KT-XL: A Knowledge Tracing Model for Predicting Learning Performance Based on Transformer-XL

Yu He, Xinying Hu, Zhongtian Xu and Guangzhong Sun†

School of Computer Science and Technology, University of Science and Technology of China, Hefei, China

{heyu3761,hxiny,xuzt}@mail.ustc.edu.cn,gzsun@ustc.edu.cn

ABSTRACT

With the development of artificial intelligence (AI) technology, online teaching systems have become intellectualized. Knowledge tracing is an important task of intelligent teaching system. The goal of knowledge tracing is to trace students' knowledge status based on their past performance. It is widely used for predicting students' performance and building knowledge graph, which plays an important role in constructing adaptive (personalized) teaching systems. Previous models can not deal with subjective problems, and the performance is somewhat poor when input exercise sequences are long. In this paper, we propose a knowledge tracing model for subjective problems based on Transformer-XL, which can predict students' performance of a specific exercise. Specifically, we introduce a recurrence mechanism to the model to capture longer-term dependency and achieve better performance on both short and long sequences. Experiments on multiple real-world data sets and synthetic data sets show that our model performs better in predicting students' performance than state of the art models.

CCS CONCEPTS

• Applied computing → Computer-assisted instruction.

KEYWORDS

knowledge tracing, self-attention, recurrence mechanism,online education

ACM Reference format:

Yu He, Xinying Hu, Zhongtian Xu and Guangzhong Sun. 2020. KT-XL: A Knowledge Tracing Model for Predicting Learning Performance Based on Transformer-XL. In *ACM Turing Celebration Conference - China (ACM TURC 2020), May 22–24, 2020, Hefei,China. ACM, New York, NY, USA, 5 pages.* https://doi.org/10.1145/3393527.3393557

†The corresponding author

ACM TURC'20, May 22–24, 2020, Hefei, China

ACM ISBN 978-1-4503-7534-4/20/05...$15.00
https://doi.org/10.1145/3393527.3393557

1 Introduction

Giving accurate and rapid feedback to examinees is very important in the field of education. It has been proved that rapid feedback improves course outcomes. In a controlled experiment, students' final grades improved when feedback was delivered quickly, but not if delayed by 24 hours [1]. At present, with the development of artificial intelligence technology, online teaching systems have become intellectualized. Intelligent teaching systems (ITS) are different from previous MOOC systems [2]. Since ITS can formulate effective learning paths for students personally. Moreover, it can trace students' current mastery of knowledge based on their performance on the previous problems, so as to teach students according to their aptitude [3]. In ITS, students can obtain appropriate guidance and acquire relevant knowledge in the process of completing the exercises [4]. In ITS, there are two models named cognitive diagnosis model and knowledge tracing model for tracing and judging the knowledge mastery of students, separately. The probability that a student will be able to answer the exercise correctly depends on student's mastery of knowledge, which represents the depth and robustness of the knowledge concepts mastered by the student [5].

The goal of knowledge tracing is to trace students' knowledge status based on their past performance. Knowledge tracing is an important task of intelligent teaching system. The teacher can provide appropriate tips and customize the sequence of exercises based on student's mastery of knowledge concepts. Students know about their progress and can focus more on less familiar knowledge concepts, so as to learn more effectively [6]. Simulating student knowledge states effectively has a significant impact in the field of education, but using a number to represent the learning process is difficult. Generally, knowledge tracing is formulated as a supervised sequence problem: given student's performance on the previous exercises, predicting the performance when the student answers new exercise. The task of knowledge tracing can be formalized as: given observations of interactions $x_0, x_1, x_2, \ldots\ldots x_t$ taken by a student on a particular learning task, predict aspects of their next interaction x_{t+1} [7].

Traditional knowledge tracing models such as Bayesian Knowledge Tracing (BKT) [8]predict the next state based on the state transition matrix. Recently, deep learning models such as Deep Knowledge Tracing (DKT) [9] and its variant DKT+ [5] used Recurrent Neural Network (RNN) to summarize the state of students' knowledge concepts in a hidden state. DKVMN uses two matrices, key and value, to learn the relationship between basic

concepts so as to directly output students' mastery of each knowledge concept [6]. SAKT, a self-attention based approach, split the entire sequence into shorter segments and train the model within each segment. But it ignores all information from previous segments [10]. Objective problems are defined as problems whose answers are fixed. So the score that a student gets in one objective problem is 0 or 1. Subjective problems refer to problems that require students to organize materials and express in a suitable way. When the teacher gives grades, they should give different grades according to the answers' quality, not just full marks or zero. Previous models can not deal with subjective problems, and the performance is somewhat poor when the input exercise sequences are long.

