Journal of Network Intelligence ©2025 ISSN 2414-8105
Taiwan Ubiquitous Information Volume **10**, Number **2**, May **2025**

Research on Deep Learning Based Student Knowledge Tracking Algorithm

Hong-Li Wang*
Nanchong Vocational and Technical College, Nanchong 637000, P. R. China
19938525553@163.com

Xiao-Dong Wang
Nanchong Vocational and Technical College, Nanchong 637000, P. R. China
15181796035@163.com

Qin-Zhong Tsai
Nakhon Phanom University, Nakhon Phanom 48000, Thailand
bz0241@163.com

*Corresponding author: Hong-Li Wang

Received April 15, 2024; revised August 31, 2024; accepted December 9, 2024.

Abstract. *The development of artificial intelligence and the rise of online education have accelerated the process of intelligent educatization, and knowledge tracking is one of the most fundamental and important tasks, which dynamically assesses students' knowledge mastery level based on their historical question answering. A large amount of work has been done to explore knowledge tracking, but there are still some problems that need to be solved, such as most of the existing methods do not take into account the individual student's learning ability, seldom make use of global information when learning topic representations, and the prediction of students' answer performance needs to take into account the long term dependency in the question-answer sequence. In this paper, we address the above issues and propose a knowledge tracking approach based on students' learning abilities. Individual learning ability is an important factor in the learning process, which influences students' question-answering performance and knowledge absorption. The method proposed in this paper will model students' knowledge status and learning ability at the same time, so that students' learning process can be analyzed from both knowledge and ability perspectives. The method first calculates students' learning ability on specific knowledge points based on historical question records, and designs a sequential neural network that incorporates learning ability as an important factor in the process of acquiring and updating students' knowledge states. Second, the knowledge states related to the topics to be predicted are extracted using an encoder-decoder structure. Finally, the two information of knowledge state and learning ability are combined to predict the students' performance in answering the next questions.*

Keywords: Knowledge tracking; Intelligent education; Graph neural networks; Self-attention mechanism

1. Introduction.

The development of information technology is gradually changing the way people learn, from the traditional campus classroom learning in the past to the many online education platforms that have emerged today, such as China University MOOC, Coursera, Tencent Classroom, etc., students can choose what they want to learn independently anytime they want to learn on the Internet. The outbreak of the COVID-19 pandemic in early 2020 had a great impact on the traditional offline teaching mode. Students were forced to stay at home and learn through online education platforms, which further highlighted the importance of online education now growing at an unprecedented scale and speed. Online education can be unrestricted by time and space, and supports unified teaching of all students. Moreover, online platforms offer a wealth of learning resources and diverse teaching methods, and with the help of artificial intelligence technology can provide smarter educational services, such as learning-resource recommendation [1], learning-path recommendation and adaptive learning, to achieve personalized learning for different students. The key to this advantage lies in the need to analyze learners' historical learning data and dynamically assess students' knowledge mastery level, which is one of the most basic and critical research problems in the field of intelligent education, known as knowledge tracking.

With the rapid development of deep learning in recent years, the field of knowledge tracking has achieved many results, but human learning is a complex process and there remain challenges to explore, specifically: (1) The complexity of students' knowledge-state changes. Knowledge-state evolution during learning is affected by many factors, such as current knowledge level, psychological state, and knowledge acceptance ability [2]. Some students may learn quickly with few practices, while others rely on strong memory to retain knowledge over time. Most existing models do not account for individual differences when modeling knowledge-state changes. Capturing personalized characteristics from question sequences or additional data and integrating them into the knowledge-state update process remains a difficulty in knowledge tracking. (2) Characterization of topics and knowledge points under limited conditions. Topic and knowledge-point representations are critical in knowledge tracking, yet most datasets contain only topic identifiers and lack rich information such as topic text or hierarchical relationships. When the number of topics is large, student interactions per topic may be sparse, increasing the difficulty of learning effective representations. Although many models focus on knowledge-state modeling, enhancing topic representations has been shown to improve performance, making topic representation another key challenge. (3) Sequential dependencies in students' question sequences. Learning is a step-by-step process: students gradually improve mastery through continuous practice, and past question records exert varying influence on current learning. Modeling these long-term dependencies becomes increasingly difficult as the history of interactions grows.

1.1. **Related work.** In 1995, Corbett and Anderson [3] proposed Bayesian Knowledge Tracing (BKT), essentially a Hidden Markov Model with two latent states (mastery/non-mastery) and two observations (correct/incorrect). BKT models four parameters: (1) prior mastery probability; (2) learning transition probability; (3) guess probability; and (4) slip probability. Zhang and Yao [4] extended BKT to three latent states by adding a "possible mastery" state, and Wang et al. [5] applied clustering to group students by learning ability. Yang and Cheung [6] combined decision trees with Deep Knowledge Tracing (DKT), selecting features such as response time and attempt count before inputting to an RNN. Song et al. [7] designed a two-layer contrastive learning scheme on an exercise-to-exercise subgraph and a concept subgraph. Liu et al. [8] introduced a Python-based benchmark for fair comparison of DLKT methods. Song et al. [9] proposed Joint Graph Convolutional Network (JKT) to fuse exercise-to-exercise and concept-to-concept relations.

With the advent of attention mechanisms, new models have emerged. Liu et al. [10] proposed Ability Enhancement Knowledge Tracking (ABKT), incorporating ability factors into feedback attribution. Gan et al. [11] introduced a KS-enhanced graph representation model with attention (KSGKT). Ni et al. [12] developed HHSKT to trace short-term attentional knowledge via hierarchical heterogeneous structures. Zhao et al. [13] presented MQFKT, exploiting multiple question factors. Graph neural networks [14] have also been applied: Hooshyar et al. [15] proposed GameDKT integrating gameplay data; Asselman et al. [16] developed a PFA approach using ensemble learners; Wu et al. [17] introduced session graph-based KT (SGKT); Huang et al. [18] studied diagnostic KT; Gan et al. [19] presented KTM-DLF modeling cognitive item difficulty; and Song et al. [20] proposed bi-graph contrastive learning. Inspired by knowledge-transfer theory [21], Lu et al. [22] developed CMKT using concept maps.

