# Multiple Attention Modules-based Knowledge Tracing

Kai Zhang, Zhengchu Qin, Yue Liu, Xinyi Qin

***Abstract*—Knowledge tracing quantifies student's knowledge state by analyzing their interaction with exercises and predicts their future answers. This study proposes a Multiple Attention Modules-based Knowledge Tracing model to improve the representation of learning and forgetting behaviors in knowledge tracing. The proposed model employs three attention modules to shape learned and forgotten behaviors. The conceptual attention module calculates the similarity between concepts, while the state attention module measures the similarity between concept mastery states. The behavioral attention module helps the model to pay explicit attention to student's exercise interactions. To assess the effectiveness of the three attention modules on modeling performance, the study explores their impact on learning and forgetting behavior by ablating them in turn. The experimental results demonstrate that all three attention modules contribute positively to the modeling performance. In comparison with several other knowledge tracing models, the proposed model shows better performance on four real datasets.**

***Index Terms*—attention mechanism, deep learning, forgetting behaviors, knowledge tracing, learning behaviors**

## I. Introduction

Knowledge tracing employs a Machine Learning approach to model the sequence of student exercises. This approach analyzes the student's exercise interactions to trace their knowledge state. Finally, it predicts the student's future performance when answering new questions. This method has proven to be effective in the field of smart education, and its use has increased in recent years due to the growing popularity of online learning platforms. In general, KT models take exercise interactions $X=(x_1,x_2,...,x_t)$ as input, where $x_t=(q_t,r_t)$ represents the exercise and corresponding answer at timestamp $t$. The answer is typically indicated as 0 for incorrect and 1 for correct. The model's output is represented as $P=(r_t\mid q_t)$, indicating the probability that the student will answer future exercises correctly.

Traditional knowledge tracing models include Bayesian Knowledge Tracing (BKT)[1], Deep Knowledge Tracing (DKT)[2], and Dynamic Key-Value Memory Networks (DKVMN)[3]. BKT traces knowledge state based on Hidden Markov Model (HMM)[4]. DKT incorporates Deep Learning methods and uses recurrent neural networks with hidden vectors to represent knowledge state. DKVMN borrows from memory networks[5] and uses two external matrices to store concepts and knowledge state respectively.

The above models provide a foundation for tracing student knowledge, but do not model learning and forgetting behavior in depth. To enhance model's ability to represent learning and forgetting behavior, this paper aims to:

***1)* The similarity between concepts is described.** Firstly, the model autonomously generates vector representations of concepts and stores them in matrix $\boldsymbol{M}^k$. Secondly, an attention mechanism is utilized to obtain attention weights between concepts, thereby characterizing their interconnections.

***2)* The similarity between mastery states of concepts is described.** Firstly, the model represents the student's mastery states of concepts and stores them as vector expressions in the matrix $\boldsymbol{M}^v$. Secondly, an attention mechanism is utilized to obtain the attention weights between concept mastery states, which describes the similarity between students as they learn and forget these concepts.

***3)* Multiple attention modules-based knowledge tracing is proposed.** Firstly, an attention mechanism is employed to model the forgetting behavior, where the output results are represented by the vector $\boldsymbol{o}_t$. Secondly, by utilizing the similarity between the concept mastery state and vector $\boldsymbol{o}_t$, the learning vector and forgetting vector of the student are obtained to update the concept mastery state, thus modeling the process of knowledge state change.

## II. Related Work

### A. Knowledge Tracing

BKT is the most representative of the probabilistic knowledge tracing models, and was the first to propose a model for knowledge tracing. BKT uses binary variables to represent student's mastery of concepts and updates them based on their past exercise interactions using HMM.

BKT models concepts individually, but in reality, they are interconnected and hierarchical[6]. To address this limitation, Kaser *et al.*[7] proposed a Dynamic Bayesian Knowledge

Manuscript received 15 September, 2022; revised 17 April, 2023.

This study was supported by the National Natural Science Foundation of China (No. 62077018); The Ministry of Science and Technology high-end foreign experts introduction plan (No. G2022027006L); Natural Science Foundation Program of Hubei, China (No.2022CFB132); Hubei Provincial Department of Education Science Research Program (No. B2022038).

Kai Zhang is a professor in the School of Computer Science, Yangtze University, Hubei 434000 China (e-mail: kai.zhang@yangtzeu.edu.cn).

Zhengchu Qin is a graduate student in the School of Computer Science, Yangtze University, Hubei 434000 China (e-mail: 2021710632@yangtzeu.edu.cn).

Yue Liu is a graduate student in the School of Computer Science, Yangtze University, Hubei 434000 China (e-mail: 291864220@qq.com).

Xinyi Qin is a graduate student in the School of Computer Science, Yangtze University, Hubei 434000 China (e-mail: 2365692561@qq.com).

Tracing that represents multiple concepts jointly, improving the model's representational power. Moreover, BKT has also extended TLS-BKT[8] and other models. However, they ignore the impact of similarities between concepts on the student's knowledge state

DKT is the first model that applies Deep Learning methods to knowledge tracing, using recurrent neural networks[9] or long short-term memory networks[10] to trace student's knowledge state. DKVMN proposes a key matrix to store the concept, and a value matrix stores the student's mastery of the concept to trace student's knowledge state. DKT and DKVMN achieved good results in predicting student’s future performance. However, rather than modeling forgetting behavior, they opted to utilize memory erasure mechanisms.

Suragani *et al*.[11] introduced three new input features, namely the number of hints, first response time, and the number of attempts, to DKT, in order to enhance its performance. Yeung *et al*.[12] utilized DKVMN to track the student's knowledge state, and combined it with Item Response Theory (IRT) to extract the student's ability and the exercise's difficulty, resulting in improved performance of the model. Liu *et al*.[13] proposed a hierarchical memory network inspired by the mechanism of human memory. Sun *et al*.[14] proposed Collaborative Embedding for Knowledge Tracing, which incorporates the student's interactions with exercises and the connections between exercises and concepts. Abdelrahman *et al*.[15] utilized a Hop-LSTM model, which is based on DKVMN, to capture the long-term dependencies of the student's interaction with the exercises, and achieved promising results.