In this paper, we propose a model named KT-XL, which based on Transformer-XL (eXtra Long). Firstly, the input sequence is embedded in the embedding layer. Then the embedded sequence passes through a self-attention layer. It then passes through the residual layer, then enters the feed-forward layer to predict the student's performance. As we show later, KT-XL use the attention mechanism to assign weights to the previously answered exercises when predicting the performance on a particular exercise. Extensive experiments demonstrate that KT-XL gains a performance improvement of 3.6%, compared with state of the art models on average. The main contributions of this paper are as follows:

- To the best of our knowledge, KT-XL is the first knowledge tracing model which can deal with subjective problems.
- We introduce a recurrence mechanism to the model to capture longer-term dependency, achieving better performance on both short and long sequences. For long sequences, the hidden state sequence computed for the previous segment is fixed and cached to be reused as an extended context when the model processes the next new segment. As a result, KT-XL can better use the information from the previous segments after segmentation, leading to the better results.

2 Related Work

Traditional knowledge tracing models are based on Markov model, such as BKT [8]. BKT can predict the next state based on the state transition matrix, so as to predict students' performance of the next exercise based on the current state. The parameters can be solved by using Expectation-Maximization algorithm or Brute-Force algorithm.

Although BKT can output students' mastery of certain predefined concepts, it lacks the ability to extract undefined concepts and simulate transitions between complex knowledge concepts. RNN is a time series model. RNN can use early information to make predictions. Therefore, Piech applied RNN to the field of knowledge tracing and achieved good results. This is the first Deep Knowledge Tracing model, which named DKT. DKT uses RNN to solve the problem of BKT. It summarizes the state of students' knowledge concepts in a hidden state [9].

Yeung noticed two major problems of DKT, the reconstruction problem of input sequences and the volatility of prediction results. As a result, they proposed a new model to improve DKT, which named DKT+ [5]. Montero compares differences between DKT and BKT to analyze the reasons why DKT can obtain better results [11]. Xiong compared DKT with Performance Factor Analysis (PFA) and BKT, and obtained three reasons to explain why DKT could achieve best result [12]. Yeung used comprehensive features of students on the basis of DKT and DKT+, and the performance was better than the basic models [13].

PDKT-C is an improvement of DKT that considers the knowledge structure information, especially the prerequisite relations between pedagogical concepts. This model achieves a significant performance improvement by comparing with other three knowledge tracing models [14]. DKVMN uses relationship between basic concepts to output students' mastery of each knowledge concept. A static matrix is used to store knowledge concepts, and a dynamic matrix is used to store and update the mastery of corresponding knowledge concepts [6]. SAKT is a self-attention based approach and perform well while dealing with sparse data. It split the entire sequence into shorter segments of manageable sizes, and train the model within each segment. But it ignores all information from previous segments [10].

Previous models can not deal with subjective problems, and the performance is somewhat poor when input exercise sequences are long. In this paper, we propose a knowledge tracing model for subjective problems based on Transformer-XL, which can predict students' performance of a specific exercise. This model can deal with subjective problems. Besides, in our model, we split the entire sequence into shorter segments and reuse the information from previous segments, achieving better performance on both short and long sequences.

Table 1: Notations

Notations	Description
x_i	i-th exercise-answer pair of a student
l	length of segment after segmentation
n	length of sequence
$\mathbf{B}$	interaction embedding matrix
$\mathbf{P}$	positional embedding matrix
$\hat{\mathbf{B}}$	embedded interaction matrix
$\hat{\mathbf{E}}$	embedded exercise matrix
$\mathbf{Q}, \mathbf{K}, \mathbf{V}$	the query, key and value matrices in self-attention layer
$\mathbf{M}$	the result matrix of multi-head self-attention layer
$\mathbf{F}$	the result matrix of Feed-Forward layer
y_i	the predictive score that student get in the i-th exercise
J	the number of identical layers in the model

3 Model Introduction

The task of knowledge tracing can be formalized as: given observations of interactions $x_0, x_1, x_2, \ldots\ldots x_t$ taken by a student on a particular learning task, predict aspects of their next interaction x_{t+1} [7].In the most ubiquitous instantiation of knowledge tracing, interactions take the form of a tuple of $x_t = \{e_t, a_t\}$ that combines a tag for the exercise being answered q_t with the score a_t. When making a prediction, the model is provided the tag of the exercise being answered e_t and must predict the score a_t [9].