1.2. **Motivation and contribution.**

Current research on knowledge tracking focuses mainly on knowledge-state estimation but does not effectively evaluate students' learning abilities. This paper addresses this gap by modeling both knowledge state and individual learning ability. Specifically, we first compute each student's learning ability on a knowledge point based on recent question records, and design a sequential neural network incorporating this ability into knowledge acquisition and update. Next, we extract topic-related knowledge states using a self-attention-based encoder-decoder. Finally, we combine knowledge state and learning ability to predict students' performance on the next question.

2. **Relevant theoretical analysis.**

2.1. **Cognitive diagnosis and knowledge tracking.**

The current methods of modeling learners' knowledge state can be roughly divided into two categories: (1) Cognitive Diagnosis (DC); (2) Knowledge Tracing (KT).

Cognitive Diagnosis simulates a student's knowledge state in a static scenario (e.g., an exam), which assumes that the student's knowledge state is constant over time as he or she answers a set of questions. The student's mastery of each knowledge point is assessed by the cognitive diagnostic algorithm after the student has finished a set of questions. Related studies include DINA [23] and DINO [24].

In real life, students' knowledge states are always in the process of changing, and it becomes a challenge to capture students' knowledge states dynamically, and this type of problem is known as knowledge tracking, which models students' knowledge states in dynamic scenarios (e.g., online learning platforms). For example, each time a student finishes a topic, the knowledge tracking model computationally updates his knowledge state. Early research on knowledge tracing was dominated by Bayesian knowledge tracing, but nowadays knowledge tracing based on deep learning has become a research hotspot. DKT (Deep Knowledge Tracing) is the first proposed deep knowledge tracing model, which dynamically calculates the knowledge state of a student according to his question sequences and predicts the performance of the student's next answer, and its model architecture is shown in Figure 1, which is the first deep knowledge tracing model. As shown in Figure 1, the input of DKT is $x_t = (q_t, a_t)$, followed by the knowledge state of the student through the hidden layer of RNN. Finally, the output y_t of DKT is the probability that the student can answer the question correctly.

2.2. **Recurrent neural network.**

Recurrent Neural Network (RNN) [25] has shown great ability in dealing with sequential data with dependencies, unlike networks such as Convolutional Neural Network (CNN) [26], which assume that the elements are independent of each other and that the data's inputs and outputs are also independent, whereas an RNN associates elements, and at its core, possesses the ability to memorize. Figure 2

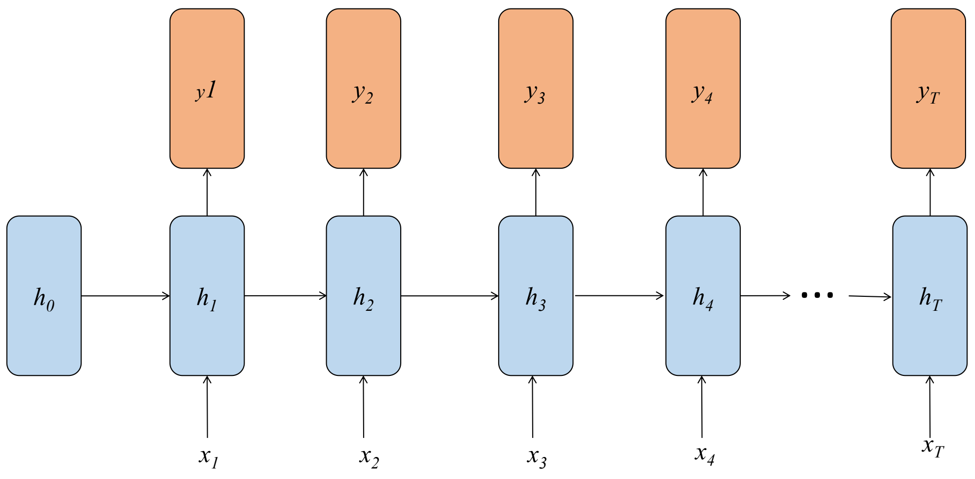

Figure 1. DKT model architecture

shows the structure of RNN, where the output of each module is adjusted based on the results of previous time steps.

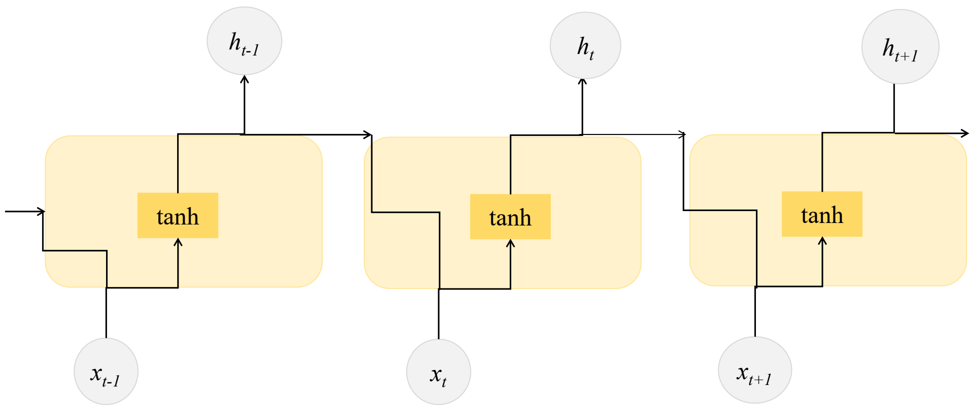

Figure 2. Recurrent neural network structure

Specifically, each cell in the RNN is computed as Equation (1):

$$h_t = \tanh\big(W[h_{t-1}, x_t] + b\big) \tag{1}$$

Although RNN can save historical information, it may encounter gradient explosion problems during training. GRU can solve this problem by using a long short-term memory network, where z_t is the update gate and r_t is the reset gate. GRU stores long-term information according to the following formulas:

$$z_t = \sigma(W_z[h_{t-1}, x_t] + b_z) \tag{2}$$

$$r_t = \sigma(W_r[h_{t-1}, x_t] + b_r) \tag{3}$$

$$\tilde{h}_t = \tanh(W_h[r_t \odot h_{t-1}, x_t] + b_h) \tag{4}$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \tag{5}$$