Although these models have shown improved performance in knowledge tracing, they continue to use memory erasure mechanisms to model the forgetting behavior of students. However, this approach may not be reasonable. Sandoval *et al*.[16] argue that forgetting is a necessary part of the brain's memory system to maintain a balance between encoding and integrating new information. Forgetting helps to eliminate unused or unwanted memories or inhibit their expression. Ebbinghaus forgetting curve[17] indicates that forgetting starts immediately after learning and the initial phase of forgetting is rapid. Subsequently, forgetting gradually slows down, and the degree of forgetting is influenced by the number of repetitions and the time interval between learning sessions.

Nagatani *et al*.[18] proposed DKT-F, which improves DKT by considering three factors that influence forgetting: the number of times a concept is repeatedly learned, the time interval between learning the same concept, and the time interval of learning. However, this model does not consider the influence of student’s mastery states of the concept and the similarity between concept mastery states on forgetting behavior. In contrast, inspired by educational psychology, Li *et al*.[19] proposed LFKT: a deep knowledge tracing model that merges learning and forgetting behavior. LFKT considers not only the above three factors affecting forgetting but also the influence of student’s conceptual mastery states on forgetting behavior. However, this model did not take into account the influence of the similarity between concept mastery states on forgetting behavior.

In general, most researchers have not modeled student’s forgetting behavior, but have used memory erasure mechanisms instead, or have modeled forgetting behavior but with incomplete consideration. Most researchers have traced student’s knowledge state using only practice interactions and used them as direct inputs to the model, without explicitly focusing the model on these interactions.

To address the aforementioned issues, this paper draws inspiration from research findings in cognitive neuroscience[16] and proposes a knowledge tracing model. This model integrates the modeling of both learning and forgetting behavior by considering various factors that influence them. It also places explicit focus on practice interactions.

### *B. Attention Mechanism*

Biologically, attentional mechanisms[20] allow humans to selectively focus their attention based on nonvolitional and volitional cues[21]. Nonvolitional cues refer to situations where a person acquires information unconsciously, such as when a book and a red glass are placed together. The person first notices the red glass or even ignores the book because the color of the glass is more eye-catching. Volitional cues, on the other hand, occur when a person acquires information consciously. In the same scenario, if the person intends to read, they will first notice the book and even disregard the glass's presence.

Before the emergence of attention mechanisms, neural networks, such as convolutional neural networks and recurrent neural networks, only accounted for nonvolitional cues when processing input features. For instance, the maximum pooling layer retains the max feature values in a given region, without consciously focusing on specific feature values. The attention mechanism addresses this limitation by mapping a query and a set of key-value pairs to an output. Here, the query acts as a volitional cue, while the key and value serve as nonvolitional cues. The output is the weighted sum of the values, with each weight assigned based on the similarity between the query and the key. As the attention mechanism considers volitional cues, the output becomes biased towards certain input features, allowing for more explicit attention to be placed on these features by the model.

The self-attention mechanism is a variation of the attention mechanism with a similar idea. However, in the self-attention mechanism, the query, key, and value are all the same, reducing dependence on external information and improving the capture of internal data similarity.

## III. Model

In this paper, we propose a model called Multiple Attention Modules-based Knowledge Tracing (MAKT). The overall structure of the model is illustrated in Figure 1.