We can transform this problem into a sequential modeling problem. The input of the model is the exercise interaction $x_0, x_1, x_2, \ldots\ldots x_t$ and the tag of the next exercise $e_1, e_2, e_3 \ldots\ldots e_t$, and the output is the student's score $a_1, a_2, a_3 \ldots\ldots a_t$. It is convenient to convert x_i into a number $u_i = e_i \times (a_{max} + 1) - a_i$, where a_{max} is the maximum score of the exercise. Therefore, the number of values that u_i can take is $l_u = (a_{max} + 1) \times l$.

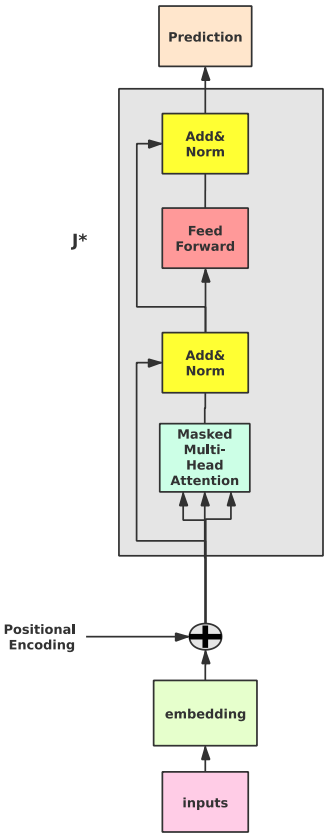

Figure 1: The architecture of KT-XL

The architecture of KT-XL is shown in Figure 1.And we will describe the layers of our KT-XL in the following parts. For easy understanding, Table 1 summarizes some mathematical notations used in the paper.

Embedding and Relative Position Encoding Layer : In our model, we split the entire sequence into shorter segments and reuse the information from previous segments. Firstly, we split the obtained input sequence into shorter segments whose length equals l. Secondly, we train an interaction embedding matrix $\mathbf{B} \in \mathbb{R}^{l_u \times d}$,where $\mathbf{B}_{x_i}$ represent the embedding of interaction x_i. And we train an exercise embedding matrix, $\mathbf{E} \in \mathbb{R}^{l \times d}$, where d is the hidden dimension. Thirdly, instead of incorporating bias statically into the initial embedding, we can inject the same information into the attention score of each layer. Following Dai, we create a set of relative positional encodings $\mathbf{P} \in \mathbb{R}^{l \times d}$, where the i-th row $\mathbf{P}_i$ indicates a relative distance of i between two positions [15]. Finally, the output of embedding and relative position encoding layer is embedded interaction matrix and embedded exercise matrix.

$$\widehat{\mathbf{B}} = \begin{bmatrix} \mathbf{B}_{x_1} + \mathbf{P}_1 \\ \mathbf{B}_{x_2} + \mathbf{P}_2 \\ \ldots \\ \mathbf{B}_{x_n} + \mathbf{P}_l \end{bmatrix}, \widehat{\mathbf{E}} = \begin{bmatrix} \mathbf{E}_{x_1} \\ \mathbf{E}_{x_2} \\ \ldots \\ \mathbf{E}_{x_l} \end{bmatrix} \quad (1)$$

Masked Multi-head Attention: In our model, we use the masked scaled dot-product attention mechanism [15]. An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key. This layer can find the relative weight between previous exercises and the next exercise.

We get $\mathbf{q}_i, \mathbf{k}_i, \mathbf{v}_i$ by the following equations:

$$\mathbf{q}_i = \widehat{\mathbf{E}}_{i+1}\mathbf{W}_i^{\mathrm{Q}},\ \mathbf{k}_i = \widehat{\mathbf{B}}_i\mathbf{W}_i^{\mathrm{K}},\ \mathbf{v}_i = \widehat{\mathbf{B}}_i\mathbf{W}_i^{\mathrm{V}} \quad (2)$$

where $\mathbf{W}^{\mathrm{Q}}, \mathbf{W}^{\mathrm{K}}, \mathbf{W}^{\mathrm{V}} \in \mathbb{R}^{d \times d}$ are the query, key and value projection matrices.

