The Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI-23)

Deep Attentive Model for Knowledge Tracing

Xinping Wang, Liangyu Chen*, Min Zhang

East China Normal University
lychen@sei.ecnu.edu.cn

Abstract

Knowledge Tracing (KT) is a crucial task in the field of online education, since it aims to predict students' performance on exercises based on their learning history. One typical solution for knowledge tracing is to combine the classic models in educational psychology, such as Item Response Theory (IRT) and Cognitive Diagnosis (CD), with Deep Neural Networks (DNN) technologies. In this solution, a student and related exercises are mapped into feature vectors based on the student's performance at the current time step, however, it does not consider the impact of historical behavior sequences, and the relationships between historical sequences and students. In this paper, we develop DAKTN, a novel model which assimilates the historical sequences to tackle this challenge for better knowledge tracing. To be specific, we apply a pooling layer to incorporate the student behavior sequence in the embedding layer. After that, we further design a local activation unit, which can adaptively calculate the representation vectors by taking the relevance of historical sequences into consideration with respect to candidate student and exercises. Through experimental results on three real-world datasets, DAKTN significantly outperforms state-of-the-art baseline models. We also present the reasonableness of DAKTN by ablation testing.

Introduction

With the development of the online education system, knowledge tracing (KT), has become an important task in educational psychology, where it aims to predict whether a student can answer an exercise correctly according to his (or her) learning history [VanLehn 1988]. The results of knowledge tracing can enable a number of applications such as exercise recommendation [Tang et al. 2019; Zhou et al. 2018c] and adaptive learning [Liu et al. 2019].

Approaches have been proposed for knowledge tracing can be categorized as traditional methods, RNN-based methods and DNN-based methods. The traditional methods, such as IRT [Embretson and Reise 2004] and KTM [Vie and Kashima 2019], rely on a designed interaction function and statistical methods (e.g., maximum likelihood estimation). However, these methods often require professional expertise and manual labelling.

Therefore, RNN-based methods, such as DKT [Piech et al. 2015], are explored to solve the problem of knowledge tracing more automatically without excessive manual effort. However, similar to applying RNN in recommendation problems [Zhou et al. 2018a], it still exists some limitations for knowledge tracing problems. Different from the text sequence in the NLP task, the historical behavior sequences of students are not strictly continuous (e.g., a student may do some exercises online, and then complete the school assignments in private), which leads to the potential discontinuous behavior in the input sequence.

DNN-based methods, such as NCDM [Wang et al. 2020] and CDGK [Wang et al. 2021], are then designed for knowledge tracing. In these models, a student and related exercises are mapped into feature vectors based on the student's performance at the current time step. However, these models have two limitations: (1) Since the DNN model is not as easy to handle sequence type data as the RNN model, the existed DNN-based methods do not consider the impact of the historical sequence of students and exercises on the prediction results. (2) Although the historical sequence is critical in knowledge tracing, the exercises that students have completed may involve some knowledge concepts, not all knowledge concepts. Therefore, the past behavior will make different contributions to predict the current exercise. As shown in Figure 1, the knowledge tracing model predicts the answer of current exercise according to the historical sequence and attention intensity.

In this paper, we present a novel model DAKTN for knowledge tracing with consideration of the effort of historical behavior sequences. DAKTN applies a designed pooling layer, which can map the historical behavior sequence into a fixed-length feature vector. Also, considering the second limitation mentioned above, DAKTN adaptively calculates the feature vector according to the candidate student and related exercise, by introducing an activation unit to pay attention on the historical sequence. In summary, our key contributions are listed as follows.

- We point out the limit of DNN-based methods for knowledge tracing without historical behavior sequences and design a novel Deep Attentive Model for Knowledge Tracing (DAKTN). DAKTN applies a designed pooling layer, which can consider the historical behavior information in the knowledge tracing model.

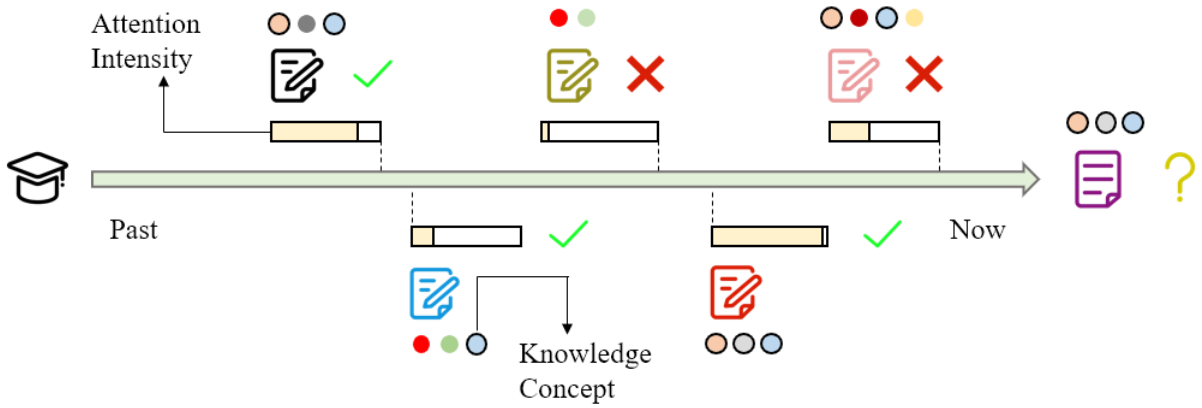

Figure 1: Illustration of knowledge tracing. The student chooses some exercises to answer and gets the scores. KT can predict the score of the new exercise by combining the student behavior sequences and the knowledge concepts.