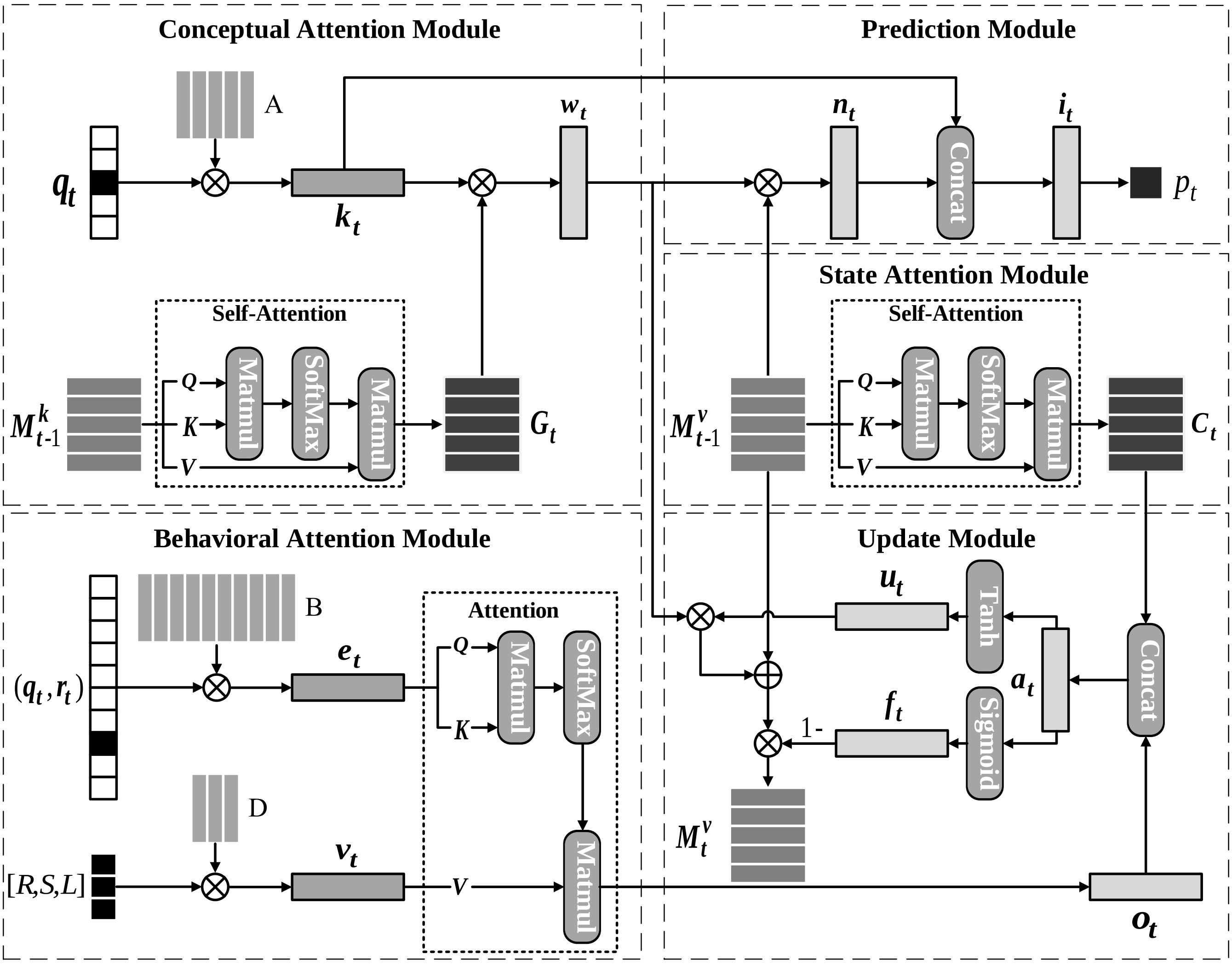

Fig. 1. Multiple Attention Modules-based Knowledge Tracing

MAKT is composed of five distinct modules: the conceptual attention module, the state attention module, the behavioral attention module, the update module, and the prediction module. The conceptual attention module computes the similarity between concepts in the matrix $\boldsymbol{M^k} \in \mathbb{R}^{d_k \times N}$ . The state attention module computes the similarity between the concept mastery states in the matrix $\boldsymbol{M^v} \in \mathbb{R}^{d_v \times N}$ . The behavioral attention module provides the model with more explicit attention to exercise interactions. The update module updates the matrix $\boldsymbol{M^v_{t\text{-}1}}$ using the output from the state attention module and the behavioral attention module. The prediction module forecasts the student's performance at time $t$ based on the matrix $\boldsymbol{M^v_{t\text{-}1}}$.

### A. Introduction to Attention Modules

Although many previous studies have recognized that the similarity between concepts and mastery states can impact student's learning and forgetting behaviors, recent findings in cognitive neuroscience indicate that the similarity between mastery states has a more direct influence on these behaviors[16]. This similarity refers to which students learn and forget concepts in a similar manner.

The conceptual attention module is designed to capture the concept similarity features, with its output represented by the concept attention matrix $\boldsymbol{G_t}$. The state attention module is responsible for capturing the concept mastery state similarity components, and its output is represented by the state attention matrix $\boldsymbol{C_t}$ .

On the other hand, Previous researchers have primarily focused on the influence of exercise interactions ( $E$ ) on student's learning and forgetting behaviors, using it as a direct input to the knowledge tracing models. However, they have overlooked the impact of certain behaviors on exercise interactions during the learning process, such as the number of repetitions of learning ( $R$ ), the sequence time interval ( $S$ ), and the learned time interval ( $L$ ). For example, the higher the value of $L$ and the smaller the values of $S$ and $R$, the better the outcome of student's exercise interaction. To address the issue, this paper incorporates volitional cues into the behavioral attention module to increase the model's explicit attention to exercise interactions. As a result, the output is represented by the behavioral attention vector $\boldsymbol{o_t}$ .

### B. Multiple Attention Modules-Based Knowledge Tracing

#### 1) Conceptual Attention Module

The conceptual attention module uses the matrix $\boldsymbol{M^k} \in \mathbb{R}^{d_k \times N}$ , which stores the concept, as input to calculate the similarity between concepts. The resulting calculation is represented by the concept attention matrix $\boldsymbol{G_t}$ . Inspired by Ashish *et al.*[22], MAKT uses the scaled dot product self-attention mechanism:

$$\boldsymbol{G_t} = \text{Softmax}(\frac{\boldsymbol{M^k_{t-1}}\boldsymbol{M^{k^T}_{t-1}}}{\sqrt{d_k}})\boldsymbol{M^k_{t-1}} \tag{1}$$

where $\text{Softmax}(x_i) = x_i / \sum_{n=1}^{N}(e^{x_n})$ ; the matrix $\boldsymbol{M^k_{t-1}}$ stores vectors representing concepts; the concept attention matrix $\boldsymbol{G_t} \in \mathbb{R}^{d_k \times N}$ is the result of the calculation, which contains information about the similarity between concepts.

To calculate the similarity between the concepts in exercise $q_t$, we first convert $q_t$ into one-hot encoding, and then multiply it with the embedding matrix $\boldsymbol{A} \in \mathbb{R}^{d_k \times N}$ to obtain the exercise embedding vector $\boldsymbol{k_t} \in \mathbb{R}^{1\times \mathrm{d_k}}$. The vector $\boldsymbol{k_t}$ is then multiplied by the concept attention matrix $\boldsymbol{G_t}$ to obtain the association weight $\boldsymbol{w_t}$, which represents the similarity between the concepts contained in exercise $q_t$:

$$\boldsymbol{w_t} = \text{Softmax}(\boldsymbol{k_t} \times \boldsymbol{G_t}) \tag{2}$$

*2) State Attention Module*

The state attention module uses the matrix $\boldsymbol{M^v} \in \mathbb{R}^{d_v \times \mathrm{N}}$, which stores the concept mastery states, as input to calculate the similarity between concept mastery states. The resulting calculation is represented by the state attention matrix $\boldsymbol{C_t}$:

$$\boldsymbol{C_t} = \text{Softmax}(\frac{\boldsymbol{M^v_{t-1}}\boldsymbol{M^{v^T}_{t-1}}}{\sqrt{d_v}})\boldsymbol{M^v_{t-1}} \tag{3}$$

where the matrix $\boldsymbol{M^v_{t-1}}$ stores vectors representing the concept mastery states; the state attention matrix $\boldsymbol{C_t} \in \mathbb{R}^{d_v \times \mathrm{N}}$ is the result of the calculation, which contains information about the similarity between concept mastery states.