In our model, the answer depends only on interactions at positions less than t when predicting a_t. Therefore, for a query $\mathbf{q}_i$, the keys $\mathbf{k}_j$ such that $j > i$ should not be considered. We implement this by masking out the values in the matrix.

In practice, we compute the attention function on a set of queries simultaneously, packed together into a matrix $\mathbf{Q}$. The keys and values are also packed together into matrices $\mathbf{K}$ and $\mathbf{V}$. We compute the matrix of outputs as:

$$\mathrm{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^{\mathrm{T}}}{\sqrt{d}}\right)\mathbf{V} \quad (3)$$

Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions. Following Vaswani [16], we then perform the attention function in parallel and the result we get is matrix $\mathbf{M}$.

Feed-Forward Layer: In addition to attention layer, we use a position-wise feed-forward layer to incorporate nonlinearity in the model, which is applied to each position separately and identically.

This consists of two linear transformations with a ReLU activation in between.

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)\,W_2 + b_2 \quad (4)$$

While the linear transformations are the same across different positions, they use different parameters from layer to layer. And we get the result matrix **F** .

Recurrence Mechanism: To address the limitations of using a fixed-length context, we introduce a recurrence mechanism to the model to capture longer-term dependency, achieving better performance on both short and long sequences.

During training, the hidden state sequence computed for the previous segment is fixed and cached to be reused as an extended context when the model processes the next new segment.

Equipping the recurrence mechanism with our proposed relative positional embedding, we finally arrive at the KT-XL architecture. For completeness, we summarize the computational procedure for a J-layer KT-XL with a single attention head here. For $j = 1,2,3 \ldots J$.

$$\tilde{\mathbf{h}}_{t+1}^{j-1} = \left[\mathrm{SG}(\mathbf{h}_t^{j-1}) \circ \mathbf{h}_{t+1}^{j-1}\right] , \quad (5)$$

$$\mathbf{q'}_{t+1}^{j}, \mathbf{k'}_{t+1}^{j}, \mathbf{v'}_{t+1}^{j} = \mathbf{h}_{t+1}^{j-1}\mathbf{W}_q^T, \tilde{\mathbf{h}}_{t+1}^{j-1}\mathbf{W}_k^T, \tilde{\mathbf{h}}_{t+1}^{j-1}\mathbf{W}_v^T, \quad (6)$$

$$\mathbf{h}_{t+1}^{j} = \mathrm{Layer}(\mathbf{q'}_{t+1}^{j}, \mathbf{k'}_{t+1}^{j}, \mathbf{v'}_{t+1}^{j}) \quad (7)$$

Where $\mathbf{h}_t^j$ stands for the j-th layer hidden state sequence produced for the t-th segment. The function $\mathrm{SG}(\cdot)$ stands for stop-gradient, the notation $[\mathbf{h}_u \circ \mathbf{h}_u]$ indicates the concatenation of two hidden sequences along the length dimension, and W denotes model parameters. Layer( ) represents the self-attention layer and Feed-Forward layer. With $\mathbf{h}_t^0 := \mathbf{B}_{x_t}$ defined as the interaction embedding sequence.

Prediction Layer: Besides, we employ a residual connection [17] between self-attention layer and feed-forward layer, followed by layer normalization [18].

Then we predict the score of objective problem by the following equations:

$$\mathbf{T}_i = \frac{\sum_0^{i-1} \mathbf{F}_x}{i-1}, \quad (8)$$

$$y_i = \mathrm{Sigmoid}((\mathbf{T}_i, \mathbf{F}_i)\mathbf{W}_3 + b_3) \quad (9)$$

where y_i is the predictive score that student gets in the next objective problem e_i, $\mathbf{F}_i$ is the i-th row of **F**.

When predicting the score of subjective problem, we use different equations:

$$\mathbf{T}_i = \frac{\sum_0^{i-1} \mathbf{F}_x}{i-1}, \quad (10)$$

$$y_i' = (\mathbf{T}_i, \mathbf{F}_i)\mathbf{W}_3 + b_3 \quad (11)$$

where y_i' is the predictive score that student gets in the next subjective problem e_i.