- To better understand the historical information, and adaptively learn the representation of students and exercises from historical behavior sequences, we introduce an activation unit which can learn corresponding weights according to the current students and exercises.
- We provide empirical evidence of DAKTN by experiments on three real-world datasets, to show our method significantly outperforms the state-of-the-art related models by improving 5.37% on average with AUC. We also conduct experiments about ablation testing and sensitivity analysis, which can show the reasonableness of our model.

The rest of this paper is organized as follows. We show the related work in Section 2, and present the problem formulation in Section 3. After describing our method in Section 4, we provide the experimental results in Section 5, and conclude in Section 6.

Related Work

Generally, previous knowledge tracing methods can be categorized into three types: traditional methods, RNN-based methods and DNN-based methods. The traditional methods design the interaction function according to classic theories in educational psychology. The most typical methods include Deterministic Inputs, Noisy and Gate Model (DINA) [de la Torre 2009], which considers that the student can answer the exercise correctly only if he (or she) has mastered all necessary knowledge concepts for one exercise, and Item Response Theory (IRT) [Embretson and Reise 2004], which designs an interaction function to capture the relationships among the students, exercises and knowledge concepts. An improved IRT model is MIRT [Adams, Wilson, and chung Wang 1997], which is the multidimensional case of IRT.

With the development of deep learning [Hunt et al. 1992; Silver and Huang 2016], RNN-based methods have been proposed to solve KT more automatically by applying a recurrent neural network or LSTM. For example, DKT [Piech et al. 2015] represents the learning process of students with the hidden states of RNN. DHKT [Wang, Ma, and Gao 2019] extends DKT by considering both the knowledge concepts and the exercises, and DKVMN [Zhang et al. 2017] applies a key-value memory network to store the representation of knowledge concepts. Another way to extend DKT is to train a pre-training embedding as the prior experience for the model. For example, PEBG [Liu et al. 2020] uses pre-training question embeddings to exploit the information between exercises and knowledge concepts.

Different from the text sequence in the NLP task, the historical behavior sequences of students are not strictly continuous in KT, which leads to the potential discontinuous behavior in the input sequence. Therefore, DNN-based methods redesign the model structure by applying the deep neural network. The typical methods include NCDM [Wang et al. 2020], which incorporates DNN to learn the exercising interactions, and CDGK [Wang et al. 2021], which considers the knowledge concept aggregation to extend the NCDM. However, DNN-based methods do not consider the impact of the historical sequence of students and exercises on the prediction results. In this way, the representation vector will be a bottleneck to express the traits of students and exercises correctly.

Problem Formulation

A learning system $\mathcal{L}$ [Anderson et al. 2014] consists of several behavior sequences. Each sequence represents a student's past exercising interactions, where one interaction contains an exercise and the score the student left on the exercise. Besides, each exercise also contains several knowledge concepts. Formally, $\mathcal{L}$ is defined as follows.

Definition 1 (Learning System) *Let S be the set of all distinct students and E be the set of all distinct exercises, where $S = \{s_1, s_2, \ldots, s_M\}$ and $E = \{e_1, e_2, \ldots, e_N\}$. A learning System $\mathcal{L} = \bigcup_{i=1}^{M} \mathcal{L}_i$ is a set of behavior sequences of student s_i, which can be explicitly expressed as $\mathcal{L}_i = \bigcup_{t=1}^{T} (e_t, r_t)_i$, where*

- *$R = \bigcup_{t=1}^{T} \{r_t\}$ is a set of scores, where r_t is corresponding to the score on the time step t, and can be formulated*

as:

$$r_t = \begin{cases} 1, \textit{if exercise is answered correctly,} \\ 0, \textit{otherwise.} \end{cases}$$

- *Let $K = \{k_1, k_2, \ldots, k_K\}$ be the set of all distinct knowledge concepts. So the matrix denotes the relationship between the exercises and the knowledge concepts can be expressed as $\mathbf{Q} = \{Q_{ij}\}_{N \times K}$. Similar to the response r_t, Q_{ij} can be expressed as:*

$$Q_{ij} = \begin{cases} 1, \textit{if exercise } e_i \textit{ contains knowledge concept } k_j, \\ 0, \textit{otherwise.} \end{cases}$$

Knowledge tracing is a model to measure students' mastery on knowledge concepts according to the student behavior sequences. As shown in Figure 1, after the student practiced some exercises, knowledge tracing can predict the student's performance on exercises that have not been practiced. We now define the knowledge tracing problem.