2.3. **Memory network.** Memory Network (MN) is also one of the commonly used models in knowledge tracking. Although models such as RNN, LSTM and GRU have memory functions, they all use only a single vector to store memorized information, while Memory Network memorizes different information through multiple vector slots.

The commonly used memory network in knowledge tracking is the key-value memory network, which consists of two parts, the key matrix and the value matrix, the key matrix corresponds to the vectors in the value matrix. The key matrix is used for addressing, while the value matrix stores the memorized information for data reading or updating. For an input, the correlation between it and each candidate memory is first calculated by the key matrix with the formula:

$$p_i = softmax(qK_i) \tag{6}$$

where K_i is the i-th vector in the key matrix. Finally, the value matrix is multiplied with its corresponding weights to complete the data reading, which is given by Equation (7):

$$o = \sum_i p_i V_i \tag{7}$$

where V_i is the i-th vector in the value matrix.

2.4. **Attention mechanism.** Currently, the use of attention mechanism is almost indispensable in each model, it was first proposed in the field of computer vision, and then the Google mind team combined the attention mechanism and RNN [27] on the task of image categorization, so that the attention mechanism began to develop rapidly.

3. Knowledge tracking based on student learning competencies.

3.1. **Problem analysis.** It has been found that most of the current knowledge tracking models only feedback the students' question records as the degree of knowledge mastery, and predict the next question performance based on the current degree of knowledge mastery. In reality, whether a student can answer a question correctly is not only related to his current knowledge status, but also related to his learning ability.

In addition, students' learning ability not only affects whether they can do the questions correctly, but also affects their knowledge acquisition. Different students should have different levels of knowledge acquisition after doing the same topic. Early models simply assumed that students would acquire the same knowledge after doing a topic, so some models in recent years have begun to improve the knowledge updating process.

For modeling students' knowledge state, most papers are based on RNN or self-attention mechanism. RNN can capture the sequential relationships in the sequence of students' questions, but its capture of long-term dependencies is affected by the length. And the self-attention mechanism can break through the limitation of length to capture important information. In the knowledge tracking problem, the students' question records undoubtedly have sequential relationships, so the knowledge tracking model based on the self-attention mechanism often needs to add additional sequential information.

The main idea of LAKT is to compute students' learning ability on specific knowledge points from the history of doing problems, and we design a Sequential Neural Network (LASNN) that incorporates Learning Ability as an important factor in students' knowledge state acquisition and updating process. Then, we use an encoder-decoder structure based on the self-attention mechanism to extract the knowledge states related to the topics to be predicted. In addition, in order to incorporate sequential information into the self-attention mechanism, we adopt a different approach from previous papers, i.e., we use GRU to capture the sequential information in the sequence of students' questions.

Finally, we combine both information of students' knowledge state and learning ability to predict their performance in answering the next questions.

3.2. **Model design.**

3.2.1. *Model concepts.* The knowledge and ability-based knowledge tracking model can be divided into the following parts:

(1) Coding layer: coding the questions, question-answer pairs, and filtering the last k practice records with the same knowledge point from the history of doing the questions, and calculating the student's learning ability on the knowledge point.

(2) Knowledge state layer: firstly, the learning ability is integrated into the process of students' knowledge acquisition and knowledge updating by means of a sequential neural network (LASNN) incorporating the learning ability; then, the encoder is used to further model the students' knowledge states at different time steps; finally, in the decoder, for the topic to be predicted, the knowledge state related to the topic is extracted from the encoder.

(3) Prediction Layer: combines the students' knowledge state and learning ability to predict the students' performance in answering questions at the next moment.

3.2.2. *Coding layer.* Due to the sparsity of the data, some existing models use knowledge points instead of questions as inputs to the model, however, this ignores the differences between questions that contain the same knowledge points. In order to better associate topics with knowledge points, the difficulty vector of topics is introduced and the embedding of the topics q_t done by students at time t is defined as:

$$q_t = c_t + \mu_t \odot d_t \tag{8}$$

where $\odot$ is the Hadamard product, c_t is the embedding of the knowledge point corresponding to topic q_t, μ_t is the difficulty vector of topic q_t, and d_t is the change vector of the knowledge point corresponding to topic q_t.

Similar to question coding, we define the embedding of a student's question answered correctly at time t as:

$$a_t = e_t + \mu_t \odot f_t \tag{9}$$

where e_t is the embedding of the knowledge response to (c_t, r_t), μ_t is the difficulty vector of the question q_t, and f_t is the change vector of the knowledge response to (c_t, r_t).

If a student can get more difficult questions correct, the student's ability to learn will be higher. Inspired by Rasch's model and taking into account that students have different learning abilities at different knowledge points, when predicting a student's performance on a question q_t at time t, we calculate a student's learning ability on the knowledge point c_t based on the history of the student's performance on the knowledge point c_t according to the knowledge point c_t that corresponds to the question q_t. Since exercises that are too far back in time may introduce additional noise, we only select the k closest question-answering records with the same knowledge point c_t from the current moment, denoted as:

$$\tilde{x} = \{\tilde{a}_{t_1}, \tilde{a}_{t_2}, ..., \tilde{a}_{t_k}\} \tag{10}$$

Intuitively, if a student is able to solve difficult or increasingly difficult topics with a smaller number of errors, there is a high probability that he has a greater ability to learn.

