IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 14 June 2022, accepted 28 June 2022, date of publication 4 July 2022, date of current version 13 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3187987

RESEARCH ARTICLE

SPAKT: A Self-Supervised Pre-TrAining Method for Knowledge Tracing

YULING MA[1], PENG HAN[2], HUIYAN QIAO[1], CHAORAN CUI[3], YILONG YIN[2], AND DEHU YU[1]

[1]School of Computer Science and Technology, Shandong Jianzhu University, Jinan 250101, China
[2]School of Software, Shandong University, Jinan 250101, China
[3]School of Computer Science and Technology, Shandong University of Finance and Economics, Jinan 250014, China

Corresponding author: Chaoran Cui (crcui@sdufe.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 62077033 and Grant 62177031, in part by the Shandong Provincial Natural Science Foundation under Grant ZR2021MF044, in part by the Shandong Provincial Undergraduate Teaching Reform Research Project Z2021133, in part by the Industry-University Cooperation Collaborative Education Project 202102423045, and in part by the Shandong Provincial Education and Teaching Research Project 2021JXY012.

ABSTRACT Knowledge tracing (KT) is the core task of computer-aided education systems, and it aims at predicting whether a student can answer the next exercise (i.e., question) correctly based on his/her historical answer records. In recent years, deep neural network-based approaches have been widely developed in KT and achieved promising results. More recently, several researches further boost these KT models via exploiting plentiful relationships including exercise-skill relations (E-S), the exercise similarity (E-E) as well as skill similarity (S-S). However, these relationship information are frequently absent in many real-world educational applications, and it is a labor-intensive work for human experts to label it. Inspired by recent advances in natural language processing domain, we propose a novel pre-training approach, namely as SPAKT, and utilize self-supervised learning to pre-train exercise embedding representation without the need for expensive human-expert annotations in this paper. Contrary to existing pre-training methods that highly rely on manually labeling knowledge about the E-E, S-S, or E-S relationships, the core idea of the proposed SPAKT is to design three self-attention modules to model the E-S, E-E, and S-S relationships, respectively, and all of these three modules can be trained in the self-supervised setting. As a pre-training approach, our SPAKT can be effortlessly incorporated into existing deep neural network-based KT frameworks. We experimentally show that, even without using expensive annotations about the aforementioned three kinds of relationships, our model achieves competitive performance compared with state-of-the-arts. Our algorithm implementations have been made publicly available at https://github.com/Vinci-hp/pretrainKT.

INDEX TERMS Knowledge tracing, student performance prediction, self-supervised learning, bidirectional encoder representation from transformers (BERT).

I. INTRODUCTION

Recent decades have witnessed the rapid growth of the computer-aided education systems, e.g., intelligent tutoring systems (ITS), which aim at automatically and significantly boosting learning gains [1]. One of the key tasks in these systems is knowledge tracing, which can model and track students' evolving knowledge state according to their historical learning interactions [2]. To be specific, the task of knowledge tracing is to predict whether a student can answer the next exercise correctly based upon all previous answer records. With the results of knowledge tracing, the system can better understand students in the teaching process so as to recommend learning resources and optimize the teaching strategy.

Recently, more and more studies have paid attention to knowledge tracing. The two most popular categories are Bayesian-based knowledge tracing (e.g., BKT [3]) and deep neural network-based knowledge tracing (e.g., DKT [4]). Conventional BKT was presented by Corbett and Anderson in 1994 [3], which infers students' hidden knowledge states mainly from their performance (i.e., correct, incorrect) on

The associate editor coordinating the review of this manuscript and approving it for publication was Essam A. Rashed.

solving problems related to a given knowledge skill (concept) and continually updates its estimation on their learning state of that skill. Later on, many scholars extended this basic BKT framework by introducing other factors, such as students' individual differences [5], difficulty level of exercises [6], forgetting factors [7], guessing and slipping factors [8].

The recent success of deep neural networks has boosted research on knowledge tracing. The deep knowledge tracing (DKT) model [4] was the first method to explore KT using long short-term memory (LSTM) network. And then more deep models were exploited in the KT task, e.g., CNN [9], GNN [10], [11], and MANN [12]. In the majority of deep neural network-based KT models, the exercise a student answered is generally converted into a fixed length input vector so as to train the deep model, e.g. LSTM. One-hot encoding is one of the most naive approaches to obtain such an input vector. However, such a simple representation is considerably hard to make use of the plentiful side information, e.g., relationships between exercises and skills (E-S), different exercises (E-E), as well as skills (S-S). Aiming at exploring these plentiful relationships, some pre-training approaches have been developed to obtain better embedding representation for exercises, such as PEBG [13] and Rasch [14]. Liu *et al.* [13] and Minn *et al.* [15] exploited exercise-skill relation so as to learn a low-dimensional embedding for each exercise. Nakagawa *et al.* [11] as well as Tong *et al.* [16] introduced the relationship among skills into KT model via conceptualizing it as the potential graph structure. Although these researches can significantly outperform the performance of existing deep KT models, one obvious flaw is that they heavily relied on the prior knowledge (i.e., E-S and S-S relationships) annotated by human experts, which is time-consuming and laborious. In these circumstances, a pre-training approach for knowledge tracing is desirable, which can automatically explore and exploit the plentiful relationships i.e., E-E, S-S as well as S-S relationships.

To this end, in this paper, we introduced self-supervised learning paradigm into knowledge tracing and proposed a novel pre-training approach for obtaining exercise embedding, namely as SPAKT (Self-supervised Pre-trAining method for Knowledge Tracing). Bert networks [17] can be trained in a self-supervised learning style through using a ''masked language model,'' which randomly masks some of the tokens from the input sequence, and the task is to predict the original ID of the masked one based only on its context. Inspired by it, in our SPAKT framework, three Bert networks have been employed to learn the relationships of E-S, E-E, and S-S, respectively. To be specific, it first utilizes duel path Bert networks in the bottom layer, one for modeling E-E relation, and the other for learning S-S. The outputs of dual path Berts are combined through a fusion layer, and then fed into another Bert network to extract E-S relation as well as generating the exercise embedding representation simultaneously. It should be mentioned that the whole training progress for our SPAKT only utilized the self-supervised information, i.e., the masked tokens in dual path Berts of bottom layer, and the difficulty level of exercises in the self-attention module of top layer. The contributions of this article are as follows:

- To the best of our knowledge, we first introduced self-supervised learning into the KT tasks and proposed a novel pre-training approach to obtain exercise embeddings based on Bert networks, namely as ''SPAKT.''
- Contrary to existing pre-training methods that highly rely on the prior knowledge about the relation of E-E, S-S, or E-S, our SPAKT employs the masked mechanism and self-supervised information (i.e., exercise difficulty level) extracted from the original dataset, which allows us to pre-train it even without any prior knowledge about the relationships.
- The proposed SPAKT significantly outperforms state-of-the-art baselines on three common KT datasets, which indicates that our method can learn a more effective embedding representation. For the sake of reproducibility, our implementation is available at https://github.com/Vinci-hp/pretrainKT.