Problem 1 *Given a learning system $\mathcal{L}$, which can be expressed as $\mathcal{L} = \bigcup_{i=1}^{M} \bigcup_{t=1}^{T} (e_t, r_t)_i$, the goal of our knowledge tracing problem is to predict the probability $p\left(r_{T+1} = 1 \mid e_{T+1}, \mathcal{L}\right)$ that student s_i will correctly answer the exercise e_{T+1} on the next time step.*

Method

Effective knowledge tracing models involve extracting targeted information from historical student behaviors. Features that depict students and exercises are the basic elements in the KT modeling of learning systems. In this section, we will present our Deep Attentive Knowledge Tracing Network (DAKTN) which can mine information from these features.

Feature Representation

To solve the knowledge tracing problem, we use three kinds of features, which can be classified as basic student features, basic exercise features and attribute features. The details are listed as follows.

Basic student features. Each student can be normally characterized with high-dimensional sparse binary features through encoding. Mathematically, the encoding vector of i-th student is formularized as $\vec{s_i} \in \{0, 1\}^M$. M denotes the dimensionality of the student features, which is equal to the number of distinct students.

Basic exercise features. Each exercise can be similarly represented as $\vec{e_j} \in \{0, 1\}^N$ through one-hot encoding. N denotes the dimensionality of the exercise features, which is equal to the number of distinct exercises.

Attribute features. Besides the basic student features and basic exercise features, data in knowledge tracing tasks also contain features related to the other information in the learning system, such as the average time of students answering questions, the attempt count, the school id and so on. The attribute features can be concatenated as $\mathbf{x} = [\mathbf{t_1}, \mathbf{t_2}, \ldots, \mathbf{t_s}]$, where the element $\mathbf{t_i}$ can be divided into the following two situations:

- $\mathbf{t_i}$ can be transformed into one-hot encoding if the i-th feature is categorical (e.g., school id).
- $\mathbf{t_i}$ can be represented as a scalar value if the i-th feature is numerical (e.g., the average time of students answering questions).

Base Model

Firstly, we introduce a base model, which combines the IRT theory in educational psychology and DNN, and also employed in [Wang et al. 2020, 2021]. It shares an Embedding Layer and Multilayer Perceptron (MLP) paradigm, as shown in Figure 2. To be specific, it consists of the following several parts:

Embedding layer. According to the Item Response Theory (IRT) [Embretson and Reise 2004] in educational psychology, the core knowledge tracing vector for the student s_i and the exercise e_j can be expressed as:

$$\vec{z_{ij}} = \vec{Q_j} \circ \left(\vec{M_i} - \vec{D_j} \right) \times \alpha_j, \tag{1}$$

where $\circ$ is element-wise product, Q_j indicates the knowledge concepts contained in the exercise e_j, $\vec{M_i}$ represents the student's mastery on each knowledge concept, $\vec{D_j}$ represents the difficulty of each knowledge concept in the exercise, and α_j indicates the discrimination of the exercise.

As the inputs, student feature $\vec{s_i}$ and exercise feature $\vec{e_j}$, are represented as one-hot encoding, then the base model applies the embedding layer to transform them into the IRT-related features mentioned above. Explicitly, the IRT-related features can be expressed as:

$$\vec{Q_j} = \vec{e_j} \times \mathbf{Q}, \tag{2}$$

$$\vec{M_i} = \sigma\left(\vec{s_i} \times \mathbf{A}\right), \tag{3}$$

where the matrix $\mathbf{Q}$ is pre-labelled by experts, $\mathbf{A} \in \mathbb{R}^{M \times K}$ is a trainable matrix, and σ represents the *sigmoid* activation function. Similarly, the difficulty vector $\vec{D_j}$ can be expressed as $\vec{D_j} = \sigma\left(\vec{e_j} \times \mathbf{B}\right)$, where $\mathbf{B} \in \mathbb{R}^{N \times K}$ is a trainable matrix, and the discrimination value α_j can be expressed as $\alpha_j = \sigma\left(\vec{e_j} \times \mathbf{C}\right)$, where $\mathbf{C} \in \mathbb{R}^{N \times 1}$ is also a trainable matrix.

Positive Multilayer Perceptron (PMLP). Given the IRT-related dense representation vector $\vec{z_{ij}}$, the base model combines monotonicity assumption [Reckase 2009; Samek et al. 2017] and fully connected layers to output the probability p_{ij} that student s_i answers the exercise e_j correctly. The details are listed as follows

- A single layer feedforward neural network is used to output the result, which can be formulated as:

$$p_{ij} = f\left(\left[\mathbf{W} \times \vec{z_{ij}} + \vec{b}\right] || \vec{a_{ij}}\right), \tag{4}$$

where f is the non-linear activation function, $\mathbf{W}$ and $\vec{b}$ are trainable parameters, and a_{ij} is the concatenation of