To capture a student's learning ability, we input $\tilde{x}$ into the GRU:

$$p_{t'} = GRU\big((\mu_{t'} \oplus r_{t'}), p_{t'-1}\big) \tag{11}$$

where $\oplus$ is the splicing operation, $\mu_{t'}$ is the difficulty vector of the topic, and $r_{t'}$ is either an all-1 vector or an all-0 vector of the same dimension as $\mu_{t'}$. Eventually, we will get k output vectors, taking p_k as the student's learning ability m_t at moment t:

$$m_t = \tanh(p_k) \tag{12}$$

3.2.3. *Knowledge state layer.* The knowledge state layer consists of two parts: the LASNN and the encoder-decoder. Among them, LASNN is a sequential neural network that we designed to integrate learning ability, through which the module incorporates the factor of students' learning ability into the process of students' knowledge state change, and its structure is shown in Figure 3, which is divided into three parts: (1) subjective difficulty, (2) knowledge acquisition, and (3) knowledge updating. Secondly the encoder decoder calculates the correlation between questions and question-answer pairs through a self-attention mechanism.

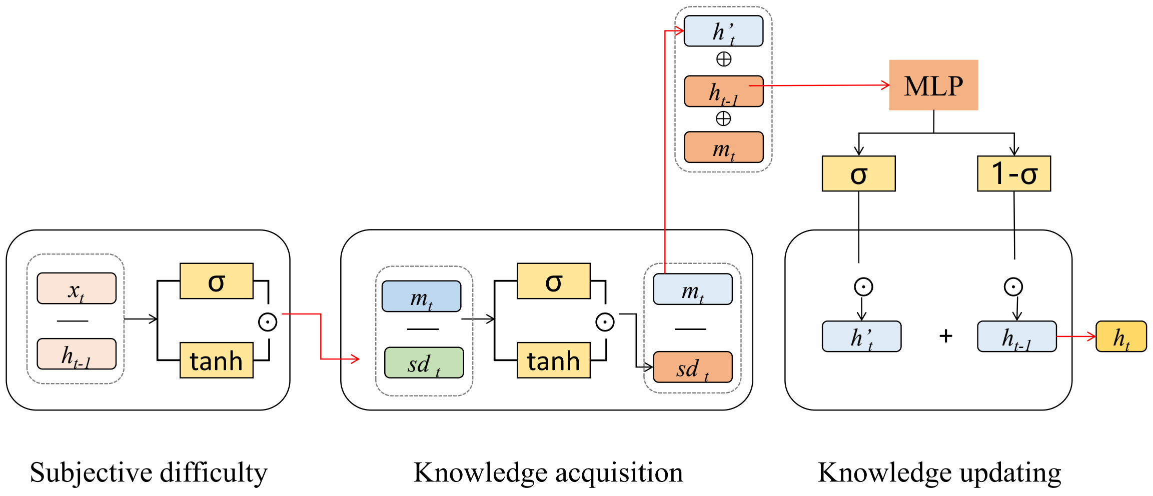

Figure 3. LASNN module architecture

Previously, we defined the objective difficulty of each topic, however, each student's perception of the difficulty of the topic is different, specifically, if the student's knowledge mastery level has reached the requirements of the topic, he will find the topic relatively easy, while for those who have not reached the requirements of the topic, they will accordingly find the topic to be difficult. We subtract the projected topic embedding q_t from the student's previous state of knowledge h_{t-1}, and use the difference between the two as a measure of the subjective difficulty of the topic:

$$x_t = W_1^T q_t + b_1 \tag{13}$$

$$\tilde{sd}_t = \tanh\left(W_2^T (x_t - h_{t-1}) + b_2\right) \tag{14}$$

$$\Gamma_t^{sd} = \sigma\left(W_3^T (x_t - h_{t-1}) + b_3\right) \tag{15}$$

$$sd_t = \Gamma_t^{sd} \odot \tilde{sd}_t \tag{16}$$

where tanh is the tanh activation function, σ is the Sigmoid activation function; W_1^T, W_2^T, W_3^T are the weight matrices; b_1, b_2, b_3 are the bias vectors; $\tilde{sd}_t$ is the direct difference between x_t and h_{t-1}. In addition, we set up a gating vector Γ_t^{sd} for retaining or removing the information in $\tilde{sd}_t$, and sd_t is the final vector of students' subjective question difficulty that we obtain.

For questions answered correctly (q_t, r_t), the amount of knowledge actually gained varies from student to student. Comparing the students' learning ability and the subjective difficulty of the question, if the subjective difficulty of the question is within the student's ability, then the student will be able to understand the content of the question and learn more from the question, while if the subjective difficulty of the question is higher than the student's current ability, then the student will gain less from doing the question accordingly. For this reason, we subtract the student's learning ability m_t from his or her subjective difficulty sd_t for the question, and define the student's personalized knowledge acquisition in terms of the difference between the two and the embeddedness of the question response pairs:

$$\widetilde{ka_t} = \tanh\big(W_4^T(m_t - sd_t) + b_4\big) \tag{17}$$

$$\Gamma_t^{ka} = \sigma\big(W_5^T(m_t - sd_t) + b_5\big) \tag{18}$$

$$ka_t = \Gamma_t^{ka} \odot \widetilde{ka_t} \tag{19}$$

$$h_t' = ka_t \odot a_t \tag{20}$$

where *tanh* is the tanh activation function, σ is the Sigmoid activation function, $\odot$ is the Hadamard product, W_4^T, W_5^T are the weight matrices, and b_4, b_5 are the bias vectors. $\widetilde{ka_t}$ is the direct difference between m_t and sd_t, in addition we set a gating vector Γ_t^{ka} for retaining or removing the information in $\widetilde{ka_t}$, and h_t' is the knowledge that the student can acquire after this exercise.