The rest of this article is organized as follows. In Section (II), we introduce the related work. Section (III) details the whole framework of our SPAKT. In Section (IV), we show the experimental results, which are followed by conclusions in Section (V).

II. RELATED WORK

Recent years, it emerged a large body of fruitful research work in knowledge tracing [18], [19]. Ramirez *et al.* [20] surveyed machine learning techniques commonly used in KT, including Bayesian Knowledge Tracing (BKT), Deep Knowledge Tracing (DKT), Long Short-Term Memory (LSTM), Bayesian Networks (BN), Support Vector Machines (SVM), Dynamic Key Value Memory Network (DKVMN), Performance Factor Analysis (PFA). Inspired by this, we propose a discussion focusing on machine learning techniques for knowledge tracing and roughly divide existing KT approaches into four categories: Bayesian-based KT models, logistic regression-based KT models, factorization machine-based KT models, and deep neural network-based models.

A. BAYESIAN-BASED KT MODELS

Bayesian approaches are the most originally and widely investigated KT approaches, including Bayesian Knowledge Tracing (BKT) [3] and Dynamic Bayesian Knowledge Tracing (DBKT) [21].

1) BAYESIAN KNOWLEDGE TRACING AND ITS EXTENSIONS

As early as 1994, Corbett and Anderson [3] proposed Bayesian knowledge tracing model, which was the first KT model to be proposed. The basic BKT model exploits four student-related probability factors, including student's initial knowledge level, the acquisition probability (i.e., a knowledge concept will make the transition from the unlearned to

the learned state), the probability of guess as well as slip. And then it traces students' changing knowledge states based on their performance history through using Hidden Markov Model (HMM). After that, many scholars extended this basic BKT framework by introducing other factors. To name a few, Pardos and Heffernan [5] improved the predictive performance over the basic BKT through individualizing students' initial knowledge parameter. Baker *et al.* [8] utilized machine learning to make dynamically estimations of the probability that a response was a guess or a slip, and then employed these dynamic performance estimates to enhance the basic BKT. Pardos and Heffernan [6] improved the performance of BKT by incorporating item difficulty. More recently, novel approaches have been proposed to handle this issue that a question might require several knowledge skills, such as feature-aware student tracing (FAST) [22] and fuzzy Bayesian knowledge tracing (FBKT) [23].

2) DYNAMIC BAYESIAN KNOWLEDGE TRACING

Considering there are implicit relationships among different knowledge skills, Kaser *et al.* [21] utilized dynamic Bayesian networks to jointly learn the parameters for different skills within a single model and proposed dynamic Bayesian knowledge tracing (DBKT) model. It represents a student's knowledge mastery as binary latent variables, and then utilizes dynamic Bayesian networks to learn the prerequisite hierarchies and relationships within knowledge skills. Compared with the aforementioned Bayesian knowledge tracing models, DBKT can potentially improve the representational power of knowledge tracing.

3) DISCUSSION

Bayesian-based KT models have both advantages and limitations. Bayesian method is a simple and effective modeling method [3]. Moreover, Bayesian-based KT models learn students' evolving knowledge state through employing several parameters with explicit meaning and function, e.g., the probability of guess and slip, which brings better interpretability to the output results of the model. However, one of its limitations is obvious that the performance of Bayesian-based KT models highly relies on the carefully handcrafted parameters by human experts, and thus it is considerably hard to fully leverage plentful side information, e.g., relationships of E-E, S-S and E-S.

B. LOGISTIC REGRESSION-BASED KT MODELS

Logistic models are a kind of models based upon logistic regression function. The two representative KT models are learning factor analysis (LFA) [24] and performance factor analysis (PFA) [25].

1) THE LFA MODEL

LFA [24] utilized three learning factors as features, including the initial knowledge state of each student, the easiness as well as learning rate of different skills, which were manually extracted from students' learning interactions. And then it used logistic regression to predict the probability of a correct answer. However, LFA does not consider different effects of getting a practice opportunity correct or incorrect, i.e., whether an answer is correct or incorrect, it is processed equally by LFA.

2) THE PFA MODEL

PFA [25] considered different effects of getting a practice opportunity correct or incorrect, and employed the easiness of different skills, the prior successes for a skill of a student as well as the prior failures as features to construct the logistic regression model. However, when PFA counts the numbers of prior exercises answered correctly/incorrectly, it does not consider data aging (i.e., the order of those exercises answered). To address this issue, Gong *et al.* [26] proposed a variant of PFA, which can reflect the time (order) of an exercise answered through a decay factor. The decay factor can lead an evolving counts about the prior successes/failures for a skill by decreasing the importance of prior performances.

3) DISCUSSION

The model fitting procedure of logistic regression can be ensured to reach global maxima, and thus we can obtain unique best fitting parameters [26]. Moreover, Mandalapu *et al.* [27] reported that through using very carefully designed features, logistic regression-based models can achieve superior performance than complex models, e.g., deep neural networks. However, the performance of logistic models are generally highly dependent on handcrafted features, and it is considerably hard to obtain efficient features in a manual way, which limits the improvement of predictive performance for logistic regression-based KT models.

C. FACTORIZATION MACHINE-BASED KT MODELS

Through referring students and exercises as users and items, respectively, Thai-Nghe *et al.* [28] introduced the time factor about students' answering an exercise, and denoted a student sample as a 3-dimensional tensor. Then tensor factorization technology was utilized to model students' knowledge state. Vie and Hisashi [29] proposed knowledge tracing machines (KTM) model, which utilized factorization machine to estimate student knowledge. In KTM model, there are multiple handcrafted features, and each feature is denoted as a one-hot encoded vector. To be specific, assume there are three kinds of features, i.e., the feature about student, exercise, and its corresponding skills, then the total length of feature vector equals the sum of #students, #exercises, and #skills, where #students/exercises/skills means the number of students/exercises/skills. Gan *et al.* [30] first learned out the difficulty level of exercises as well as learning and forgetting factors, and introduced them into factorization machine for knowledge tracing. Chen *et al.* [31] utilized learning curve and forgetting curve in educational theory and proposed a novel factorization machine model for knowledge tracing. Gan *et al.* [32] incorporated learners' abilities, item difficulty, and learning and forgetting factors together and

utilized the factorization machine to trace the evolution of each student's knowledge acquisition. Liu *et al.* [33] combined student-related (e.g., learning and forgetting curve) and exercise-related priors (e.g., knowledge structure), and designed a probabilistic matrix factorization framework to track a student's knowledge mastery levels. Furthermore, Liu *et al.* [34] introduced the relationship between exercises owning the same knowledge concept and further improved the model performance.