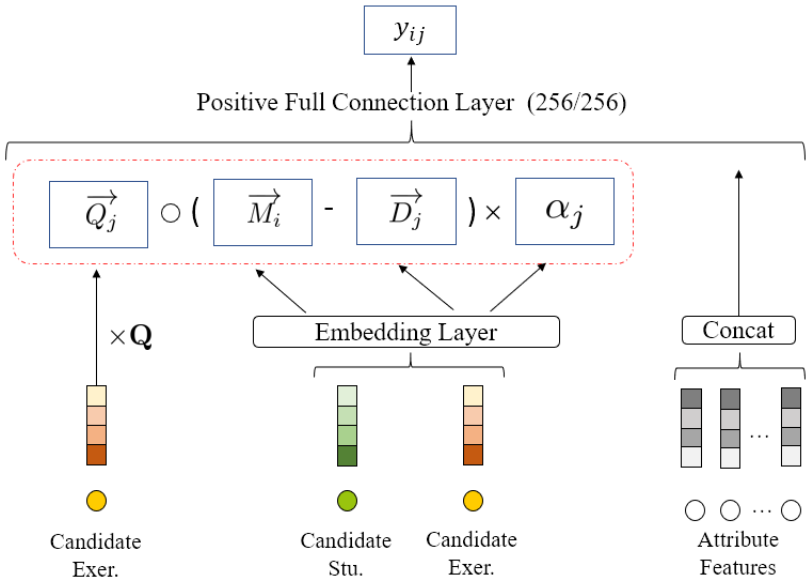

Figure 2: Illustration of the base model. The base model combines the IRT theory in educational psychology and Deep Neural Network by sharing an Embedding and MLP paradigm.

the attribute features. The number of layers of the fully connected layer can be adjusted according to the situation.

- The monotonicity assumption holds that the probability of a student answering the exercise correctly increases monotonically with the student's mastery of the knowledge concepts. The base model restricts each element of $\mathbf{W}$ to be positive to satisfy the monotonicity assumption, so that the gradient can be positive during the process of training.

The Structure of Deep Attentive Knowledge Tracing Network

The base model obtains a representation vector of IRT-related features by mapping from one-hot encoding of students and exercises to the trait vectors $\overrightarrow{M_i}$, $\overrightarrow{D_j}$ and α_j respectively, as shown in Equation (3). These representations only consider the current student and exercise, in regardless of the student behavior sequence. In this way, the IRT-related vector will be a bottleneck to express student and exercise traits truly. Indeed, the exercises that have been answered correctly by which students in the past can help construct the difficulty vector of the exercises. Similarly, the student has mastered which exercises in the past can also help build the student's mastery vector on knowledge concepts.

We design a novel model named Deep Attentive Knowledge Tracing Network (DAKTN) to incorporate the student behavior sequence in the embedding layer. Also, considering that although the student behavior sequence is critical in knowledge tracing, the exercises that students have completed may involve several knowledge concepts, not all knowledge concepts. Therefore, the past behavior will make different contributions to predict the current exercise, as shown in Figure 1. Naturally, we introduce an activation unit, with which the representation of IRT-related features can vary adaptively according to different candidate students and exercises, as shown in Figure 3. Details are introduced as below.

Pooling layer. As to the first limitation mentioned in Section 1, we design a pooling layer to incorporate the historical sequence in our model. Notice that different students have different numbers of behaviors, and different exercises have been answered by different numbers of students, which will generate a list of embedding vectors with different lengths. As positive multilayer perceptrons can only handle fixed-length IRT-related vectors, we transform the list of embedding vectors via a pooling layer to get a fixed-length vector inspired by the practice in recommender system [Covington, Adams, and Sargin 2016], which can be expressed as the following equations:

$$\overrightarrow{M_i} = \text{pooling}\left(\overrightarrow{M_{i1}}, \overrightarrow{M_{i2}}, \ldots, \overrightarrow{M_{iN}}\right), \tag{5}$$

$$\overrightarrow{D_j} = \text{pooling}\left(\overrightarrow{D_{j1}}, \overrightarrow{D_{j2}}, \ldots, \overrightarrow{D_{jN}}\right), \tag{6}$$

where N represents the length of the historical sequence. Sum pooling and average pooling are two kinds of pooling layers which are most commonly used. Sum pooling applies element-wise sum operations to the list of embedding vectors, while average pooling applies element-wise average operations.

Weighted Sum Pooling. For the second limitation of different effects from different historical sequences, we extend the pooling layer with weighted sum pooling. The pooling layer can incorporate the student behavior sequence in the embedding layer. Instead of expressing all historical behavior with the same vector, our model adaptively calculates