Network Training: We train the model by using two different loss function. Predicting students' performance of the objective problems can be seen as a binary classification problem. And we use L_1 to tune parameters. Predicting students' performance of the subjective problems can be seen as a regression problem. And we use L_2 to tune parameters.

$$L_1 = -\sum_t (a_t log(y_t) + (1-a_t) log\,(1-y_t)) \quad (12)$$

$$L_2 = \sum_t (y_t - a_t)^2 \quad (13)$$

4 Experiments

We conduct experiments on three real-world data sets and two synthetic data sets.

ASSIST is collected from the ASSISTments blended learning platform. It is an ITS that provides students with mathematical problems, combined with their school curriculum. When a student answers a question on ASSISTments, the system generates a click-stream record about the learning behavior of the student in several educational aspects [13]. AI-CM and AI-PH are collected from Smart Learner Platform. The platform serves more than 4,000 students at 31 local schools [14].S1 and S2 are synthetic data sets. We simulated 100,000 question-answering interactions of 1,000 student responses for 100 questions.

The relevant descriptions of the data sets are shown in Table 2. In Table 2, the "Average records/student" row shows the average number of question each student answers, which is between 33 and 100. Another row "Question coverage/student" describes the average percentage of questions each student answers, which is defined as "Question records/students" divides "# of questions". As shown in the table, this value varies from 7% to 26% in the real-world data sets. This low coverage percentage reflects the data sparseness issue on knowledge tracing.

Table 2: Data Sets Statistics

Statistics	ASSIST	AI-CM	AI-PH	S1	S2
# of records	526,163	125,726	67,874	100,000	100,000
# of students	15,931	2,648	964	1000	1000
Average records/student	33.0	47.5	70.4	100	100
# of questions	124	655	483	100	100
Question coverage/student	0.26	0.07	0.15	1	1
Full score	1	8	6	1	5

Metrics: The objective of KT-XL is to predict the score of students for each exercise. Predicting students' performance of the objective problems can be seen as a binary classification problem. And we use Area Under Curve (AUC) as metric. Predicting the students' performance of the subjective problems can be seen as a regression problem. And we use R-square as metric.

We compare our model with DKT, DKT+,DKVMN and SAKT. These methods are described in the related work.In order to verify the validity of KT-XL, we performed experiments on the PEP task, which is to predict the score of examinees for each problem.For all the methods, we tried the hidden dimension $d = \{50, 100, 150, 200\}$. For the other methods, we used the same hyper-parameters as their respective paper. We implemented KT-XL with pytorch and the learning rate is 0.001. The batch size is 128. We used a dropout rate of 0.2. We set the maximum length of the sequence $L = \{10, 25, 50, 100\}$.

Table 3: PEP task comparison

Statistics	DKT	DKT+	DKVMN	SAKT	KT-XL	Gain%
ASSIST	0.784	0.812	0.808	0.818	0.823	0.6
AI-CM	0.744	0.751	0.761	0.763	0.791	3.7
AI-PH	0.689	0.711	0.718	0.716	0.756	5.3
S1	0.782	0.789	0.771	0.814	0.834	2.5
S2	0.728	0.731	0.755	0.769	0.815	6.0
Average	0.745	0.758	0.762	0.776	0.803	3.6

From Table 3, we can observe that KT-XL performs best over all the data sets. Specifically, the effect has been improved by 3.6% on average, compared with state of the art models. We attribute this gain to the introduction of attention mechanism and recurrence mechanism which aim to capture long-term dependencies. Especially in AI-PH, the average length of interaction sequence in this data set is the largest over all data sets, so the effect of the recurrence mechanism is more obvious.

In order to verify the effect of the recurrence mechanism we introduced, we conducted a comparative experiment. By removing the recurrence mechanism of KT-XL, we obtain a new model named KT-XL'. We conducted experiments on all data sets, setting the maximum length of the sequence $L = \{10, 25, 50, 100\}$.