*3) Behavioral Attention Module*

In the behavioral attention module, The student's exercise interactions $E$ are converted into a one-hot encoding first, and then multiplied with the embedding matrix $\boldsymbol{B} \in \mathbb{R}^{d_v \times 2N}$ to obtain a vector $\boldsymbol{e_t} \in \mathbb{R}^{1\times d_v}$, which represents the student's exercise interactions. The values of $R$, $S$, and $L$ are concatenated to form a row vector $[R,S,L]$ of dimension three and are then normalized. The resulting vector is then multiplied with the embedding matrix $\boldsymbol{D} \in \mathbb{R}^{3\times d_v}$ to obtain a vector $\boldsymbol{v_t} \in \mathbb{R}^{1\times d_v}$, which represents the student's behavior during the learning process.

Certain student behaviors during the learning process have a direct impact on exercise interactions. MAKT leverages an attention mechanism to reveal the relationship between the two. By using vector $\boldsymbol{v_t}$ to represent volitional cues as the input for the query, and vector $\boldsymbol{e_t}$ to represent nonvolitional cues as the input for the key and value, MAKT becomes more explicitly attentive to exercise interactions with the addition of volitional cues:

$$\boldsymbol{o_t} = \text{Softmax}(\boldsymbol{v_t}\boldsymbol{e_t}^{\boldsymbol{T}})\boldsymbol{e_t} \tag{4}$$

where, the behavioral attention vector $\boldsymbol{o_t}$ is the result of the calculation, and its dimension is $d_v$, which is a composite representation of $R$, $S$, $L$ and $E$.

*4) Update Module*

The behavioral attention vector $\boldsymbol{o_t}$ and the state attention matrix $\boldsymbol{C_t}$ are essential in modeling learning and forgetting behavior. They are spliced and fed into a fully connected layer with a Tanh activation function, resulting in a vector $\boldsymbol{a_t} \in \mathbb{R}^{1\times \boldsymbol{d_v}}$ that represents a composite of the $\boldsymbol{o_t}$ and $\boldsymbol{C_t}$ components:

$$\boldsymbol{a_t} = \text{Tanh}(\boldsymbol{w}_1^{\boldsymbol{T}}[\boldsymbol{C_t},\boldsymbol{o_t}] + \boldsymbol{b}_1) \tag{5}$$

To convert the vector $\boldsymbol{a_t}$ into a forgotten vector $\boldsymbol{f_t} \in \mathbb{R}^{1\times \boldsymbol{d_v}}$, a fully connected layer with a Sigmoid activation function is used:

$$\boldsymbol{f_t} = \text{Sigmoid}(\boldsymbol{W_f^T}\boldsymbol{a}_t + \boldsymbol{b_f}) \tag{6}$$

where $\text{Sigmoid}(x_i) = 1/(1+e^{-x_i})$, it is used to ensure that the forgotten vector $\boldsymbol{f_t}$ is a valid probability distribution, where each element in $\boldsymbol{f_t}$ represents the probability that the corresponding concept has been forgotten. And $\boldsymbol{W_f}$ is the weight matrix of the fully connected layer, $\boldsymbol{b_f}$ is the bias vector.

To convert the vector $\boldsymbol{a_t}$ into a learning vector $\boldsymbol{u_t} \in \mathbb{R}^{1\times \boldsymbol{d_v}}$, a fully connected layer with a Tanh activation function is used:

$$\boldsymbol{u_t} = \text{Tanh}(\boldsymbol{W_u^T}\boldsymbol{a_t} + \boldsymbol{b_u}) \tag{7}$$

where $\text{Tanh}(x_i) = (e^{x_i} - e^{-x_i})/(e^{x_i} + e^{-x_i})$, it is utilized to constrain the output values of the fully connected layer within a suitable range, allowing the modeling of learned behavior. Each element in the resulting learning vector $\boldsymbol{u_t}$ corresponds to a learned concept. And $\boldsymbol{W_u}$ is the weight matrix of the fully connected layer, $\boldsymbol{b_u}$ is the bias vector.

The matrix $\boldsymbol{M^v_{t-1}}$ is updated by utilizing the learning vector $\boldsymbol{u_t}$ and forgetting vector $\boldsymbol{f_t}$, along with the associated weights $\boldsymbol{w_t}$:

$$\boldsymbol{M^v_t}(\boldsymbol{i}) = \boldsymbol{M^v_{t-1}}(\boldsymbol{i})(1-\boldsymbol{f_t}) + \boldsymbol{u_t}\boldsymbol{w_t}(\boldsymbol{i}) \tag{8}$$

*5) Prediction Module*

In the prediction module, the association weights $\boldsymbol{w_t}$ are multiplied by the matrix $\boldsymbol{M^v_{t-1}}$ to obtain the vector $\boldsymbol{n_t}$, which is the output of the student's concept mastery state:

$$\boldsymbol{n_t} = \boldsymbol{w_t}\boldsymbol{M^v_{t-1}} \tag{9}$$

Considering that there will be some differences between exercises, such as different difficulty coefficients, the exercise embedding vector $\boldsymbol{k_t}$ is concatenated with the vector $\boldsymbol{n_t}$. This concatenated vector contains both the student's conceptual mastery state and the exercise information. Inputting this vector to the fully connected layer with the Tanh activation function to obtain the vector $\boldsymbol{i_t}$:

$$\boldsymbol{i_t} = \text{Tanh}(\boldsymbol{w}_2^{\boldsymbol{T}}[\boldsymbol{n_t},\boldsymbol{k_t}] + \boldsymbol{b}_2) \tag{10}$$