1) DISCUSSION

Through encoding exercises-related or students-related information into the model, FMs allow us to make use of the side information available at hand [2]. However, there still exists plentiful side information under explored, e.g., relationships about S-S. Additionally, factorization machine-based KT models mapped students' knowledge level to an implicit space, and thus it is difficult to give a reasonable explanation for the prediction results.

D. DEEP NEURAL NETWORK-BASED KT MODELS

Inspired by recent advances in deep learning, there is a growing trend to model students' knowledge based on deep models. We group deep neural network-based KT models into two categories as follows:

1) DEEP KT MODELS WITHOUT PRE-TRAINING

In 2015, Piech *et al.* [4] first applied the recurrent neural network (RNN) to the task of knowledge tracing, and proposed the deep knowledge tracing model (i.e., DKT). Given a student's interaction sequence $\{(q_1, a_1), (q_2, a_2), \ldots, (q_t, a_t)\}$, in which q_t is the ID of the question answered by the student at the t time step, and a_t means whether the student answered the question correctly, DKT model first represents q_i and a_i $(1 \leq i \leq t)$ as a fixed length vector (e.g., one-hot code) and feeds it into deep neural networks, such as RNN and LSTM, then obtains the output prediction vector. Thanks to the strong representation learning ability and function fitting ability of the deep neural networks, the area under the curve (AUC, a commonly used evaluation metric) of the DKT model is nearly 25% higher than that of the traditional BKT model on the ASSISTments dataset [4]. Since then, deep neural network based approaches have become one of the most concerned KT modeling methods.

More and more deep neural networks were introduced into knowledge tracing. Convolutional neural networks (CNNs) were utilized to model the individualization of students in KT based on their continuous learning interactions [9]. Graph neural networks (GNN) were introduced into modeling student proficiency through reformulating the KT task as a time-series node-level classification problem [10], [11]. Memory-augmented neural networks (MANN) with one static matrix and one dynamic matrix were proposed for processing knowledge tracing tasks [12]. In terms of prediction performance, such deep models have achieved the state-of-the-art results on the majority of KT benchmark datasets.

2) DEEP KT MODELS WITH PRE-TRAINING

Although the aforementioned deep KT models achieved promising results, there is still a considerably large room to boost its performance, owing to the inherent complexity of human knowledge. As aforementioned, the exercise a student answered is generally converted into a fixed length input vector before being fed into the deep model, e.g. LSTM. Related research shows that the effectiveness of representation vectors for exercises can affect the performance of the deep KT models [13]. Aiming at obtaining embedding representation with high quality for exercises, Minn *et al.* [15] employed the explicit exercise-skill relation as a constraint to train exercise embedding so as to improve KT model performance. Liu *et al.* [13] proposed a pre-training approach to learn a low-dimensional embedding for each question based upon a bipartite exercise-skill relation graph where vertices were exercises and skills respectively. Nakagawa *et al.* [11] and Tong *et al.* [16] introduced the relationship among skills into knowledge tracing via conceptualizing it as the potential graph structure, in which the nodes represent the set of skills and the edges represent the relationships between these skills.

3) DISCUSSION

Benefiting from the strong representation learning ability and function fitting ability of the deep neural networks, Deep Neural Networks based KT models have achieved quite good performance. Moreover, they can be trained in an end-to-end learning style and thus do not require manually designed features. Despite the encouraging progress, the performance of deep KT models are far from satisfactory and the approaches for good interpretability is still in its infancy. Deep KT models with pre-training can obtain better embedding for exercises and further enhance the performance of deep KT models. However, one obvious flaw is that existing approaches heavily relied on the knowledge about E-S or S-S relationships annotated by human experts, which is time-consuming and laborious.

Due to space constraints, it is considerably hard for us to give a comprehensive and detailed introduction to all the related works and characterize the benefits and drawbacks for all related works one by one. Instead, we refer readers to several survey or review articles [2], [19], [20] about knowledge tracing.

III. PROPOSED METHOD

In this section, we first introduce the notations employed in this article, then further elaborate the essential component of our proposed SPAKT, i.e., single Transformer layer. Finally, we detail the overview framework of SPAKT.

A. NOTATIONS

We first introduce some notations that will be required in this paper. We use capital letters (e.g., X) and bold lowercase letters (e.g., $\mathbf{x}$) to denote sets and vectors, respectively. We employ non-bold lowercase letters (e.g., x) to represent

scalars. T represents transpose operator of a matrix. If not otherwise clarified, all vectors are in column form.

In this paper, we denote $Q = \{q_i | i = 1, \ldots, |Q|\}$ as the set of all distinct $|Q|$ exercises (i.e., questions), and q_i is the ID of the i^{th} exercise. Let $S = \{s_j | j = 1, \ldots, |S|\}$ to be the set of all distinct $|S|$ skills (concepts) and s_j is the ID of the j^{th} knowledge skill (concept). Given a student's historical question-answering responses $L = \{(\widetilde{q}_1, a_1), (\widetilde{q}_2, a_2), \ldots, (\widetilde{q}_t, a_t)\}$, where $\widetilde{q}_t \in Q$ is the ID of the question answered by the student at the t time step, and a_t means whether the student answered the question correctly. Concretely, $a_t = 1$ means answering correctly, and 0 otherwise. The goal of knowledge tracing is to predict the probability that the student will correctly answer the next exercise.

Contrary to existing methods that highly rely on the prior knowledge from domain experts, which results in highly human labor costs, we attempt to model these information, i.e., E-E, S-S as well as E-S relationships, through using self-supervised learning. In the following, we will first briefly introduce the basic component (i.e., Single Transformer Layer) of the proposed SPAKT, and then detail the whole framework of the pre-training network.

