Multi-factor Memory Attentive Model for Knowledge Tracing

Congjie Liu JAPHERY@QQ.COM

Xiaoguang Li XGLI@LNU.EDU.CN
School of Information, Liaoning University, 110036, China

Editors: Vineeth N Balasubramanian and Ivor Tsang

Abstract

The traditional knowledge tracing with neural network usually embeds the required information and predicates the knowledge proficiency by embedded information. Only few information, however, is considered in traditional methods, such as the information of exercises in terms of concept. In this paper, we propose a multi-factor memory attentive model for knowledge tracing (MMAKT). In terms of Neural Cognitive Diagnosis (NeuralCD) framework, MMAKT introduces the factors of the knowledge concept relevancy, the difficulty of each concept, the discrimination among exercises and the student's proficiency to construct interaction vectors. Moreover, in order to achieve more accurate prediction precision, MMAKT introduces attention mechanism to enhance the expression of historical relationship between interactions. With the experiments on the real-world datasets, MMAKT shows better performance of knowledge tracing and prediction in comparision with the state-of-the-art approaches.

Keywords: intelligent education, knowledge tracing, deep neural network, cognitive diagnosis

1. Introduction

Intelligent tutoring systems provide personalized learning activity recommendations by analyzing data from learners' learning history. Item Response Theory(IRT)(Rasch (1993)) and Cognitive Diagnosis are main methods of the systems. Especially in the era of big data, the machine-learning based cognitive diagnosis has been pushed to the forefront. The dynamic cognitive diagnosis is also known as knowledge tracing (KT). Bayesian knowledge tracing (BKT) (Corbett and Anderson (1994)) and its variant (Baker and Yacef (2009); Yudelson et al. (2013)) became the primary methods accompanied by the proposition of KT. Later, RNN based knowledge tracing methods such as deep knowledge tracing (DKT) (Piech et al. (2015)) and dynamic key-value memory networks (DKVMN) (Zhang et al. (2017)) emerged with the rise of deep learning. These deep learning-based models aim to capture the long-term dependence of student-question interactions, and attempt to represent students' latent knowledge states with high-dimension matrices. SAKT (Pandey and Karypis (2019)), a recently proposed self-attentive model for knowledge tracing, utilized the self-attention mechanism to predict the student's response. Despite these deep learning-based models have proved excellent performance in realistic experiments, their interaction vectors are not

as reasonable as static cognitive diagnosis model's. Specifically, the interaction vectors of these KT models only require knowledge concepts of exercises, while static cognitive diagnosis methods usually need the student's proficiency, the knowledge concept relevancy, the exercise discrimination etc. besides the knowledge concept difficulty (Wang et al. (2020)). Moreover, these KT methods cannot present the historical relationship between exercises and interactions directly. For instance, RNN based KT methods relay on latent states to transfer information along the input sequence. No explicit historical relationship can be presented with the mess-up information in latent states. And SAKT only focuses on the weights between the target exercise and interactions using a single self-attention layer ignoring the historical relationship between exercises/interactions and themselves.

In this paper, we address these issues in a way of proposing a multi-factor memory attentive model for knowledge tracing (MMAKT). Specifically, MMAKT uses multi-factor such as knowledge concept relevancy, concept difficulty, exercise discrimination to construct exercise vectors, and dynamically traces the student's knowledge proficiency with DKVMN. Then the student-exercise interaction vectors are constructed with the factors above in terms of Neural Cognitive Diagnosis (Neural CD) framework making the interaction vectors more reasonable and containing abundant information. In order to express the historical relationship of exercises and interactions, the attention mechanism is implemented. The attention mechanism obtains historical interaction vectors and historical exercise vectors by multi-head attention networks. Our experiments show that MMAKT outperforms other baseline methods on four datasets, and MMAKT can trace the student's knowledge proficiency better.

Our main contributions are summarized as follows:

1. We propose a multi-factor Neural CD-based DKVMN method to trace the student's knowledge proficiency and to construct exercise vectors and interaction vectors, making them as reasonable as cognitive diagnosis.

2. We explicitly enhance the representation of historical information by taking a student's practice history into account using the attention mechanism.