Finally, an output layer with a Sigmoid activation function, used $\boldsymbol{i_t}$ as input, is used to predict how well students will perform on exercise $q_t$:

$$p_t = \mathrm{Sigmoid}(\boldsymbol{w}_3^{\boldsymbol{T}} \boldsymbol{i_t} + \boldsymbol{b}_3) \tag{11}$$

### C. Training

In the training process, we use the cross-entropy loss function to minimize the discrepancy between the predicted and actual labels and to learn the embedding matrices $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{D}$, as well as other parameters like $\boldsymbol{M}_t^k$.

$$Loss = -\sum_t (r_t \log p_t + (1-r_t)\log(1-p_t)) \tag{12}$$

## IV. Experiments

### A. Dataset

To validate the effectiveness of the MAKT model, experiments were conducted on four actual datasets: ASSISTments2012, ASSISTments2017, Slepemapy.cz, and JunyiAcademy. For the sake of brevity, they will be referred to as Assist12, Assist17, Slepemapy, and Junyi respectively. Table I displays the fundamental information of the four datasets.

TABLE I
Dataset Introduction

| Dataset | Number of | | |
|---|---|---|---|
| | Students | Records | Concepts |
| Assitst12 | 46674 | 5818868 | 266 |
| Assitst17 | 1709 | 942816 | 102 |
| Slepemapy | 87952 | 10087305 | 1458 |
| Junyi | 238120 | 26666117 | 684 |

Compared to ASSISTments2009, the Assist12 and Assist17 datasets provide more information about student’s answers, such as startTime and endTime. However, Xiong *et al.*[23] found problems with the ASSISTments2009 dataset, such as duplicate records and confusion between central questions and scaffolding questions. Although the ASSISTments platform promptly resolved the issue of duplicate records, it still mixes the main question with the scaffolding question in different researchers' studies. This paper's experiments will address these issues by filtering out scaffolding questions and removing student exercise interactions with fewer than three interaction records.

### B. Evaluation indicators

In this paper, we evaluate the performance of the proposed model using two commonly used metrics in the field of knowledge tracing: AUC (Area Under Curve) and ACC (Accuracy). AUC is calculated as the area under the ROC (Receiver Operating Characteristic) curve and ranges between 0.5 and 1. A value of 0.5 indicates a random prediction model, while a higher value indicates better prediction performance. ACC is the ratio of correct predictions to total predictions, and a higher value indicates more accurate predictions.

### C. Experimental Details

The experimental environment of this paper is shown in Table II.

TABLE II
Experimental environment

| Experimental configuration | Parameter Value |
|---|---|
| OS | Windows 11 |
| CPU | Inter(R) Core(TM) i9-9900K CPU@3.60GHz |
| GPU | NVIDIA GeForce RTX 3080 Ti |
| Python | 3.10 |
| Pytorch | 1.10.2 |
| RAM | 64GB |

In each dataset, 80% of the data is allocated as the training set and 20% is allocated as the test set. From the 20% of the training set, a further 20% is randomly selected as the validation set, which is used to tune the hyperparameters and select the best model.

To account for variations in the number of students, exercise interactions, and concepts across datasets, we set the initial learning rate to 0.001, which is relatively high, and employ learning rate decay after every ten training epochs. We use Adam as the optimizer and a batch size of 32. The model parameters are initialized randomly with a Gaussian distribution with a mean of zero and a standard deviation.

The MAKT model also has a dimensional $d_v$ of the concept mastery state and a dimensional $d_k$ of the concept in terms of parameter settings. To simplify the calculation, $d$ is set to be equal to both $d_v$ and $d_k$. The model's performance is evaluated based on the AUC value, and the impact of different parameters on its performance is explored. The experimental results are presented in Table III.

TABLE III
Hyperparameters impact AUC value

| d | Assist12 | Assist17 | Slepemapy | Junyi |
|---|---|---|---|---|
| 8 | 0.803 | 0.745 | 0.813 | 0.809 |
| 16 | 0.807 | 0.754 | 0.825 | 0.839 |
| 32 | 0.810 | 0.769 | **0.831** | 0.827 |
| 64 | **0.816** | **0.784** | 0.826 | **0.844** |
| 128 | 0.815 | 0.779 | 0.825 | 0.817 |

The experimental results indicate that the MAKT model's performance is affected by different hyperparameter settings. Specifically, in the Assist12 dataset, the optimal model performance is achieved when $d$ is set to 64, resulting in an AUC of 0.816. In the Assist17 dataset, the optimal model performance is also achieved when $d$ is 64, resulting in an AUC of 0.784. For the Slepemapy dataset, the optimal model performance is achieved when $d$ is set to 32, resulting in an AUC of 0.831. Finally, in the Junyi dataset, the optimal model performance is achieved when $d$ is set to 64, resulting in an AUC of 0.844.

### D. Effect of Different Modules on Model Performance

This study aims to investigate the impact of the performance of the model by ablating the three attention modules (conceptual attention module, state attention module, and behavioral attention module) in MAKT. To replace the ablated modules, the corresponding functions of DKVMN, which also uses external matrices to store concepts and mastery states of concepts, are used instead. Table IV presents the specific structure of the model, where MAKT-C denotes that only the conceptual attention module is retained while the other two modules are ablated.