B. SINGLE TRANSFORMER LAYER

In this paper, as shown on the left in Figure (1), each single transformer layer contains a Multi-Head Self-Attention sub-layer, a Feed-Forward Network, two ADD & Norm units as well as Dropout. Multi-Head Self-Attention layer can benefit the model to pay attention to the information of different representation subspaces from different locations. Feed-Forward Network can make the model with nonlinear and interaction between different dimensions. ADD & Norm unit is used to solve the problem of vanishing gradient, and Dropout unit can enhance the generalization ability of the model. Considering the use of these components has become common and our SPAKT is almost identical to the original, we will omit the description about exhaustive working principles of these components. Instead, we refer readers to the article about transformer model [35].

C. SPAKT: SELF-SUPERVISED PRE-TRAINING METHOD FOR KT

Figure (1) visualizes the architecture of SPAKT, which is comprised of dual path self-attention modules (i.e., E-E Module and S-S Module) followed by another self-attention module (i.e., E-S Module). Each module is essentially a Bert network [17], which is composed of multiple transformer layers [35].

1) E-E MODULE

Intuitively, teachers or intelligent tutors generally supply students some exam exercises that meet their current knowledge state and knowledge mastery level, and there thus exist certain correlations between the exercises answered by a student in a period of time. Inspired by this, we utilized Bert network, and thus can borrow "masked mechanism" from NLP (natural language process) model [17] to learn E-E relationship.

Given an input exercise sequence, the masked mechanism randomly masks some of the exercises from the sequence, and the task is to predict the original ID of the masked exercise based on its context. Specifically, we denote q_i^0 as raw embedding vector (e.g., one-hot coding) of the i^{th} exercise in the student's interaction sequence and $\widetilde{q}_i$ is its original ID in exercise set Q. Given an exercise sequence $\{q_1^0, \ldots, q_i^0, \ldots, q_n^0\}$, we simply and randomly mask some percentage of the sequence, and then feed it into the E-E module. The module will generate the intermediate embedding representation of each exercise and output it as a new exercise sequence $\{q_1^1, \ldots, q_i^1, \ldots, q_n^1\}$. Then, we utilize the intermediate sequence to predict the original ID of those masked exercises based on its context information. Through the binary cross-entropy loss function, this process can be formulated as:

$$\mathcal{L}_1 = -\sum_i (\widetilde{q}_i \log \hat{q}_i), \tag{1}$$

where $\widetilde{q}_i$ is original ID of the masked exercise q_i, and $\hat{q}_i$ is the prediction result. Given the intermediate embedding sequence, we use a linear function followed by Softmax function to obtain $\hat{q}_i$, which can be formulated as:

$$\hat{q}_i = Softmax(\mathbf{w}_1^T \mathbf{q}_i^1 + b_1), \tag{2}$$

where q_i^1 is the intermediate embedding representation of the i^{th} exercise, w_1 and b_1 are parameters to be learned.

2) S-S MODULE

Similarly, we also use "masked mechanism" to model S-S relationships in the S-S module, which takes masked skill sequence as the input. Given the set of all distinct $|S|$ skills $S = \{s_j | j = 1, \ldots, |S|\}$, we randomly select some skills from it and generate a m-skill sequence $\{\widetilde{s}_1, \widetilde{s}_2, \ldots, \widetilde{s}_m\}$. $\widetilde{s}_j (j = 1, \ldots, m) \in S$ is the original ID of the selected j^{th} skill. We represent $\widetilde{s}_j$ as a fixed length input vector s_j so as to train the S-S module. To be specific, given the skill sequence $\{s_1^0, \ldots, s_j^0, \ldots, s_m^0\}$, we simply and randomly mask some percentage of the sequence, and then input it into the S-S module. Based on the output novel embedding skill sequence $\{s_1^1, \ldots, s_j^1, \ldots, s_m^1\}$, we predict those masked ones based on the binary cross-entropy loss function, this process can be formulated as:

$$\mathcal{L}_2 = -\sum_j (\widetilde{s}_j \log \hat{s}_j), \tag{3}$$

where $\widetilde{s}_j$ is the ID of masked skill, and $\hat{s}_j$ is the prediction result, which can be obtained through a linear function given as:

$$\hat{s}_j = Softmax(\mathbf{w}_2^T \mathbf{s}_j^1 + b_2), \tag{4}$$

where s_j^1 is the intermediate embedding representation of the j^{th} skill, $\boldsymbol{w}_2$ and b_2 are parameters to be learned.

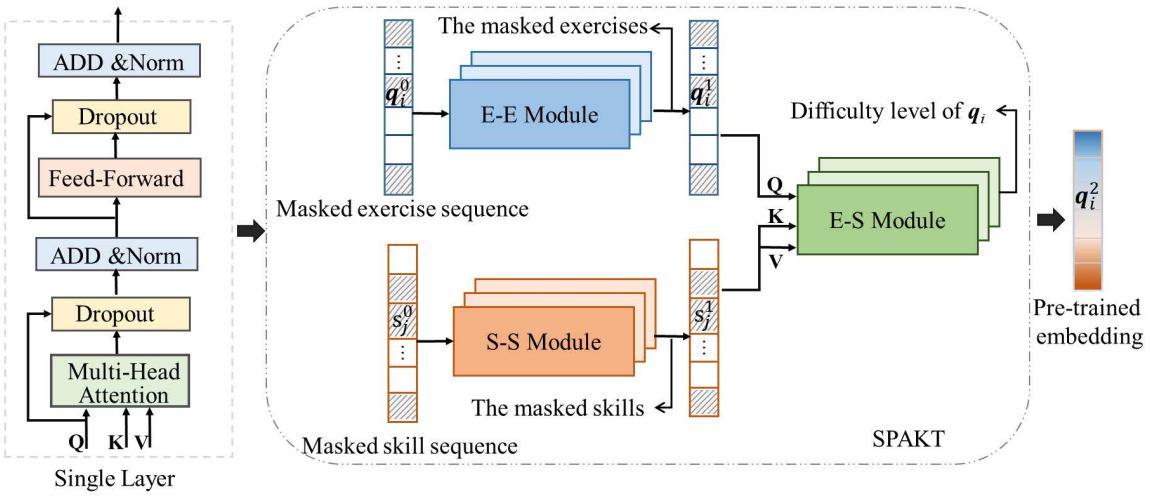


FIGURE 1. Detailed illustration of our self-supervised learning based pre-training network architecture.

3) E-S MODULE

As shown in Figure (1), through using two self-attention modules, we can obtain the intermediate embedding representation of exercise and skill, which are denoted as q_i^1 and s_i^1, respectively. In the following, we design the third self-attention module, i.e., E-S Module, to model the relationship between exercise and skill. In this paper, we set the difficulty level of exercises as the self-supervised information to train the E-S module.