3. Experiments on four real-world online datasets prove that MMAKT outperforms other baseline methods.

2. Related Works

Existing works about student cognitive diagnosis mainly came from educational psychology area. Most of those methods are based on linear handcrafted interaction function such as logistic function or inner production. Wang et al. (2020) proposed Neural Cognitive Diagnosis (Neural CD) framework by incorporating neural networks to model complex non-linear interactions in cognitive diagnosis. In Contrast to traditional models which designed manually with non-neural functions making it hard for them to leverage exercise text content, Neural CD explored the rich information contained in exercise text content for cognitive diagnosis with neural network. The framework of Neural CD is shown in Figure 1.

Dynamic Cognitive Diagnosis is known as knowledge tracing which can dynamically update the student's knowledge proficiency. Deep knowledge tracking (DKT) (Piech et al. (2015)), based on recurrent neural networks (RNN), exploits the utility of latent states in LSTM (Hochreiter and Schmidhuber (1997)) to learn a student's knowledge proficiency.

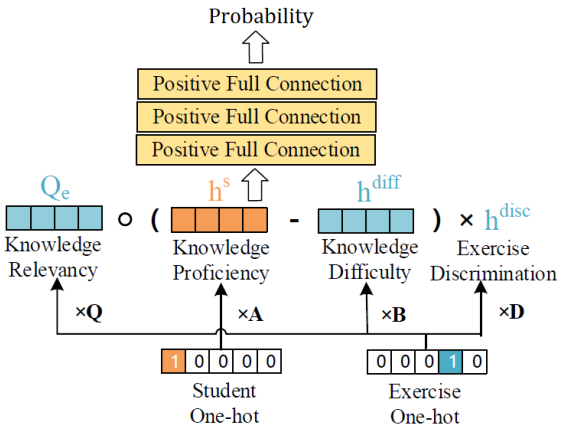

Figure 1: Neural cognitive diagnosis model.Student and exercise vectors are embedded to different factors with neural networks to formulate the interaction function.

Zhang et al. (2017) proposed dynamic key-value memory networks (DKVMN) to trace a student's proficiency of concepts by introducing memory-augmented neural networks (MANN) (Santoro et al. (2016)). Abdelrahman and Wang (2019) used attention mechanism, Sun et al. (2019) added behavior features of students and Ai et al. (2019) considered the containment relationship among concepts, they all improved DKVMN from different sides. Although these DKVMN-based methods simulate the process by which students get proficiency of concepts with elegant methods, their interaction vectors still only require knowledge concepts, and cannot show the relationship between them directly either.

Recently, Pandey and Karypis (2019) proposed the model of Self-Attentive Knowledge Tracing (SAKT). Attention mechanism is more flexible than recurrent neural networks which has been demonstrated in natural language processing tasks (Devlin et al. (2018)). In many sequence-to-sequence predictions tasks, it also has outstanding and effective performance (Kang and McAuley (2018); Zhang et al. (2019)). Similar with the previous methods, SAKT still requires the concepts of exercise to embed vectors, and cannot show a student's proficiency without latent states in RNN. In the aspect of interactions' relationship, SAKT has better performance than the previous methods because it utilizes a single layer of attention mechanism to calculate the weights of the target exercise on historical interactions.

As illustrated above, few deep learning-based KT models have high reasonable interaction function with clear historical relationship. Towards this end, in this paper, we propose a multi-factor memory attentive model which borrows concepts from neural cognitive diagnosis and attention mechanism from transformer (Vaswani et al. (2017)), and combines them with DKVMN. MMAKT could achieve a reasonable interaction form with the assistance of neural CD, and reflects the historical relationship with attention mechanism as well.

3. Model

3.1. Model Overview

The MMAKT method is constructed with two components: Multi-factor based DKVMN layer and Historical Attention layer. Each learner's study record consists of exercises' questions, knowledge concepts and responses at each timestamp.One student's record at timestamp t is defined as (q_t, p_t, r_t) , where $q_t \in \mathbb{N}^+$ is concept index, $p_t \in \mathbb{N}^+$ is question index in concept set C and question set P. $r_t \in \{0, 1\}$ is the student's response. Students are independent of each other. The aim of MMAKT is to make the prediction for the student's response to exercise (q_t, p_t), based on the record S.