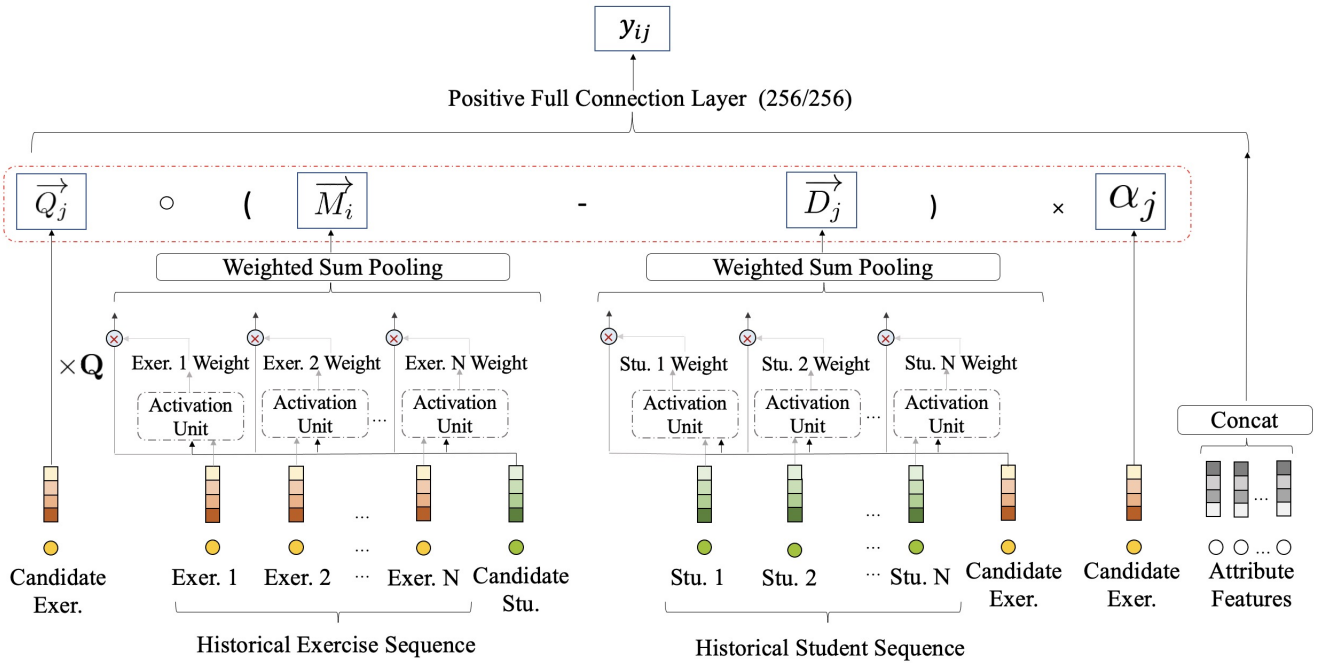

Figure 3: Description of knowledge tracing by DAKTN. Compared with the base model, it incorporates the student behavior sequence in the embedding layer. DAKTN also introduces an activation unit, with which the representation of IRT-related features varies adaptively according to different candidate students and exercises.

the representation vector of IRT-related features by taking the relevance of historical sequences into consideration with respect to candidate student and candidate exercise.

Figure 3 illustrates the architecture of our model DAKTN. Specifically, DAKTN introduces a local activation unit to conduct a weighted sum pooling, which can adaptively calculate the IRT-related vectors:

$$\overrightarrow{V_i} = g\left(\overrightarrow{p_C}, \overrightarrow{p_1}, \overrightarrow{p_2}, \ldots, \overrightarrow{p_N}\right), \tag{7}$$

$$g\left(\overrightarrow{p_C}, \overrightarrow{p_1}, \overrightarrow{p_2}, \ldots, \overrightarrow{p_N}\right) = \sum_{k=1}^{N} \overrightarrow{w_k} \cdot \overrightarrow{p_k}, \tag{8}$$

$$\overrightarrow{w_k} = h\left(\left[\overrightarrow{p_C} - \overrightarrow{p_k} \,||\, \overrightarrow{p_C} \,||\, \overrightarrow{p_k}\right] \times \mathbf{E}\right), \tag{9}$$

where $||$ is a concatenation operator, $\overrightarrow{V_i}$ can correspond to $\overrightarrow{D_j}$ and $\overrightarrow{M_i}$ according to exercise sequence and student sequence respectively, $\left\{\overrightarrow{p_1}, \overrightarrow{p_2}, \ldots, \overrightarrow{p_N}\right\}$ is the list of embedding vectors of behavior sequence with length of N, $\overrightarrow{p_C}$ is the embedding vector of the candidate student or the candidate exercise, $h(\cdot)$ is a feedforward network which outputs the activation weight. Different from traditional attention mechanisms, we concat the element-wise difference vectors and two input embedding vectors to reduce the loss of information, which is inspired by DIN [Zhou et al. 2018b].

We train DAKTN using cross entropy between the real score r_t and the predicted probability y_{ij}. The loss function can be expressed as:

$$-\sum_{i}\sum_{j}\left(r_t \log y_{ij} + (1 - r_t)\log\left(1 - y_{ij}\right)\right). \tag{10}$$

Experiments

In this section, we conduct experiments to evaluate the performance of our model DAKTN on knowledge tracing. Specifically, we answer the following three research questions.

- RQ1: How do DAKTN and other baselines perform on real-world datasets?
- RQ2: How do each module of DAKTN contribute to the model performance?
- RQ3: How do the sensitivity of different hyperparameters affect the prediction results?

We describe the datasets and baselines in Section 5.1 and 5.2 respectively. After that, we introduce some implementation details of the experiments in Section 5.3, and present the explanations to answer the above research questions in Section 5.4 $\sim$ 5.6 respectively.