For one exercise, we employ a linear projection to learn its difficulty level, as shown in Eq.(5).

$$\hat{d}_i = \sigma(\mathbf{w}_3^T \mathbf{q}_i^2 + b_3), \tag{5}$$

where $\sigma(x) = 1/(1 + e^{-x})$ is the sigmoid function, $\boldsymbol{w}_3$ and b_3 are parameters to be learned. q_i^2 is the final embedding representation for an exercise q_i. Then, we train the self-attention module in top layer through mean squared error loss. The process can be formulated as:

$$\mathcal{L}_3 = \sum_i (d_i - \hat{d}_i), \tag{6}$$

where $\hat{d}_i$ is the predicted difficulty level obtained through the equation as shown in (5), and d_i is the ground truth, which can be obtained by calculating the error ratio based on the training dataset. The process can be formulated as:

$$d_i = \frac{\sum_k \Pi(a_{ik} = 0)}{\sum_k \Pi(a_{ik} = 0) + \sum_k \Pi(a_{ik} = 1)}, \tag{7}$$

where $\Pi(\cdot)$ is an indicator function, that is, the value is 1 if $(\cdot)$ is true, and 0 otherwise. a_{ik} means whether the k^{th} student answered the i^{th} question correctly. As aforementioned, $a_{ik} = 1$ means answering correctly, and 0 otherwise.

4) JOINT OPTIMIZATION

Combining the aforementioned three loss functions, as shown in (1), (3) and (6), the total loss function is derived as:

$$min\ \lambda_1 \mathcal{L}_1 + \lambda_2 \mathcal{L}_2 + \lambda_3 \mathcal{L}_3, \tag{8}$$

where $\lambda_1, \lambda_2, \lambda_3$ are tradeoff parameters. Once the joint optimization is completed, we can get the final exercise embedding q_i^2, which can be feed into existing deep KT models, such as DKT [4] and DKVMN [12].

IV. EXPERIMENTS

A. EXPERIMENTAL SETUP

1) DATASETS

Our proposed SPAKT is evaluated on three real world datasets which are summarized in Table (1). ASSIST2009 and ASSIST2012 are both the most popular datasets used to KT models from the free online tutoring ASSISTment platform[1]. EdNet[2] was collected over two years from the intelligent online tutoring platform named Riid TUTOR13 in South Korea. Details about each dataset are presented below.

TABLE 1. The statistics of the datasets.

	ASSIST2009	ASSIST2012	EdNet
#students	4151	28834	5000
#exercises	16891	50983	12368
#skills	101	198	188
#records	283104	2629870	815570

- **ASSIST2009** was collected during the school year 2009-2010. Due to including duplicates when first released, the dataset was updated later. Based on the latest updated version, we remove users with less than three records, and remove the records without skills as well as scaffolding problems. After preprocessing, the dataset used in this article contains 283,104 interactions given by 4,151 students to a total of 16,891 distinct exercises and 101 skills.
- **ASSIST2012** is a larger version of the ASSISTments datasets consisting of data collected from Sept 2012 to Oct 2013. After the same preprocessing procedure as

[1]https://www.assistments.org/
[2]https://github.com/riiid/ednet

ASSIST2009, the dataset used in this article contains 2,629,870 interactions given by 28,834 students to a total of 50,983 distinct exercises and 198 skills.

- **EdNet** is a hierarchical dataset composed of four subsets, and each subset contains different types of student activities. In this article, we utilize the one subset composed of student's question-solving logs. Based upon sampling randomly, the EdNet dataset employed in our experiments consists of 12,368 distinct exercises answered by 5,000 students resulting in 8,155,706 interactions.

2) BASELINES

In this paper, we propose a novel pre-training approach for knowledge tracing, namely as SPAKT, which can be effortlessly incorporated into existing deep neural network-based KT frameworks. Thus the baselines used in this article aim to answer the following two research questions (RQ):

- RQ1: Whether our proposed SPAKT can enhance existing deep neural network based KT models as a data-preprocessing procedure for better features?
- RQ2: As a novel pre-training approach for knowledge tracing, whether our proposed SPAKT is superior (competitive) compared with existing pre-training approaches for knowledge tracing?

To this end, we utilize two groups of baselines for comparisons as follows: (1) for answering RQ1, we have taken the two most popular deep KT models, i.e., DKT [4] and DKVMN [12] as the baselines to evaluate the effectiveness of the pre-trained embedding representation by our proposed SPAKT. We first train our SPAKT to obtain embedding representation of exercises, and then feed it into DKT (namely SPAKT+DKT) or DKVMN model (namely SPAKT+DKVMN). More deep KT models [9], [10], [15] can also be adopted here, but we leave them for future exploration. (2) for answering RQ2, we make comparison with two existing embedding approaches, i.e., PEBG [13] and Rasch model [14], to evaluate the superior of the SPAKT. To the best of our knowledge, we do not find other pre-training approaches for knowledge tracing.

Details about each baseline are presented below.

- **DKT** [4] is the first deep learning-based KT model proposed by Piech *et al.* in 2015, to the best of our knowledge. It introduced the Long Short Term Memory (LSTM) model into the KT task. In order to train the LSTM model on student interactions, it converts those interactions into a sequence of fixed length input vectors based upon one-hot encoding mechanism. To be specific, for datasets with a small number of unique exercises, it simply employs one-hot encoding to represent an interaction. For datasets with a large number of unique exercises, it generates a random low-dimensional representation for the initial one-hot high-dimensional vector based on compressed sensing.
- **DKVMN** [12] is one of the most popular deep KT models using a variant of Memory-Augmented Neural Networks (MANN), and it is composed of one static key matrix and one dynamic value matrix. The input for MANN is a joint embedding of a distinct exercise tag and its response which is a binary value indicating whether the student answered the exercise correctly.
- **PEBG** [13] is a deep KT model proposed recently, which aims at boosting KT model performance through pre-training embedding for each question on abundant side information, including question difficulty and plentiful relations. It first represents the relationship between questions and skills as a bipartite graph, and then utilizes product-based neural networks to generate pre-training embedding representation.
- **Rasch** model is a popular embedding approach used in KT models [12], [14]. It constructs the embedding for an exercise from the skills this exercise covers, and characterizes the probability that a student correctly response an exercise through the question difficulty level and the students ability. Related research [14] shows that the exercise embedding learned by Rasch model can keep the similarity of exercises labeled as covering the same skill, and catch the individual differences between these exercises simultaneously.