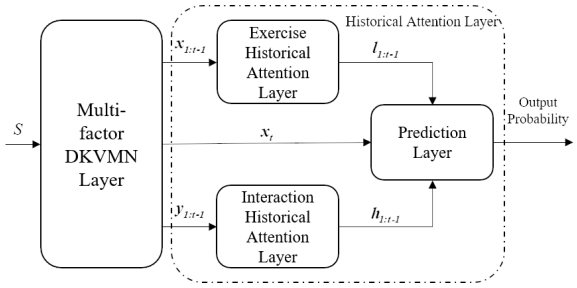

Figure 2: Overview of MMAKT methods.

The MMAKT's structure is shown in Figure 2. The record S is firstly put into the Multi-factor DKVMN layer, where the interaction tuples are embedded into exercise vectors with multi-factor. Then the layer traces the student's proficiency using DKVMN and outputs the interaction vectors in terms of Neural CD in order to make them more reasonable. After that, exercise vectors and interaction vectors are delivered to historical attention layer respectively in order to enhance the representation of historical relationship. Finally, the Prediction layer outputs the predicted response of the target exercise.

3.2. Muiti-factor DKVMN Layer

For a knowledge tracing model, the key is to grasp the student's proficiency. In this paper, MMAKT address the issue by using DKVMN with multi-factors, which is shown in Figure 3. Given a student's interaction tuple (q_i, p_i, r_i) at any timestamp $i \in (1, t)$, MMAKT constructs multi-factors such as concept difficulty $\boldsymbol{b_i} \in \mathbb{R}^d$, knowledge relevancy $\boldsymbol{c_i} \in \mathbb{R}^{\boldsymbol{d}}$, exercise discrimination $dis_i \in \mathbb{R}$ by embedding q_i and p_i with correlated matrices. Then MMAKT embeds response r_i and gets the response vector $\boldsymbol{g_i} \in \mathbb{R}^d$. Different from traditional methods' exercise vector $\boldsymbol{Q}$ and interaction vector $\boldsymbol{QA}$ which is gotten from (q_i, r_i) directly (Piech et al. (2015); Zhang et al. (2017); Pandey and Karypis (2019)), MMAKT, inspired by Neural CD, constructs the exercise vector $\boldsymbol{x_i} \in \mathbb{R}^d$ referring to the factors mentioned above.

$$\boldsymbol{x_i} = \boldsymbol{c_i} \circ \boldsymbol{b_i} \times dis_i \tag{1}$$

where $\circ$ is element-wise product. With exercise vector $\boldsymbol{x_i}$, MMAKT traces the student's proficiency using DKVMN by reading and writing key-value memory networks.

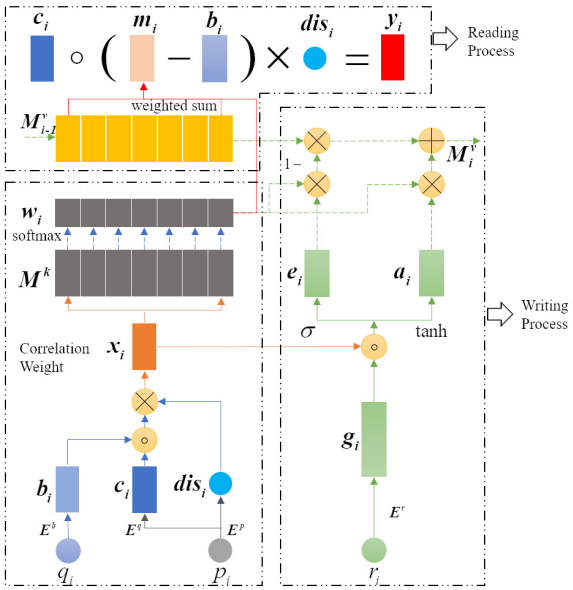

Figure 3: The architecture for Multi-factor DKVMN. The model is drawn at the timestamp i.