After calculating the personalized knowledge acquisition, we need to calculate the student's knowledge state at the current moment t. We adaptively update the knowledge state by a gating mechanism that splices the student's knowledge state h_{t-1} at the previous moment, the knowledge h_t' acquired at the current moment, and the learning ability m_t at the current moment into the fully connected network:

$$\Gamma_t = \sigma\big(W_6^T(h_{t-1} \oplus h_t' \oplus m_t) + b_6\big) \tag{21}$$

$$h_t = \Gamma_t \odot h_t' + (1 - \Gamma_t) \odot h_{t-1} \tag{22}$$

where σ is activation function, $\oplus$ is the vector splicing operation, $\odot$ is the Hadamard product.

The multi-head attention mechanism is formulated as:

$$MHA(Q,K,V) = [head_1, ..., head_h]W^O \tag{23}$$

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \tag{24}$$

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{25}$$

where W^O, W_i^Q, W_i^K, W_i^V are the weight matrices and d_k is the dimension of the key-value vectors. The LASNN models the student's knowledge state at different time steps, which is further updated in the encoder next:

$$M = LN\big(Q + MHA(Q,K,V)\big) \tag{26}$$

$$H_E = LN\big(M + FFN(M)\big) \tag{27}$$

where LN is the normalization operation, FFN is the feed-forward neural network, MHA is the multi-head attention layer, and Q, K, V are outputs from LASNN. H_E is the state of the student's knowledge at different time-steps after the self-attention layer.

In the decoder, for the topic q_t to be predicted at time t, based on its relation to the historical topics $[q_1, q_2, ..., q_{t-1}]$, the knowledge states related to q_t are extracted from the encoder. Before extracting the relevant knowledge states in the encoder, the order information of the topic sequence is modeled by GRU:

$$q_t' = GRU(q_t, q_{t-1}') \tag{28}$$

$$\tilde{q}_t = q_t + q_t' \tag{29}$$

The decoder consists of two self-attentive layers with the process:

$$M_1 = LN\big(Q + MHA(Q, K, V)\big) \tag{30}$$

$$M_2 = LN\big(M_1 + MHA(M_1, M_1, H_E)\big) \tag{31}$$

$$H_D = LN\big(M_2 + FFN(M_2)\big) \tag{32}$$

where H_D is the extracted state of the student's knowledge related to the topic.

After the above steps, we obtain the student's learning ability m_t and the extracted knowledge state $\tilde{h}_{t-1}$ associated with the topic q_t at time t. In order to predict the student's

$$\hat{r}_t = \sigma\big(W_{out}^T(m_t \oplus \tilde{h}_{t-1} \oplus q_t) + b_{out}\big) \tag{33}$$

Finally, we train the model with a cross-entropy loss function:

$$L = -\sum_i \big((r_i \log \hat{r}_i) + (1 - r_i)\log(1 - \hat{r}_i)\big) \tag{34}$$

4. **Experiment.**

4.1. **Experimental data set.**
We conducted experiments on three public datasets, ASSIST2017, Junyi, and EdNet. The ASSIST2017 dataset comes from the ASSISTments data mining competition in 2017, which records data on students' learning using the ASSISTments platform during 2004–2007. The Junyi dataset comes from Junyi Academy, which records the students' doing records in 2015. Due to its large size, we randomly select 5,000 students from EdNetKT1 for our model testing.

4.2. **Evaluation criteria.**
In this experiment, AUC (Area Under Curve) was used as the evaluation criterion to analyze the experimental results. The ROC curve was used, and the larger the area under the curve, the better the performance of the model.

4.3. **Experimental design.**
This experiment is implemented based on the Pytorch deep learning framework and runs on RTX 3090 GPUs. For model training, the embedding dimensions of questions, knowledge points, question-answer pairs, and learning ability in the model are all set to 256, and the number of heads in the attention layer is set to 8. All learnable parameters are optimized by Adam.

In the process of analyzing the experimental results, we used the following models as controls:

1. DKT is the first deep learning-based knowledge tracking model which represents the student's knowledge state by means of hidden vectors in RNNs.
2. DKTQ builds on the DKT model by using questions instead of knowledge points as input to the model.
3. DKVMN introduces a dynamic key-value memory network.
4. DKVMNQ builds on DKVMN by using questions instead of knowledge points as input to the model.
5. SAKT uses a self-attention mechanism to identify interactions related to a given knowledge point from students' past question-playing interactions and predicts their knowledge mastery based on their question-playing performance on the relevant interactions.
6. AKT uses a new monotonic attentional framework to relate students' expression on the topic to be predicted to their historical problem performance.

Table 1 presents the experimental results. Specifically, the LAKT model improves the AUC values by 2.48%, 0.28% and 0.40% on the three datasets, ASSIST17, Junyi and EdNet, respectively. It can be found that the results of DKT and DKVMN models based on sequential relationship are close to each other in general, while SAKT and AKT models based on the self-attention mechanism have a larger gap in effect, which is due to the fact that SAKT is only a simple application of self-attention network, and its feature extraction ability may be insufficiently strong; and AKT model adopts the structure of encoder-decoder, indicating that this structure can be more effective. The AKT model uses the encoder-decoder structure, which effectively improves the accuracy of the prediction results, suggesting that the structure can extract relevant and useful information more effectively; our model also uses the encoder and decoder structure, and the results are better than those of AKT.