Table 4: Effect of the recurrence mechanism

	L=10		L=25		L=50		L=100	
	KT-XL	KT-XL'	KT-XL	KT-XL'	KT-XL	KT-XL'	KT-XL	KT-XL'
ASSIST	0.818	0.801	0.823	0.804	0.789	0.770	0.782	0.770
AI-CM	0.788	0.754	0.780	0.763	0.783	0.772	0.791	0.783
AI-PH	0.721	0.716	0.730	0.721	0.744	0.738	0.752	0.749
S1	0.812	0.805	0.821	0.811	0.824	0.815	0.799	0.797
S2	0.788	0.773	0.792	0.784	0.815	0.786	0.765	0.765

It can be seen from Table 4 that the introduction of the recurrence mechanism has an important effect on the improvement of the model's performance in PEP task.

Conclusion

In this paper, we propose a knowledge tracing model for subjective problems based on Transformer-XL, which can predict students' performance of a specific exercise. Besides, for a long input sequence, the information of previous segments can be better utilized after segmentation. Extensive experimentation on a variety of real-world data sets shows that our model can perform better in predicting students' performance. In future, we will apply this model to programming education, which is one of the most important and basic parts in Computer Education.

ACKNOWLEDGMENTS

This work is supported by Youth Innovation Promotion Association of CAS, and Anhui Provincial Major Teaching Reform Research Project (2019jyxm0001).

REFERENCES

[1] C. Kulkarni, M. S. Bernstein, and S. Klemmer. Peerstudio: Rapid peer feedback emphasizes revision and improves performance. 2015.

[2] Yi-Min L , Wang-Ming X , Jin W , et al. The design and implementation of an online teaching system based on B/S structure and ASP technology[C]// International Conference on E-health Networking. IEEE, 2010.

[3] Gong Y , Beck J E , Heffernan N T . Comparing Knowledge Tracing and Performance Factor Analysis by Using Multiple Model Fitting Procedures[C]// Intelligent Tutoring Systems, 10th International Conference, ITS 2010, Pittsburgh, PA, USA, June 14-18, 2010, Proceedings, Part I. Springer Berlin Heidelberg, 2010.

[4] Cai Y, Niu Z, Wang Y, et al. Learning Trend Analysis and Prediction Based on Knowledge Tracing and Regression Analysis[C]//International Conference on Database Systems for Advanced Applications. Springer, Cham, 2015: 29-41.

[5] Yeung C K , Yeung D Y . Addressing two problems in deep knowledge tracing via prediction-consistent regularization[C]// the Fifth Annual ACM Conference. ACM, 2018.

[6] Zhang J, Shi X, King I, et al. Dynamic key-value memory networks for knowledge tracing[C]//Proceedings of the 26th international conference on World Wide Web. 2017: 765-774.

[7] Corbett A T , Anderson J R . Knowledge tracing: Modeling the acquisition of procedural knowledge[J]. User Modeling and User-Adapted Interaction, 4(4):253-278.

[8] JE Rollinson . From Predictive Models to Instructional Policies.[J]. International Educational Data Mining Society, 2015.

[9] Piech C, Bassen J, Huang J, et al. Deep knowledge tracing[C]//Advances in neural information processing systems. 2015: 505-513.

[10] Pandey S, Karypis G. A Self-Attentive model for Knowledge Tracing[J]. arXiv preprint arXiv:1907.06837, 2019.

[11] Montero S, Arora A, Kelly S, et al. Does Deep Knowledge Tracing Model Interactions among Skills?[J]. International Educational Data Mining Society, 2018.

[12] Xiong X, Zhao S, Van Inwegen E G, et al. Going deeper with deep knowledge tracing[J]. International Educational Data Mining Society, 2016.

[13] Yeung C K, Yeung D Y. Incorporating features learned by an enhanced deep knowledge tracing model for stem/non-stem job prediction[J]. International Journal of Artificial Intelligence in Education, 2018: 1-25.

[14] Chen P, Lu Y, Zheng V W, et al. Prerequisite-driven deep knowledge tracing[C]//2018 IEEE International Conference on Data Mining (ICDM). IEEE, 2018: 39-48.

[15] Dai Z, Yang Z, Yang Y, et al. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context[C]//Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019: 2978-2988.

[16] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[C]//Advances in neural information processing systems. 2017: 5998-6008.

[17] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

[18] Ba J L, Kiros J R, Hinton G E. Layer normalization[J]. arXiv preprint arXiv:1607.06450, 2016.