3.2.1. Correlation Weights and Reading Process.

To trace the student's knowledge proficiency $\boldsymbol{m_i}$ at timestamp i, MMAKT constructs two memory matrices, $\boldsymbol{M^k}$ and $\boldsymbol{M^v}$ to store key information of concept and value information of the student's mastery. Given an exercise $\boldsymbol{x_i}$, we firstly calculate the correlation weight on the knowledge concept key matrix $\boldsymbol{M^k}$ by taking the softmax activation of the inner product between $\boldsymbol{x_i}$ and each key slot $\boldsymbol{M_j^k}$.

$$\boldsymbol{w_{i,j}} = \text{Softmax}(\boldsymbol{x_i}^T \boldsymbol{M_j^k}) \tag{2}$$

where $\boldsymbol{M_j^k} \in \mathbb{R}^d$ is the j^{th} slot of the knowledge concept key matrix $\boldsymbol{M^k} \in \mathbb{R}^{|C| \times d}$, and $w_{i,j}$ is the j^{th} value of $\boldsymbol{w_i} \in \mathbb{R}^{|C|}$ representing the correlation weight between the exercise and each latent concept.

Then MMAKT gets the student's knowledge proficiency $\boldsymbol{m_i} \in \mathbb{R}^d$ from the value matrix $\boldsymbol{M_{i-1}^v}$ with respect to $\boldsymbol{w_i}$.

$$\boldsymbol{m_i} = \sum_{j=1}^{|C|} w_{i,j} \boldsymbol{M_{i-1,j}^v} \tag{3}$$

where $\boldsymbol{M_{i-1,j}^v} \in \mathbb{R}^d$ is the j^{th} slot in the knowledge concept value matrix $\boldsymbol{M_{i-1}^v} \in \mathbb{R}^{|C| \times d}$.

The interaction vector $\boldsymbol{y_i} \in \mathbb{R}^d$ is then constructed with $\boldsymbol{m_i}$ and the other factors mentioned above in terms of the reasonable method Neural CD.

$$\boldsymbol{y_i} = \boldsymbol{c_i} \circ (\boldsymbol{m_i} - \boldsymbol{b_i}) \times dis_i \tag{4}$$

In the vanilla DKVMN, the student's proficiency vector $\boldsymbol{m_i}$ is directly used to make response prediction,ignoring many factors in the realistic learning process. Therefore, MMAKT refers to cognitive diagnosis for simulating the situation of a student's solving a exercise problem.

3.2.2. Writing Porcess

In order to make the next computation at timestamp $i+1$, $\boldsymbol{M_{i-1}^v}$ needs to be updated to $\boldsymbol{M_i^v}$. Unlike the knowledge growth vector in vanilla DKVMN, MMAKT constructs the update vector with the element-wise product between the exercise vector $\boldsymbol{x_i}$ and the response vector $\boldsymbol{g_i}$ in order to match with $\boldsymbol{x_i}$. The following update process is the same as that of DKVMN, including erase subprocess and add subprocess.The eraser vector $\boldsymbol{e_i} \in \mathbb{R}^d$ and the add vector $\boldsymbol{a_i} \in \mathbb{R}^d$ are calculated as follows:

$$\boldsymbol{e_i} = \text{Sigmoid}(\boldsymbol{W_e}(\boldsymbol{x_i} \circ \boldsymbol{g_i}) + \boldsymbol{b_e}) \tag{5}$$

where $\boldsymbol{W_e} \in \mathbb{R}^{d \times d}$, $\boldsymbol{b_e} \in \mathbb{R}^d$ are parameter vectors.

$$\boldsymbol{a_i} = \text{Tanh}(\boldsymbol{W_a}(\boldsymbol{x_i} \circ \boldsymbol{g_i}) + \boldsymbol{b_a}) \tag{6}$$

where $\boldsymbol{W_a} \in \mathbb{R}^{d \times d}$, $\boldsymbol{b_a} \in \mathbb{R}^d$ are parameter vectors. Then, $\boldsymbol{M_{i-1}^v}$ is updated to $\boldsymbol{M_i^v}$ with $\boldsymbol{e_i}$ and $\boldsymbol{a_i}$:

$$\boldsymbol{M_{i,j}^v} = \boldsymbol{M_{i-1,j}^v}(\boldsymbol{1} - w_{i,j}\boldsymbol{e_i} + w_{i,j}\boldsymbol{a_i}) \tag{7}$$

where $\boldsymbol{1}$ is d dimension one vector.