In this article, we evaluate the performance of our SPAKT and baselines on predicting binary valued future student responses to exercises. The metric is the receiver operating characteristics curve (AUC), which is widely used to quantify the performance of KT models.

3) PARAMETERS SETTING

Without prior knowledge about exercise-skill relations, we utilize mask mechanism to train the S-S module and E-E module in our SPAKT framework. We randomly mask 15% exercises/skills of the input sequence for prediction. Since the [mask] exercises/skills do not appear in the actual training process, we do not always replace a ''masked'' word with the actual [mask] token. Instead, if the i^{th} token is selected, we replace it as (1) the [mask] token with 80% probability (2) a random exercise embedding with 10% probability, and (3) the unchanged i^{th} token with 10% probability. Our SPAKT can be effectively optimized with the Adam method [36]. It adopts the same attenuated learning rate as transformer [35], which varied over the course of training. The min-batch and the dimension of an exercise embedding are both set to be 128. For the context layer in E-E and S-S module, we set the number layer $L = 2$, head number $h = 8$, and for the cross-fusion layer in E-S module, we set the number layer $L = 1$, head number $h = 1$. For the hyperparameters λ_i, $i \in \{1, 2, 3\}$, we utilize grid search to find best performing values in the range of $[0, 1]$.

B. MODEL COMPARISON

As a novel approach for obtaining the pre-train embedding of exercises, our proposed SPAKT is first trained to

TABLE 2. The AUC results followed by DKMVN over three datasets.

Methods	ASSIST2009	ASSIST2012	EdNet
DKVMN [12]	73.98	67.10	64.16
Rasch+DKVMN [14]	72.78	69.91	68.84
PEBG+DKVMN [13]	**81.72**	**76.16**	74.56
SPAKT+DKVMN	81.56	76.13	**74.62**

TABLE 3. The AUC results followed by DKT over three datasets.

Methods	ASSIST2009	ASSIST2012	EdNet
DKT [4]	**74.88**	67.13	67.06
Rasch+DKT [14]	74.19	66.81	66.73
PEBG+DKT [13]	73.92	**72.86**	72.65
SPAKT+DKT	74.29	72.77	**73.64**

generate the embedding representation, which will be fed into existing deep learning-based KT models, i.e., DKT and DKVMN. Thus, in this section, we renamed our SPAKT as SPAKT+DKT and SPAKT+DKVMN, which means that our SPAKT is followed by DKT and DKVMN, respectively. We also compare our SPAKT against two other pre-training exercise embedding methods, i.e., PEBG [13] and Rasch model [14]. Similarly, we renamed PEBG and Rasch as PEBG+DKT (PEBG+DKVMN) and Rasch+DKT (Rasch+DKMVM), respectively, if it was followed by DKT (DKVMN). Results are presented in Table (2) and Table (3), in which we highlight the optimal results in bold and the second best one with an underscore.

As seen, even without using expensive E-E, S-S, and E-S relationship annotations, our SPAKT still achieves competitive performance against other baselines. On the larger dataset EDNet, it both achieves the optimal performance whether followed by DKVMN (see Table (2)) or DKT (see Table (3)). On ASSIST2009 and ASSIST2012, it still achieves suboptimal performance against other competitors. Particularly, Table (2) illustrates the predictive performance of DKVMN and DKVMN-followed pre-training approaches. It can be easily observed: 1) the proposed SPAKT can lead to substantial performance gains, compared with the DKVMN model. On ASSIST2009, ASSIST2012 and EdNet, SPAKT improves the AUC by 7.58%, 9% and 10.46%, over DKVMN, respectively. It implies the effectiveness of the embedding representations pre-trained by our SPAKT. 2) even without any prior knowledge about E-S, E-E and S-S relationships, the proposed SPAKT outperforms all other methods on EdNet. On the two other datasets, it achieves the suboptimal performance against other competitors, i.e., it's just slightly worse than PEBG [13] model, which utilizes the expert-labeled relationships to obtain exercise embedding. The similar observations can be achieved from Table (3), which illustrates the AUC values of DKT model and DKT-followed pre-training approaches. Additionally, we note that the more recent DKVMN method is worse than DKT in our implementation. The reason may be that data preprocessing operations are different across different articles.

TABLE 4. The AUC results of SPAKT variants followed by DKT over three datasets.

Variant	DKT		
	ASSIST2009	ASSIST2012	EdNet
SPAKT-EE	↓73.88	72.82	↓73.63
SPAKT-SS	↓73.93	↓72.55	↓73.63
SPAKT-ES	↓71.10	↓69.30	↓70.48
SPAKT	74.29	72.77	73.64

TABLE 5. The AUC results of SPAKT variants followed by DKVMN over three datasets.

Variant	DKVMN		
	ASSIST2009	ASSIST2012	EdNet
SPAKT-EE	↓81.55	76.30	74.82
SPAKT-SS	↓81.37	76.19	↓74.60
SPAKT-ES	↓75.70	↓72.73	↓72.04
SPAKT	81.56	76.13	74.62

C. STUDY ON EFFECTIVENESS OF NETWORK COMPONENTS

We investigate the effectiveness of essential components of our model by presenting an ablation study. We provide three variants through removing one of the three self-attention modules, and rename it as "SPAKT-EE," "SPAKT-SS," and "SPAKT-ES," which means removing the E-E, S-S, and E-S module, respectively. The results are summarized in Table (4) and Table (5), in which ↓ represents performance degradation compared to baseline SPAKT.

As seen, removing the self-supervised module suffers from performance degradation in most cases. Specifically, directly removing the E-S module brings significant performance drops (e.g., 5.86 on ASSIST2012), which clearly shows the E-S relationship is indeed useful for KT tasks. It is worth noticing that "SPAKT-EE" suffers from 3 times performance drops, while have 3 times slightly performance increase. This observation suggests that it is considerably more challenging to learn the E-E relationship from those datasets owning more exercises or records. Overall, all components introduced in our approach lead to promising and competitive performance on the three real-world datasets.