Datasets

In our experiments, we use the following three real-world datasets. The basic statistics of the datasets are summarized in Table 1.

ASSIST09 and **ASSIST12** are both collected by the ASSISTments online tutoring system [Feng, Heffernan, and Koedinger 2009]. ASSIST09 provides 190 thousand scores

Dataset	ASSIST09	ASSIST12	EdNet
Students	4,163	24,432	5000
Exercises	17,746	46,394	13,169
Knowledge Concepts	123	265	188
Answers	190,320	1,867,167	222,141
KC per Exercise	1.19	1.02	2.28
Positive Label Rate	63.80 %	69.60 %	59.69 %

Table 1: Dataset summary.

answered by 4 thousand students on 17 thousand exercises with 123 knowledge concepts. ASSIST12 contains 24 thousand students, 46 thousand exercises with 265 knowledge concepts, and 1.8 million records.

EdNet [Choi et al. 2020] is an open dataset collected by a multi-platform AI tutoring service Santa. We use EdNet-KT1 dataset in our experiments, which contains 5 thousand students, 13 thousand exercises with 188 knowledge concepts, and 222 thousand records.

Baselines

All of the baselines and DAKTN use the same datasets in our experiments which have been described in Section 5.1. The compared models can be classified into the following three groups.

Traditional methods. These methods predict the students' scores according to the theory of educational psychology. We evaluate the following methods:

- IRT [Embretson and Reise 2004] uses parameter estimation to learn the traits which affect students' scores.
- KTM [Vie and Kashima 2019] applies the factorization machine to predict students' scores.

RNN-based methods. Considering that the data for knowledge tracing are consisted of sequences, these methods apply Recurrent Neural Network (RNN) to predict the results.

- DKT [Piech et al. 2015] uses recurrent neural network to model students' mastery on knowledge concepts.
- DHKT [Wang, Ma, and Gao 2019] is an extension model of DKT, which considers both the knowledge concepts and the exercises.
- PEBG [Liu et al. 2020] uses pre-training question embeddings to improve DKT.

DNN-based methods. These methods apply Deep Neural Network to capture non-linear interactions between exercise features, student traits and knowledge concepts. But the historical sequences are not addressed in these methods.

- NCDM [Wang et al. 2020] uses artificial neural network to make prediction.
- CDGK [Wang et al. 2021] is an extension model of NCDM, which considers the aggregation of knowledge concepts to improve the performance.

Implementation Details

In the step of data preprocessing, we first filter students who have answered fewer than 5 exercises to ensure each student with enough learning data. Then each dataset is divided into three parts, namely, 70%, 10% and 20% for training, validating and testing respectively. Note that different datasets have different attribute features. Thus, for ASSIST09 and ASSIST12 datasets, we select the teacher id, school id and average response time as attribute features. For the EdNet dataset, we only select the average response time as an attribute feature.

As to the implementation parameters, we initialize the parameters with batch normalization [Ioffe and Szegedy 2015], and use the Adam algorithm [Kingma and Ba 2015] to optimize our model. The dimensions of the positive full connection layers in Equation (4) are 256, 256, 1 respectively. We set the learning rate as 0.001, and use dropout with $p = 0.4$ to alleviate overfitting. We repeat the experiment five times on each model, and evaluate the performance through the average test accuracy, RMSE and AUC.

All models are implemented by Tensorflow 2.3 using Python 3.7, and all experiments are executed on a CentOS Linux server with the main configuration of GPU RTX 2080Ti, CPU@3.30GHz, 8GB RAM, and 1TB SSD Disk.

Performance Prediction

Table 2 shows the experimental results of our model DAKTN and other baseline models with three evaluation metrics including accuracy, RMSE (root mean square error) and AUC (area under the curve) [Bradley 1997]. The thresholds of accuracy are set based on the maximum value of F1-score.

From Table 2, we can observe that our model DAKTN works best. Detailedly, DAKTN achieves an AUC of 0.8466 on the ASSIST09 dataset, which outperforms the runner-up result by 4.34%. On the ASSIST12 dataset, DAKTN achieves an AUC of 0.7908, which improves the performance by 6.46% compared with the runner-up result. As for the EdNet dataset, DAKTN also improves the runner-up result by 5.30% with the AUC result of 0.8123. Besides, our model also outperforms other baselines in accuracy and RMSE. Thus, we answer RQ1.

We also observe that the RNN-based methods, which consider the historical sequence in the model, achieve an average better performance than the DNN-based methods on all metrics including AUC, RMSE and accuracy. This observation can also illustrate the importance of historical performance sequences in knowledge tracing task, and prove that classical RNN structure cannot handle such non-strictly continuous sequences with a good performance.