Table 1. Comparative experimental results of student answer prediction with LAKT

	ASSIST17	**Junyi**	**EdNet**
DKT	0.7252	0.7488	0.6923
DKT-Q	0.7733	0.7692	0.6720
DKVMN	0.7121	0.7514	0.6939
DKVMN-Q	0.7492	0.7784	0.7080
SAKT	0.6609	0.7478	0.6919
AKT	0.7640	0.7883	0.7311
LAKT	0.7888	0.7911	0.7351

To demonstrate the validity of each module in the model, we designed the following ablation experiment:

(1) LAKT-RLA removes student learning ability from the prediction module, i.e., only knowledge state and question information are used to predict students' answers.

(2) LAKT-RKS removes student knowledge status from the prediction module, i.e., uses only knowledge status and question information to predict student responses.

(3) LAKT-RSN removes the LASNN module and embeds the question-answer pairs directly into the input encoder.

(4) LAKT-RLG replaces the LASNN module with GRU and directly embeds the question-answer pairs into GRU.

(5) LAKTRGRU removes the GRU from the decoder part, i.e., it does not integrate the information about the order in which the students do the questions into the question embedding.

The results of the ablation experiments are shown in Table 2. (1) the AUC values of LAKTRLA with learning ability removed in the prediction module decreased by 4.51%, 0.92% and 0.23% on the three datasets, respectively, which indicates that our incorporation of the students' learning ability into the model can effectively improve the model performance; at the same time, LAKTRKS with knowledge status removed in the prediction module decreased by 0.18%, 1.77% and 1.8%, indicating that it is also necessary to use students' knowledge status in the prediction module; (2) we can find that LAKTRKS in the prediction module with the knowledge status removed decreases the AUC values on the three datasets by 0.18%, 1.77% and 1.8% respectively, which indicates that it is also necessary to use the students' knowledge status in the prediction module; (3) the AUC values of LAKTRSN in the three datasets with the LASNN module removed decreases the AUC values on the three datasets by 0.45%, 1.4% and 0.43% respectively, which indicates that our design of the LASNN module is effective; (4) the AUC values of LAKTRLG using GRU instead of LASNN decreased by 0.39%, 0.88% and 0.21% on the three datasets, which indicates that our designed LASNN module is better than the direct use of GRU; (5) the AUC values of LAKTRGRU with the GRU before removing decoder decreased by 0.37% on the three datasets. values decreased by 0.37%, 0.57%, and 0.09%, respectively, suggesting that the GRU can capture the sequential information of the students' question sequences and improve the model performance before feeding the data into the self-attention layer.

Table 2. Results of ablation experiments with LAKT

	ASSIST17	**Junyi**	**EdNet**
LAKT-RLA	0.7437	0.7819	0.7328
LAKT-RKS	0.7870	0.7734	0.7171
LAKT-RSN	0.7843	0.7771	0.7308
LAKT-RLG	0.7849	0.7823	0.7330
LAKT-RGRU	0.7851	0.7854	0.7342
LAKT	0.7888	0.7911	0.7351

In the student ability modeling part of the LAKT model, we select the k most recent question records of students with the same knowledge points, where k is a very important hyperparameter, and we design the corresponding experiments to explore the appropriate values of k. We also design the experiments to investigate the effect of k on the ASSIST17 and EdNet datasets, and observe the effect of the model on the ASSIST17 and Junyi datasets. Specifically, we make k take the value of $\{4, 6, 8, 10, 12, 14\}$ and observe the effect of the model on three datasets, namely, ASSIST17, Junyi and EdNet.

Table 3. AUC results at different k values

	ASSIST17	**Junyi**	**EdNet**
$k = 4$	0.7830	0.7905	0.7312
$k = 6$	0.7863	0.7908	0.7326
$k = 8$	0.7878	0.7911	0.7351
$k = 10$	0.7888	0.7909	0.7303
$k = 12$	0.7864	0.7905	0.7259
$k = 14$	0.7856	0.7903	0.7233

The AUC results for different k values are shown in Table 3, and we plotted Figure 4, Figure 5 and Figure 6 to show the trend of AUC. From the results, we can see that the AUCs of the models under the six experiments are generally close to each other, and the optimal k-values under the ASSIST17, Junyi, and EdNet datasets are 10, 8, and 8, respectively. It can be seen that the model may not be able to learn the useful features under small k-values, and may introduce too much noise under large k-values, since the learning record of the past will not have much effect on the present. now will not have much effect.

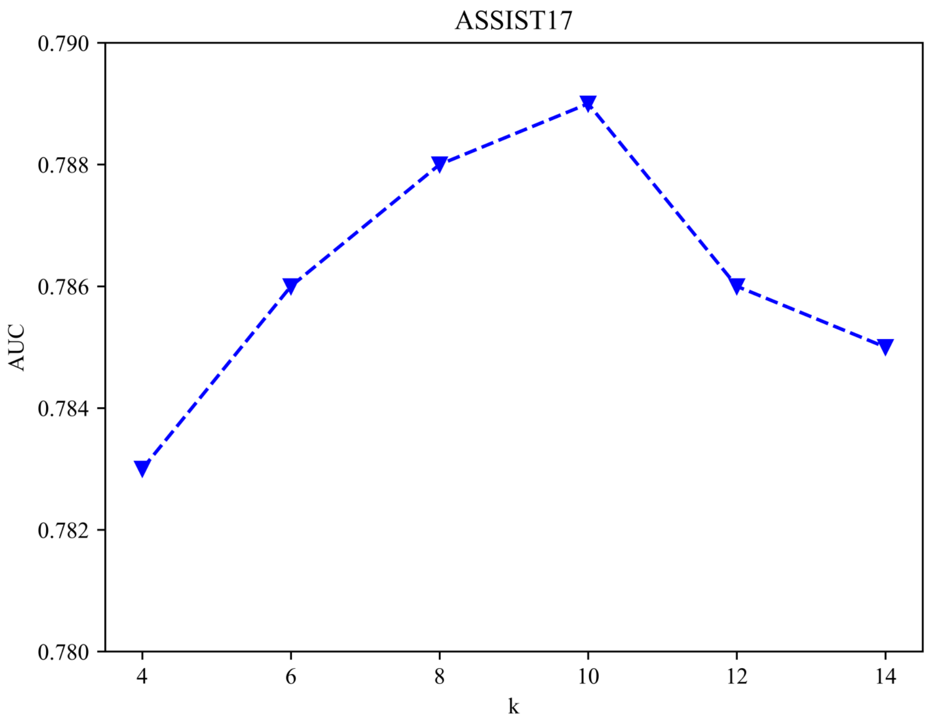

Figure 4. AUC results under different k values (ASSIST17).

