme. Factorization techniques for predicting student performance. In *Educational recommender systems and technologies: Practices and challenges*, pages 129–153. IGI Global, 2012.

[27] Shuting Wang, Alexander Ororbia, Zhaohui Wu, Kyle Williams, Chen Liang, Bart Pursel, and C Lee Giles. Using prerequisites to extract concept maps from text books. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 317–326. ACM, 2016.

[28] Run-ze Wu, Qi Liu, Yuping Liu, Enhong Chen, Yu Su, Zhigang Chen, and Guoping Hu. Cognitive modelling for predicting examinee performance. In *IJCAI*, pages 1017–1024. AAAI Press, 2015.

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

[30] Michael V Yudelson, Kenneth R Koedinger, and Geoffrey J Gordon. Individualized bayesian knowledge tracing models. In *International Conference on Artificial Intelligence in Education*, pages 171–180. Springer, 2013.

[31] Jiani Zhang, Xingjian Shi, Irwin King, and Dit-Yan Yeung. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th International Conference on World Wide Web*, pages 765–774. International World Wide Web Conferences Steering Committee, 2017.

[32] Vincent W. Zheng and Kevin Chen-Chuan Chang. Regularizing structured classifier with conditional probabilistic constraints for semi-supervised learning. In *CIKM*, pages 1029–1038, 2016.