D. STUDY ON THE LEARNED EXERCISE-SKILL RELATION

The important advantage of SPAKT is that it can implicitly discover the plentiful relationships via self-supervised information, which makes it convenient and inexpensive applied in real educational applications. To make further illustrations about this claim, we visualize the learned relation between exercises and its corresponding skills based on the learned attention weight matrix, in which the line and column denote an exercise and its predicted skills. For simplicity, the weight matrix generated from the results on dataset ASSIST2009 is used as an example. Firstly, we normalized the weight matrix row by row via the Softmax function. For better visualization, we then randomly selected five exercises from those exercises whose real skills ranked in the top 10 in the list of its predicted skills. Heatmap depicts the probability of each predicted skill

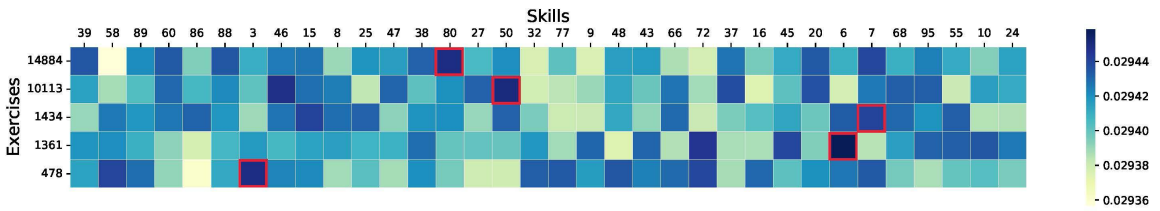


FIGURE 2. The quantized distribution map of the predicted probability of skills computed for an exercise.

for the five exercises, and the red boxes mark the ground truth (i.e., its corresponding skill), as shown in Figure (2).

As can be seen from the figure, SPAKT can learn the relationship between exercises and skills, to a certain extent. Take the exercise No.1361 as an example, the predicted skill No.6 is obviously owning the higher probability compared with other predicted skills, and it happens to be the real label of the exercise No.1361. Similar findings can be obtained from other four exercises.

V. CONCLUSION

In this paper, we proposed a novel pre-training model called Self-supervised Pre-trAining method for Knowledge Tracing (SPAKT), which introduced self-supervised learning to learn exercise embedding presentation. It allows us to automatically explore and exploit the plentiful E-E, E-S, and S-S relationships, instead of human-expert annotations, which is highly time-consuming and laborious. Our proposed pre-training model can be easily combined with existing deep neural network-based KT models. Experimental results showed that the proposed model can boost the predictive performance of existing KT models, such as DKT and DKVMN. Furthermore, compared with existing pre-training approaches based upon human-labeled relationships, our SPAKT is still competitive and achieves promising performance on three real-world datasets.

REFERENCES

[1] A. C. Graesser, P. Chipman, B. C. Haynes, and A. Olney, "AutoTutor: An intelligent tutoring system with mixed-initiative dialogue," *IEEE Trans. Educ.*, vol. 48, no. 4, pp. 612–618, Nov. 2005.

[2] Q. Liu, S. Shen, Z. Huang, E. Chen, and Y. Zheng, "A survey of knowledge tracing," 2021, *arXiv:2105.15106*.

[3] A. T. Corbett and J. R. Anderson, "Knowledge tracing: Modeling the acquisition of procedural knowledge," *User Model. User-Adapted Interact.*, vol. 4, no. 4, pp. 253–278, 1995.

[4] C. Piech, J. Spencer, J. Huang, S. Ganguli, M. Sahami, L. Guibas, and J. Sohl-Dickstein, "Deep knowledge tracing," in *Proc. 28th Int. Conf. Neural Inf. Proccess. Syst. (NIPS)*. Cambridge, MA, USA: MIT Press, 2015, pp. 505–513.

[5] Z. A. Pardos and N. T. Heffernan, "Modeling individualization in a Bayesian networks implementation of knowledge tracing," in *Proc. Int. Conf. User Modeling Adaptation Personalization (UMAP)*. Big Island, HI, USA, Jun. 2010, pp. 255–266.

[6] Z. A. Pardos and N. T. Heffernan, "KT-IDEM: Introducing item difficulty to the knowledge tracing model," in *Proc. Int. Conf. User Modeling Adaptation Personalization (UMAP)*. Cham, Switzerland: Springer, 2011, pp. 243–254.

[7] P. Nedungadi and M. S. Remya, "Incorporating forgetting in the personalized, clustered, Bayesian knowledge tracing (PC-BKT) model," in *Proc. Int. Conf. Cogn. Comput. Inf. Process. (CCIP)*, Mar. 2015, pp. 1–5.

[8] R. S. D. Baker, A. T. Corbett, and V. Aleven, "More accurate student modeling through contextual estimation of slip and guess probabilities in Bayesian knowledge tracing," in *Proc. Int. Conf. Intell. Tutoring Syst. (ITS)*. Cham, Switzerland: Springer, 2008, pp. 406–415.

[9] S. Shen, Q. Liu, E. Chen, H. Wu, Z. Huang, W. Zhao, Y. Su, H. Ma, and S. Wang, "Convolutional knowledge tracing: Modeling individualization in Student learning process," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Xi'an, China, Jul. 2020, pp. 1857–1860.

[10] Y. Yang, J. Shen, Y. Qu, Y. Liu, K. Wang, Y. Zhu, W. Zhang, and Y. Yu, "GIKT: A graph-based interaction model for knowledge tracing," 2020, *arXiv:2009.05991*.

[11] H. Nakagawa, Y. Iwasawa, and Y. Matsuo, "Graph-based knowledge tracing: Modeling student proficiency using graph neural network," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell. (WI)*, Oct. 2019, pp. 156–163.

[12] J. Zhang, X. Shi, I. King, and D.-Y. Yeung, "Dynamic key-value memory networks for knowledge tracing," in *Proc. 26th Int. Conf. World Wide Web*, Apr. 2017, pp. 765–774.

[13] Y. Liu, Y. Yang, X. Chen, J. Shen, H. Zhang, and Y. Yu, "Improving knowledge tracing via pre-training question embeddings," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, Jul. 2020, pp. 1577–1583.

[14] A. Ghosh, N. Heffernan, and A. S. Lan, "Context-aware attentive knowledge tracing," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 2330–2339.

[15] S. Minn, M. C. Desmarais, F. Zhu, J. Xiao, and J. Wang, "Dynamic student classiffication on memory networks for knowledge tracing," in *Advances in Knowledge Discovery and Data Mining*. Cham, Switzerland: Springer, 2019, pp. 163–174.

[16] S. Tong, Q. Liu, W. Huang, Z. Hunag, E. Chen, C. Liu, H. Ma, and S. Wang, "Structure-based knowledge tracing: An influence propagation view," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2020, pp. 541–550.

[17] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 2018, *arXiv:1810.04805*.

[18] Y. Ma, C. Cui, X. Nie, G. Yang, K. Shaheed, and Y. Yin, "Pre-course student performance prediction with multi-instance multi-label learning," *Sci. China Inf. Sci.*, vol. 62, no. 2, p. 29101, Feb. 2019.