Ablation Study

To see how the historical performance sequences and weighted sum pooling affect model performance, we run DAKTN on three datasets to show our results of ablation study. We set four ablation modules, and the results are presented in Table 3. The details of the four ablation modules are listed below:

Category	Method	ASSIST09			ASSIST12			EdNet		
		Accuracy	RMSE	AUC	Accuracy	RMSE	AUC	Accuracy	RMSE	AUC
Traditional	IRT	0.6502	0.4673	0.6744	0.6137	0.4855	0.6308	0.5981	0.5012	0.6022
	KTM	0.7218	0.4463	0.7141	0.6911	0.4613	0.6882	0.6692	0.4679	0.6719
RNN-based	DKT	0.7179	0.4415	0.7252	0.6842	0.4602	0.6993	0.6889	0.4629	0.6810
	DHKT	0.7339	0.4274	0.7458	0.7298	0.4308	0.7276	0.7126	0.4364	0.7219
	PEBG	0.7992	0.4091	0.8032	0.7507	0.4211	0.7535	0.7538	0.4240	0.7593
DNN-based	NCDM	0.7208	0.4466	0.7192	0.6804	0.4606	0.6895	0.6728	0.4651	0.6704
	CDGK	0.7281	0.4392	0.7344	0.7018	0.4454	0.7172	0.7012	0.4581	0.7006
	DAKTN	**0.8387**	**0.3872**	**0.8466**	**0.8204**	**0.4030**	**0.8181**	**0.8080**	**0.4056**	**0.8123**

Table 2: The average test accuracy, RMSE and AUC of student performance prediction on three datasets. Best results are in bold, and runner-up results are underlined.

- **RHS** (Remove Historical Sequences) does not consider both the historical exercise sequences and the historical student sequences.
- **RHES** (Remove Historical Exercise Sequences) does not consider the historical exercise sequences.
- **RHSS** (Remove Historical Student Sequences) does not consider the historical student sequences.
- **RSPSP** (Replace Weighted Sum Pooling with Sum Pooling) applies the sum pooling to generate the IRT-related vectors.

Except for the settings mentioned above, other components of the model and experimental settings remain unchanged.

Model	ASSIST09	ASSIST12	EdNet
RHS	0.7192	0.6895	0.6704
RHES	0.7569	0.7421	0.7270
RHSS	0.7611	0.7304	0.7392
RSPSP	0.8123	0.7780	0.7702
DAKTN	**0.8466**	**0.8181**	**0.8123**

Table 3: Results of ablation testing on the average test AUC.

From Table 3, we can observe that each component removed will cause a decrease in performance, which indicates the effect of each component of the model. We can also observe that the absence of the weighted sum pooling hurts the precision worse, which indicates that our model DAKTN is reasonable.

Sensitivity Analysis

To investigate the sensitivity of our model DAKTN, we evaluate the impact of different lengths of the historical sequences on the prediction performance. For the case of insufficient historical sequences, we set a null value in the experiments. From Figure 4, we can observe that the average AUC trends for different lengths of the historical sequences differ greatly. When the sequence length of the historical sequence is small, the performance of the model is poor, but when the length reaches a certain level, the effect of the model tends to be stable, which can explain the importance of the pooling layer in the model component, and also prove that DAKTN can adaptively handle sequence features with weighted sum pooling. Thus, we answer RQ3.

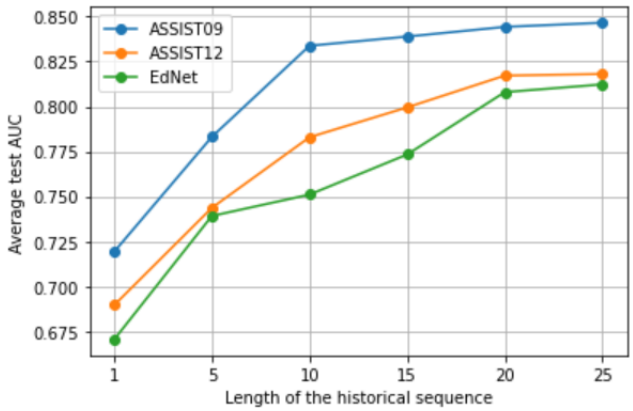

Figure 4: Prediction performance with different lengths of the historical sequences.

Conclusion

Knowledge tracing is a crucial task in the field of education analysis and evaluation. The historical behavior sequences which contain much learning information, are useful and necessary to be analyzed. In this paper, we present a novel model DAKTN, which combines DNN and historical behavior sequences for better performance. By a designed pooling layer, DAKTN can map the historical behavior sequence into a fixed-length feature vector. Besides, it can adaptively calculate the feature vector according to the candidate student and exercise, by introducing an activation unit to pay attention on the historical sequence. Experimental results of DAKTN on real-world datasets show that our model significantly outperforms the state-of-the-art related models. We also provide empirical evidence with ablation tests to prove that the optimization is critical to the result of our model.

Acknowledgments

This work is supported by NSFC (Nos. 61871186, 62272416) and NSFC-ISF Joint Program (Nos. 62161146001,3420/21)

References

Adams, R. J.; Wilson, M.; and chung Wang, W. 1997. The Multidimensional Random Coefficients Multinomial Logit Model. *Applied Psychological Measurement*, 21(1): 1–23.