TABLE IV
MODELS FOR FUSING DIFFERENT MODULES

| Conceptual Attention Module | State Attention Module | Behavioral Attention Module | Model |
|---|---|---|---|
| √ | | | MAKT-C |
| | √ | | MAKT-S |
| | | √ | MAKT-B |
| √ | √ | | MAKT-CS |
| | √ | √ | MAKT-SB |
| √ | | √ | MAKT-CB |
| √ | √ | √ | MAKT |

This paper compares the model in Table IV with DKVMN on four real data sets for performing experiments. The experimental results are shown in Table V and Table VI.

TABLE V
COMPARISON OF AUC VALUES OF DIFFERENT MODELS

| Model | Dataset | | | |
|---|---|---|---|---|
| | Assist12 | Assist17 | Slepemapy | Junyi |
| MAKT-C | 0.733 | 0.693 | 0.803 | 0.817 |
| MAKT-S | 0.753 | 0.738 | 0.824 | 0.839 |
| MAKT-B | 0.747 | 0.709 | 0.809 | 0.820 |
| MAKT-CS | 0.756 | 0.745 | 0.821 | 0.833 |
| MAKT-SB | 0.781 | 0.768 | 0.829 | 0.841 |
| MAKT-CB | 0.750 | 0.711 | 0.800 | 0.832 |
| MAKT | **0.816** | **0.784** | **0.831** | **0.844** |
| DKVMN | 0.732 | 0.707 | 0.792 | 0.822 |

TABLE VI
COMPARISON OF ACC VALUES OF DIFFERENT MODELS

| Model | Dataset | | | |
|---|---|---|---|---|
| | Assist12 | Assist17 | Slepemapy | Junyi |
| MAKT-C | 0.701 | 0.695 | 0.753 | 0.735 |
| MAKT-S | 0.728 | 0.720 | 0.785 | 0.763 |
| MAKT-B | 0.715 | 0.711 | 0.763 | 0.741 |
| MAKT-CS | 0.720 | 0.720 | 0.785 | 0.761 |
| MAKT-SB | 0.737 | 0.722 | 0.797 | 0.768 |
| MAKT-CB | 0.709 | 0.697 | 0.763 | 0.743 |
| MAKT | **0.749** | **0.724** | **0.803** | **0.770** |
| DKVMN | 0.686 | 0. 677 | 0.743 | 0.751 |

The experimental results demonstrate that the performance of MAKT is superior to that of DKVMN in most cases, even though some models in the Junyi dataset have lower performance than DKVMN. Furthermore, the higher the number of retained modules in MAKT, the higher the model's performance. This suggests that incorporating inter-concept similarity, concept mastery state similarity, and volitional cues can help trace student's knowledge state more effectively.

Among the three MAKT models, MAKT-S achieves the best performance, followed by MAKT-B and MAKT-C. This indicates that the state attention module is the most important when modeling learning and forgetting behavior, followed by the behavioral attention module, and then the conceptual attention module. This is also evident in the comparison of the three models, MAKT-CS, MAKT-SB, and MAKT-CB, where the model including the state and behavioral attention modules performs the best, the model containing the state and conceptual attention modules comes in second, and the model including the behavioral and conceptual attention modules has the lowest performance.

### *E. Comparative Analysis of Model Performance*

To evaluate the model's performance in this paper, MAKT is compared with other KT models. The experimental results are shown in Table VII and Table VIII.

TABLE VII
AUC VALUES OF DIFFERENT KNOWLEDGE TRACING MODELS

| Model | Dataset | | | |
|---|---|---|---|---|
| | Assist12 | Assist17 | Slepemapy | Junyi |
| DKT | 0.717 | 0.726 | 0.742 | 0.814 |
| DKT-F | 0.722 | 0.729 | 0.749 | 0.840 |
| DKVMN | 0.732 | 0.707 | 0.792 | 0.822 |
| LFKT | 0.751 | — | 0.803 | — |
| MAKT | **0.816** | **0.784** | **0.831** | **0.844** |

TABLE VIII
ACC VALUES OF DIFFERENT KNOWLEDGE TRACING MODELS

| Model | Dataset | | | |
|---|---|---|---|---|
| | Assist12 | Assist17 | Slepemapy | Junyi |
| DKT | 0.679 | 0. 682 | 0.731 | 0.744 |
| DKT-F | 0.685 | 0.681 | 0.753 | 0.759 |
| DKVMN | 0.686 | 0. 677 | 0.743 | 0.751 |
| LFKT | 0.723 | — | 0.762 | — |
| MAKT | **0.749** | **0.724** | **0.803** | **0.770** |

Based on the experimental results presented in this paper, it was observed that DKT-F outperformed DKT in the four real datasets, as DKT-F models forgetting behavior based on DKT. Additionally, DKVMN demonstrated higher overall AUC and ACC values than DKT due to its unique approach of modeling the mastery state of each concept instead of just the overall knowledge state of the student. The varying performance among these models is attributed to their diverse modeling strategies.

In comparison to DKVMN, both MAKT, and LFKT demonstrated higher AUC and ACC values on the four real datasets, as they model forgetting behavior while also modeling student's learning behavior. However, MAKT outperformed LFKT primarily because it considers not only the influence of concept mastery state on student learning and forgetting behavior, but also the influence of similarity between concepts and similarity between concept mastery states on these behaviors. Additionally, MAKT adds volitional cues to give explicit attention to the exchange of exercises. Overall, the AUC and ACC values of MAKT exceeded those of the baseline model on all datasets, indicating that the model performance of this paper has some advantages.