[19] G. Abdelrahman, Q. Wang, and B. P. Nunes, "Knowledge tracing: A survey," 2022, *arXiv:2105.15106*.

[20] S. I. R. Luelmo, N. El Mawas, and J. Heutte, "Machine learning techniques for knowledge tracing: A systematic literature review," in *Proc. 13th Int. Conf. Comput. Supported Educ.* Setúbal, Portugal: Science and Technology Publications, 2021, pp. 60–70.

[21] T. Kaser, S. Klingler, A. G. Schwing, and M. Gross, "Dynamic Bayesian networks for student modeling," *IEEE Trans. Learn. Technol.*, vol. 10, no. 4, pp. 450–462, Oct./Dec. 2017.

[22] J. Gonzalez-Brenes, Y. Huang, and P. Brusilovsky, "General features in knowledge tracing to model multiple subskills, temporal item response theory, and expert knowledge," in *Proc. 7th Int. Conf. Educ. Data Mining*. Pittsburgh, PA, USA: Univ. of Pittsburgh, vol. 2014, pp. 84–91.

[23] F. Liu, X. Hu, C. Bu, and K. Yu, "Fuzzy Bayesian knowledge tracing," *IEEE Trans. Fuzzy Syst.*, early access, May 24, 2021, doi: 10.1109/TFUZZ.2021.3083177.

[24] H. Cen, K. Koedinger, and B. Junker, "Learning factors analysis—A general method for cognitive model evaluation and improvement," in *Proc. Int. Conf. Intell. Tutoring Syst.* Cham, Switzerland: Springer, 2006, pp. 164–175.

[25] P. I. Pavlik, Jr., H. Cen, and K. R. Koedinger, "Performance factors analysis—A new alternative to knowledge tracing," in *Proc. Conf. Artif. Intell. Educ.* Amsterdam, The Netherlands: IOS Press, vol. 2009, pp. 531–538.

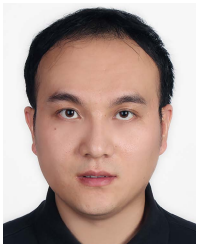
[26] Y. Gong, J. E. Beck, and N. T. Heffernan, "How to construct more accurate student models: Comparing and optimizing knowledge tracing and performance factor analysis," *Int. J. Artif. Intell. Educ.*, vol. 21, nos. 1–2, pp. 27–46, 2011.

[27] V. Mandalapu, J. Gong, and L. Chen, "Do we need to go deep? Knowledge tracing with big data," 2021, *arXiv:2101.08349*.
[28] N. Thai-Nghe, T. Horvath, and L. Schmidtthieme, "Factorization models for forecasting student performance," in *Proc. Int. Conf. EDM*, 2010, pp. 11–20.
[29] J. J. Vie and H. Kashima, "Knowledge tracing machines: Factorization machines for knowledge tracing," in *Proc. Conf. AAAI*, 2019, pp. 750–757.
[30] W. Gan, Y. Sun, S. Ye, Y. Fan, and Y. Sun, "Field-aware knowledge tracing machine by modelling students' dynamic learning procedure and item difficulty," in *Proc. Int. Conf. Data Mining Workshops (ICDMW)*, Nov. 2019, pp. 1045–1046.
[31] Y. Chen, Q. Liu, Z. Huang, L. Wu, E. Chen, R. Wu, Y. Su, and G. Hu, "Tracking knowledge proficiency of students with educational priors," in *Proc. ACM Conf. Inf. Knowl. Manage.*, Nov. 2017, pp. 989–998.
[32] W. Gan, Y. Sun, X. Peng, and Y. Sun, "Modeling learner's dynamic knowledge construction procedure and cognitive item difficulty for knowledge tracing," *Int. J. Speech Technol.*, vol. 50, no. 11, pp. 3894–3912, Nov. 2020.
[33] H. Liu, T. Zhang, F. Li, Y. Gu, and G. Yu, "Tracking knowledge structures and proficiencies of students with learning transfer," *IEEE Access*, vol. 9, pp. 55413–55421, 2021.
[34] Q. Liu, Z. Huang, Y. Yin, E. Chen, H. Xiong, Y. Su, and G. Hu, "EKT: Exercise-aware knowledge tracing for Student performance prediction," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 1, pp. 100–115, Jan. 2021.
[35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[36] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

YULING MA received the B.Sc., M.Sc., and Ph.D. degrees in computer science from Shandong University, China in 2003, 2008, and 2020, respectively. From 2014 and 2015, she was a Visiting Scholar with Nanjing University, China. She is currently a Lecturer with the School of Computer Science and Technology, Shandong Jianzhu University, China. Her research interests include machine learning and educational data mining.

PENG HAN received the bachelor's degree in computer science and technology from the Shandong Youth Political College, China, in 2018. He is currently pursuing the master's degree in software engineering with Shandong University, China. He is engaged in machine learning and data mining research with the MLA Laboratory. His research interests include educational data mining, focusing on the analysis and understanding of time series data.

HUIYAN QIAO received the bachelor's degree in electronic commerce from Luoyang Normal University, China, in 2020. She is currently pursuing the master's degree in computer application technology with Shandong Jianzhu University, China. She is engaged in machine learning and data mining research. Her research interest includes educational data mining.



CHAORAN CUI received the B.E. degree in software engineering and the Ph.D. degree in computer science from Shandong University, China, in 2010 and 2015, respectively. From 2015 to 2016, he was a Research Fellow with Singapore Management University, Singapore. He is currently a Professor with the School of Computer Science and Technology, Shandong University of Finance and Economics, China. His research interests include information retrieval, recommender systems, multimedia, and machine learning.

YILONG YIN received the Ph.D. degree from Jilin University, China, in 2000. From 2000 to 2002, he worked as a Postdoctoral Fellow with the Department of Electronic Science and Engineering, Nanjing University, China. He is currently a Professor with the School of Software Engineering and the Director of the MLA Laboratory. His research interests include machine learning and data mining, computational medicine, and biometrics.

DEHU YU received the B.Eng. degree from Northeast Petroleum University, in 1997, the M.Eng. degree from the Harbin University of Civil Engineering and Architecture, in 1999, and the Ph.D. degree from the Harbin Institute of Technology, China, in 2003. From 2005 to 2006, he was a Postdoctoral Researcher with Tongji University, China. He is currently the President and a Professor with Shandong Jianzhu University, China. His research interests include the intersection of artificial intelligence and engineering construction.