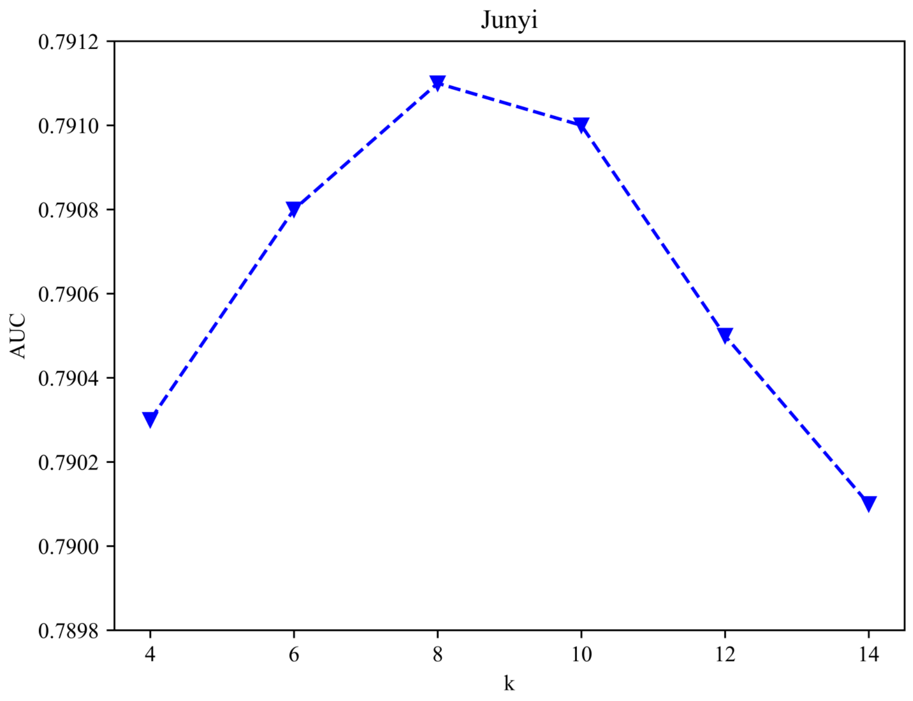

Figure 5. AUC results under different k values (Junyi).

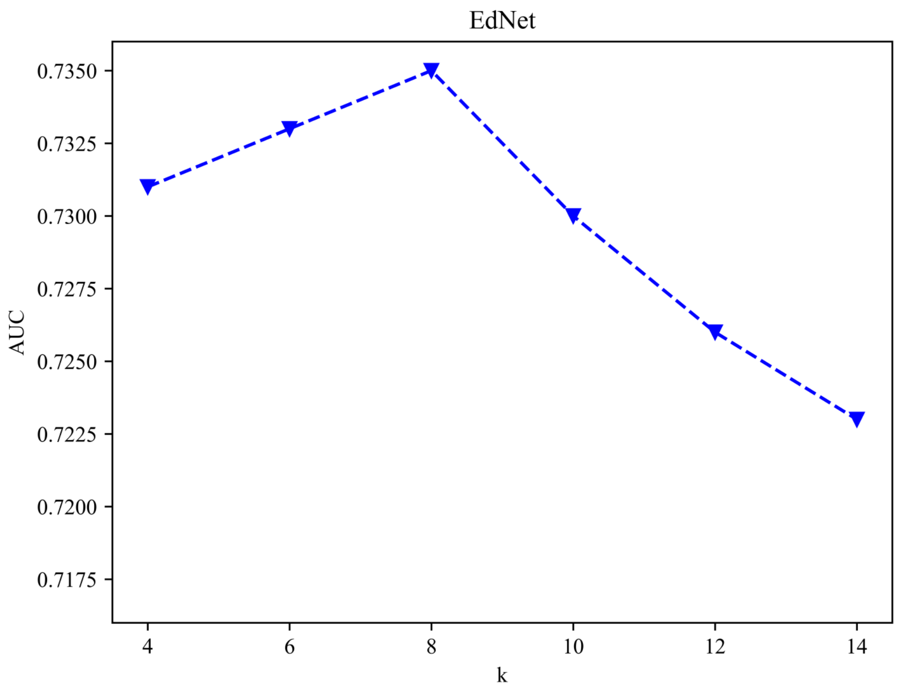

Figure 6. AUC results under different k values (EdNet).

5. **Conclusion.** In this paper, we propose a knowledge tracking model based on students' learning ability, which calculates their learning ability at the current moment on a specific knowledge point by analyzing their records of answering questions on the same knowledge point, and designs a sequential neural network to calculate the acquisition and update of the knowledge state in conjunction with the students' learning ability. Then we use an encoder–decoder structure based on the self-attention mechanism to extract the knowledge states related to the topic to be predicted. Finally, we combine both information about students' knowledge states and learning abilities to jointly predict students' question-answering performance, and experiments on multiple datasets demonstrate the effectiveness of the model. We incorporate the factor of students' learning ability in the model; however, students' personal characteristics are not limited to learning ability, and in the future, more personal factors can be considered, such as memory ability and subject interest, to further personalize learning.