### *F. Model consistency comparison experiment*

If student A and student B are answering the same exercise related to concept C, it is more likely that student A will answer correctly if student A’s knowledge state of concept C is better than that of student B. Degree of Agreement (DOA) quantifies the quality of the knowledge state portrayed by the model based on this fact. The DOA metric is calculated by the following formula：

$$DOA(c)=\frac{1}{\boldsymbol{Z}}\sum_{a=1}^{S}\sum_{b=1}^{S}\delta(V_c^a,V_c^b)\sum_{j=1}^{M}\boldsymbol{I}_{jc}\frac{\boldsymbol{C}(\boldsymbol{c},\boldsymbol{a},\boldsymbol{b})\wedge\delta(\boldsymbol{r}_{jc}^{a},\boldsymbol{r}_{jc}^{b})}{\boldsymbol{C}(\boldsymbol{c},\boldsymbol{a},\boldsymbol{b})}$$

where $\boldsymbol{V}_c^a$ denotes the knowledge state of student $\boldsymbol{a}$ about the concept $\boldsymbol{c}$ ; $\delta(\boldsymbol{x},\boldsymbol{y})=1$ if $\boldsymbol{x}>\boldsymbol{y}$ , otherwise $\delta(\boldsymbol{x},\boldsymbol{y})=0$ ; $\boldsymbol{I}_{jc}=1$ if exercise $\boldsymbol{j}$ contains the concept $\boldsymbol{c}$ , otherwise $\boldsymbol{I}_{jc}=0$ ; $\boldsymbol{C}(\boldsymbol{c},\boldsymbol{a},\boldsymbol{b})=1$ if both student $\boldsymbol{a}$ and student $\boldsymbol{b}$ answered the exercise about concept $\boldsymbol{c}$ , otherwise $\boldsymbol{C}(\boldsymbol{c},\boldsymbol{a},\boldsymbol{b})=0$ . $\boldsymbol{r}_{jc}^{a}$ denotes the true state of student $\boldsymbol{a}$ answer to exercise $\boldsymbol{j}$ about the concept $\boldsymbol{c}$ . The average DOA value was used to evaluate the quality of the knowledge state traced by the model for each concept. Figure 2 illustrates the comparison between MAKT and the baseline model in terms of DOA.

The results in Figure 2 demonstrate that the DOA values for MAKT are consistently higher than those of the baseline model across all datasets. This suggests that the knowledge state generated by the MAKT model are more accurate, of higher quality, and more closely aligned with the actual situation than those generated by the comparison model.

### *G. Model training efficiency comparison experiment*

The number of iterations during model training is indicative of the model's training efficiency. A lower number of iterations implies a higher training efficiency, provided that the optimal AUC and ACC values are achieved. Figure 3 presents the experimental results comparing the number of iterations in MAKT and the baseline model.

Figure 3 presents the experimental results comparing the number of iterations in MAKT and the baseline model training on all datasets. The findings indicate that MAKT has higher training efficiency than other baseline models, as the number of iterations required for MAKT training is consistently lower.

Further, we explored the training time efficiency of MAKT and the comparison models. Specifically, we compared the average running time of training a single batch of different models on the Assist12 dataset. The experimental results are shown in Table IX.

TABLE IX
COMPARISON OF TRAINING TIME EFFICIENCY OF EACH MODEL

| Model | DKT | DKVMN | DKT-F | LFKT | MAKT |
|---|---|---|---|---|---|
| Time | 28.30 | 17.53 | 30.68 | 53.41 | 19.68 |

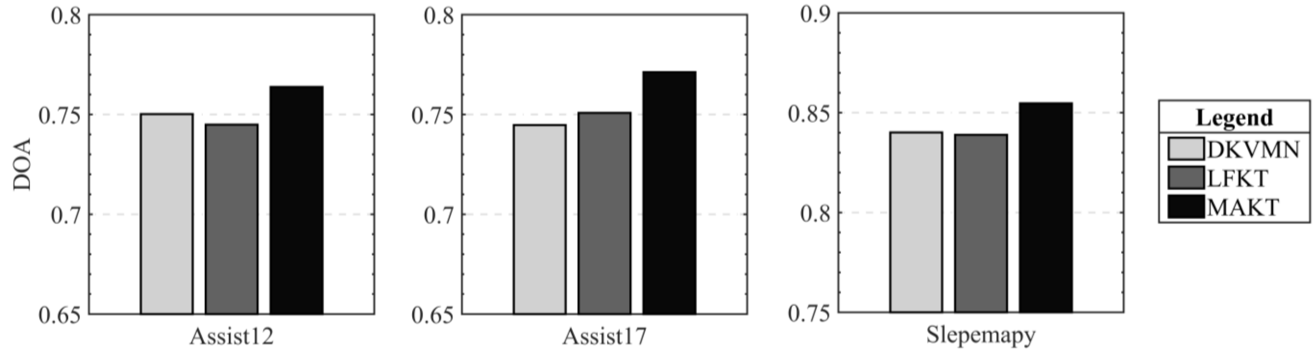

Fig. 2. DOA value of the models

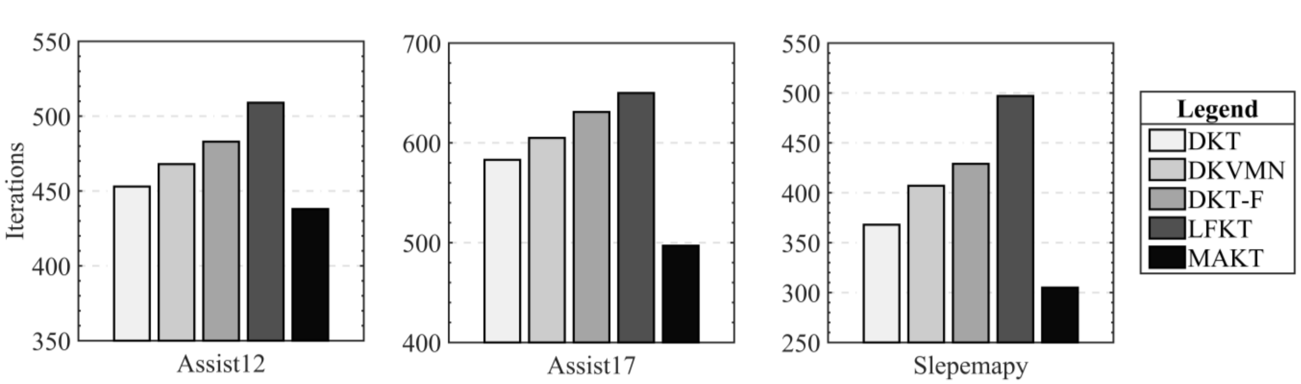

Fig. 3. Iterations value of the models

As can be seen from table IX, The training time required for a single batch of MAKT is lower than that of most of the other models used for comparison. On the other hand, LFKT takes the longest time for training a single batch, followed by DKT and DKT-F, which is related to their use of RNN or LSTM models. When computing LSTM networks, gating units such as forgetting gates, input gates, and output gates need to calculate the current output based on the previous output results, which increases the time cost.