Anderson, A.; Huttenlocher, D.; Kleinberg, J.; and Leskovec, J. 2014. Engaging with Massive Online Courses. In *Proceedings of the 33th AAAI Conference on Artificial Intelligence*, 687–698.

Bradley, A. P. 1997. The Use of the Area under the ROC Curve in the Evaluation of Machine Learning Algorithms. *Pattern recognition*, 30(7): 1145–1159.

Choi, Y.; Lee, Y.; Shin, D.; Cho, J.; Park, S.; Lee, S.; Baek, J.; Bae, C.; Kim, B.; and Heo, J. 2020. EdNet: A Large-Scale Hierarchical Dataset in Education. In *Proceedings of the 21st International Conference on Artificial Intelligence in Education*, 69-73.

Covington, P.; Adams, J.; and Sargin, E. 2016. Deep Neural Networks for YouTube Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems*, 191–198.

de la Torre, J. 2009. DINA Model and Parameter Estimation: A Didactic. *Journal of Educational and Behavioral Statistics*, 34(1): 115–130.

Embretson, S.; and Reise, S. P. 2004. Item Response Theory For Psychologists. *Quality of Life Research*, 13: 715–716.

Feng, M.; Heffernan, N.; and Koedinger, K. 2009. Addressing the assessment challenge with an online system that tutors as it assesses. *User Modeling and User-Adapted Interaction*, 19(3): 243–266.

Hunt, K. J.; Sbarbaro, D. G.; Zbikowski, R.; and Gawthrop, P. 1992. Neural networks for controlsystems: A survey. *Automatica*, 28(6): 1083–1112.

Ioffe, S.; and Szegedy, C. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the 32nd International Conference on Machine Learning*, 448-456.

Kingma, D. P.; and Ba, J. 2015. Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6980.

Liu, Q.; Tong, S.; Liu, C.; Zhao, H.; Chen, E.; Ma, H.; and Wang, S. 2019. Exploiting Cognitive Structure for Adaptive Learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 627-635. ACM.

Liu, Y.; Yang, Y.; Chen, X.; Shen, J.; Zhang, H.; and Yu, Y. 2020. Improving Knowledge Tracing via Pre-training Question Embeddings. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, 1577-1583.

Piech, C.; Bassen, J.; Huang, J.; Ganguli, S.; Sahami, M.; Guibas, L.; and Sohl-Dickstein, J. 2015. Deep Knowledge Tracing. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, 505–513.

Reckase, M. D. 2009. Multidimensional Item Response Theory Models. *Springer Press*, 79-112.

Samek, W.; Binder, A.; Montavon, G.; Lapuschkin, S.; and Muller, K.-R. 2017. Evaluating the Visualization of What a Deep Neural Network Has Learned. *IEEE Transactions on Neural Networks and Learning Systems*, 28(11): 2660-2673.

Silver, D.; and Huang, A. 2016. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587): 484–489.

Tang, X.; Chen, Y.; Li, X.; Liu, J.; and Ying, Z. 2019. A reinforcement learning approach to personalized learning recommendation systems. *British Journal of Statistical Psychology*, 72(1): 108–135.

VanLehn, K. 1988. Student modeling. *Foundations of Intelligent Tutoring Systems*, 55–78.

Vie, J.-J.; and Kashima, H. 2019. Knowledge Tracing Machines: Factorization Machines for Knowledge Tracing. In *Proceedings of the 33th AAAI Conference on Artificial Intelligence*, 750-757.

Wang, F.; Liu, Q.; Chen, E.; Huang, Z.; Chen, Y.; Yin, Y.; Huang, Z.; and Wang, S. 2020. Neural Cognitive Diagnosis for Intelligent Education Systems. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, 6153-6161.

Wang, T.; Ma, F.; and Gao, J. 2019. Deep Hierarchical Knowledge Tracing. In *Proceedings of the 12th International Conference on Educational Data Mining*, 671-674.

Wang, X.; Huang, C.; Cai, J.; and Chen, L. 2021. Using Knowledge Concept Aggregation towards Accurate Cognitive Diagnosis. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 2010-2019.

Zhang, J.; Shi, X.; King, I.; and Yeung, D.-Y. 2017. Dynamic Key-Value Memory Networks for Knowledge Tracing. In *Proceedings of the 26th International Conference on World Wide Web*, 765–774.

Zhou, G.; Zhu, X.; Song, C.; Fan, Y.; Zhu, H.; Ma, X.; Yan, Y.; Jin, J.; Li, H.; and Gai, K. 2018a. Deep Interest Network for Click-Through Rate Prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, 1059–1068. Association for Computing Machinery.

Zhou, G.; Zhu, X.; Song, C.; Fan, Y.; Zhu, H.; Ma, X.; Yan, Y.; Jin, J.; Li, H.; and Gai, K. 2018b. Deep Interest Network for Click-Through Rate Prediction. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1059–1068.

Zhou, Y.; Huang, C.; Hu, Q.; Zhu, J.; and Tang, Y. 2018c. Personalized learning full-path recommendation model based on LSTM neural networks. *Information Sciences*, 444: 135–152.