REFERENCES

[1] J. Cárdenas-Cobo, A. Puris, P. Novoa-Hernández, J. A. Galindo, and D. Benavides, "Recommender systems and scratch: An integrated approach for enhancing computer programming learning," *IEEE Transactions on Learning Technologies*, vol. 13, no. 2, pp. 387–403, 2019.

[2] A. Al-Hmouz, J. Shen, R. Al-Hmouz, and J. Yan, "Modeling and simulation of an adaptive neuro-fuzzy inference system (ANFIS) for mobile learning," *IEEE Transactions on Learning Technologies*, vol. 5, no. 3, pp. 226–237, 2011.

[3] A. T. Corbett and J. R. Anderson, "Knowledge tracing: Modeling the acquisition of procedural knowledge," *User Modeling and User-Adapted Interaction*, vol. 4, pp. 253–278, 1994.

[4] K. Zhang and Y. Yao, "A three learning states Bayesian knowledge tracing model," *Knowledge-Based Systems*, vol. 148, pp. 189–201, 2018.

[5] D. Wang, Z. Zhang, J. Song, and Y. Lu, "Traditional Knowledge Tracing Models for Clustered Students," *The Educational Review, USA*, vol. 4, no. 12, pp. 244–251, 2020.

[6] H. Yang and L. P. Cheung, "Implicit heterogeneous features embedding in deep knowledge tracing," *Cognitive Computation*, vol. 10, pp. 3–14, 2018.

[7] X. Song, J. Li, Q. Lei, W. Zhao, Y. Chen, and A. Mian, "Bi-CLKT: Bi-graph contrastive learning based knowledge tracing," *Knowledge-Based Systems*, vol. 241, p. 108274, 2022.

[8] Z. Liu, Q. Liu, J. Chen, S. Huang, J. Tang, and W. Luo, "pyKT: a python library to benchmark deep learning based knowledge tracing models," in *Advances in Neural Information Processing Systems*, vol. 35, pp. 18542–18555, 2022.

[9] X. Song, J. Li, Y. Tang, T. Zhao, Y. Chen, and Z. Guan, "Jkt: A joint graph convolutional network based deep knowledge tracing," *Information Sciences*, vol. 580, pp. 510–523, 2021.

[10] S. Liu, J. Yu, Q. Li, R. Liang, Y. Zhang, X. Shen, and J. Sun, "Ability boosted knowledge tracing," *Information Sciences*, vol. 596, pp. 567–587, 2022.

[11] W. Gan, Y. Sun, and Y. Sun, "Knowledge structure enhanced graph representation learning model for attentive knowledge tracing," *International Journal of Intelligent Systems*, vol. 37, no. 3, pp. 2012–2045, 2022.

[12] Q. Ni, T. Wei, J. Zhao, L. He, and C. Zheng, "HHSKT: A learner–question interactions based heterogeneous graph neural network model for knowledge tracing," *Expert Systems with Applications*, vol. 215, p. 119334, 2023.

[13] Y. Zhao, H. Ma, W. Wang, W. Gao, F. Yang, and X. He, "Exploiting multiple question factors for knowledge tracing," *Expert Systems with Applications*, vol. 223, p. 119786, 2023.

[14] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2008.

[15] D. Hooshyar, Y.-M. Huang, and Y. Yang, "GameDKT: Deep knowledge tracing in educational games," *Expert Systems with Applications*, vol. 196, p. 116670, 2022.

[16] A. Asselman, M. Khaldi, and S. Aammou, "Enhancing the prediction of student performance based on the machine learning XGBoost algorithm," *Interactive Learning Environments*, vol. 31, no. 6, pp. 3360–3379, 2023.

[17] Z. Wu, L. Huang, Q. Huang, C. Huang, and Y. Tang, "SGKT: Session graph-based knowledge tracing for student performance prediction," *Expert Systems with Applications*, vol. 206, p. 117681, 2022.

[18] Z. Huang, Q. Liu, Y. Chen, L. Wu, K. Xiao, E. Chen, H. Ma, and G. Hu, "Learning or forgetting? a dynamic approach for tracking the knowledge proficiency of students," *ACM Transactions on Information Systems (TOIS)*, vol. 38, no. 2, pp. 1–33, 2020.

[19] W. Gan, Y. Sun, X. Peng, and Y. Sun, "Modeling learner's dynamic knowledge construction procedure and cognitive item difficulty for knowledge tracing," *Applied Intelligence*, vol. 50, pp. 3894–3912, 2020.

[20] X. Song, J. Li, Q. Lei, W. Zhao, Y. Chen, and A. Mian, "Bi-CLKT: Bi-graph contrastive learning based knowledge tracing," *Knowledge-Based Systems*, vol. 241, p. 108274, 2022.

[21] J. M. Royer, "Theories of the transfer of learning," *Educational Psychologist*, vol. 14, no. 1, pp. 53–69, 1979.

[22] T.-Y. Wu, H. Li, S. Kumari, and C.-M. Chen, "A Spectral Convolutional Neural Network Model Based on Adaptive Fick's Law for Hyperspectral Image Classification," *Computers, Materials & Continua*, vol. 79, no. 1, pp. 19–46, 2024.

[23] T.-Y. Wu, A. Shao, and J.-S. Pan, "CTOA: Toward a Chaotic-Based Tumbleweed Optimization Algorithm," *Mathematics*, vol. 11, no. 10, p. 2339, 2023.

[24] T.-Y. Wu, H. Li, and S.-C. Chu, "CPPE: An Improved Phasmatodea Population Evolution Algorithm with Chaotic Maps," *Mathematics*, vol. 11, no. 9, p. 1977, 2023.

[25] S. Grossberg, "Recurrent neural networks," *Scholarpedia*, vol. 8, no. 2, p. 1888, 2013.

[26] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[27] V. Mnih, N. Heess, and A. Graves, "Recurrent models of visual attention," in *Advances in Neural Information Processing Systems*, vol. 27, p. 3218, 2014.