## V. Conclusion

In this study, we propose a model called Multiple Attention Modules-based Knowledge Tracing (MAKT), which aims to improve the representation of learning and forgetting behaviors in knowledge tracing. MAKT employs three attention modules: the conceptual attention module discovers concept-to-concept similarities; the state attention module detects similarities between student’s concept mastery states; and the behavioral attention module amplifies the model's attention on exercise interactions. The three modules work together to model student’s learning and forgetting behaviors. The update module describes changes in student’s concept mastery states, and the prediction module forecasts their future answers. This paper experimentally investigates the effects of different modules on model performance and compares them with multiple knowledge tracing models on four real data sets. The experimental results reveal that the multiple attention modules proposed in this paper effectively trace student’s knowledge state and outperform other models in terms of performance.

## References

[1] Albert T. Corbett and John R. Anderson, “Knowledge tracing: modeling the acquisition of procedural knowledge,” User Modeling and User-Adapted Interaction, vol. 4, no. 4, pp. 253–278, 1995.

[2] Chris Piech, Jonathan Bassen, Jonathan Huang, et al., “Deep knowledge tracing,” Advances in Neural Information Processing Systems, vol. 28, 2015.

[3] Jiani Zhang, Xingjian Shi, Irwin King, et al., “Dynamic key-value memory networks for knowledge tracing,” in Proceedings of the 26th international conference on World Wide Web, pp. 765–774, Perth, Australia, April 2017.

[4] L. Rabiner and B. Juang, “An introduction to hidden Markov models,” Ieee Assp Magazine, vol. 3, no. 1, pp. 4–16, 1986.

[5] Adam Santoro, Sergey Bartunov, Matthew Botvinick, et al., “Meta-learning with memory-augmented neural networks,” in Proceedings of the International conference on machine learning, pp. 1842–1850, 2016.

[6] Qi Liu, Shuanghong Shen, Zhenya Huang, et al., “A survey of knowledge tracing,” 2021, https://arxiv.org/abs/2105.15106.

[7] Tanja Kaser, Seven Klingler, Alexander G. Schwing, et al., “Dynamic bayesian networks for student modeling,” IEEE Transactions on Learning Technologies, vol. 10, no. 4, pp. 450–462, 2017.

[8] Kai Zhang and Yiyu Yao, "A three learning states Bayesian knowledge tracing model," Knowledge-Based Systems, vlo.148, pp189-201, 2018.

[9] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” Science, vol. 313, no. 5786, pp. 504–507, 2006.

[10] Seep Hochreiter, Jurgen Schmidhuber, “Long short-term memory,” Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[11] Girish Suragani, Lakshmi Narayana Pothuraju, Kamal Sandeep Reddi, et al. "Enhancing Deep Knowledge Tracing (DKT) Model by Introducing Extra Student Attributes," 2019 IEEE 5th International Conference for Convergence in Technology (I2CT). IEEE, pp1-5, 2019.

[12] Chun-Kit Yeung, "Deep-IRT: Make deep learning based knowledge tracing explainable using item response theory," https://arxiv.org/abs/1904.11738

[13] Sannyuya Liu, Rui Zou, Jianwen Sun et al. “A hierarchical memory network for knowledge tracing,” Expert Systems with Applications, vol. 177, Article ID 114935, 2021.

[14] Jianwen Sun, Jianpeng Zhou, Kai Zhang, et al. "Collaborative embedding for knowledge tracing," International Conference on Knowledge Science, Engineering and Management. Springer, Cham, vol.12816, pp333-342, 2021.

[15] Ghodai Abdelrahman, Qing Wang, "Knowledge tracing with sequential key-value memory networks," Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval. pp175-184, 2019.

[16] Isaac Cervantes-Sandoval, Molee Chakraborty, Courtney MacMullen, et al. "Scribble scaffolds a signalosome for active forgetting." Neuron vol.90, no.6, pp1230-1242, 2016.

[17] Jaap MJ murre, Joeri Dros. "Replication and analysis of Ebbinghaus’ forgetting curve," PloS one, vol.10, no.7, Article ID e0120644, 2015.

[18] Nagatani Koki, Zhang Qiang, Sato Masahiro, et al. "Augmenting knowledge tracing by considering forgetting behavior," The world wide web conference, pp3101-3107, 2019.

[19] Li Xiaoguang, Wei Siqi, Zhang Xin, et al. “LFKT: Deep knowledge tracing model with learning and forgetting behavior merging,” Ruan Jian Xue Bao/Journal of Software, vol.32, no.3, pp818−830, 2021.

[20] Shalini Pandey, George Karypis, "A self-attentive model for knowledge tracing," https://arxiv.org/abs/1907.06837.

[21] Aston Zhang, Zachary C. Lipton, Mu li, et al. "Dive into deep learning," https://arxiv.org/abs/2106.11342.

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. "Attention is all you need," Proceedings of the 31st International Conference on Neural Information Processing Systems, Curran Associates Inc., pp.6000-6010, 2017.

[23] Xiong Xiaolu, Zhao Siyuan, Van inwegen, et al., "Going deeper with deep knowledge tracing," International Educational Data Mining Society, 2016.