3.3. Historical Attention Layer

The predicted probability is usually made by a fully connected layer after the RNN module in DKT and DKVMN, ignoring the historical relationship between exercises or interactions and themselves. To deal with that, MMAKT uses two multi-head attention networks to represent their historical relationship, and makes prediction with another one. We provide the explanation of the multi-head attention networks in the next subsection.

3.3.1. Multi-head Attention Networks

As illustrated in Transformer(Vaswani et al. (2017)), the multi-head attention network gets the relevance of the value V_{in} to the corresponding query Q_{in} by the dot-product between the query Q_{in} and the key K_{in} in several parallel single self-attention layers which are projected with different matrices. In order to avoid the influence from the future information, the network utilizes a masking mechanism which replaces upper triangular part of the product matrix with $-\infty$ in order to zero out the attention weights of the subsequent positions.As shown in Figure 4, the single self-attention head is,

$$head_i = \text{Softmax}(\text{Mask}(\frac{Q_{in}\boldsymbol{W_i^Q}(K_{in}\boldsymbol{W_i^K})^T}{\sqrt{d}}))V_{in}\boldsymbol{W_i^V} \tag{8}$$

where $\boldsymbol{W_i^Q}$, $\boldsymbol{W_i^K}$, $\boldsymbol{W_i^V}$ are projection matrices.

The final output of the multi-head attention networks is the concatenated tensor of heads attention heads multiplied by $\boldsymbol{W^O}$.

$$\text{MultiHead}(Q_{in}, K_{in}, V_{in}) = \text{Concat}(head_1, ..., head_{heads})\boldsymbol{W^O} \tag{9}$$

Due to the linear transformation, the self-attention layer applies position-wise feed-forward networks to increase the non-linearty of the model.

$$\text{FFN}(\boldsymbol{M_H}) = \text{ReLU}(\boldsymbol{M_H W_1^{FF}} + \boldsymbol{b_1^{FF}})\boldsymbol{W_2^{FF}} + \boldsymbol{b_2^{FF}} \tag{10}$$

where $\boldsymbol{M_H} = \text{Multihead}(Q_{in}, K_{in}, V_{in})$ and $\boldsymbol{W_1^{FF}}, \boldsymbol{W_2^{FF}}, \boldsymbol{b_1^{FF}}$ and $\boldsymbol{b_2^{FF}}$ are weight matrices and bias vectors.

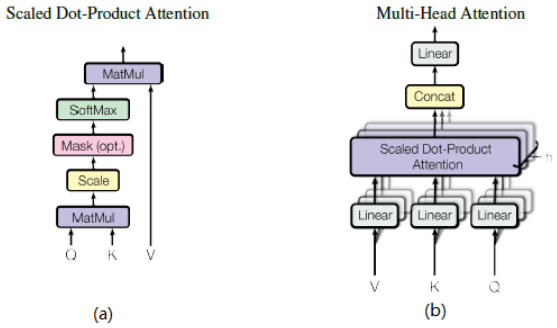

Figure 4: (a) Network of a masked dot-product attention. (b) Network of a multi-head attention. Each multi-head attention consists of several attention layers running in parallel.

3.3.2. Historical Attention Layer

MMAKT utilizes two masked muti-head attention networks mentioned above to calculate the historical vectors of the exercise vector sequence $\boldsymbol{x_{1:t-1}}$ and the interaction vector sequence $\boldsymbol{y_{1:t-1}}$ for improving the expression of the historical relationship in MMAKT. For the interaction vector sequence $\boldsymbol{y_{1:t-1}}$, since a student's knowledge proficiency evolves gradually and steadily with time, the knowledge proficiency at a particular time instance should not show wavy transitions(Yeung and Yeung (2018)). $\boldsymbol{y_i}$ needs to be position-encoded as follows:

$$\hat{\boldsymbol{y}}_{\boldsymbol{i}} = \boldsymbol{y_i} + \boldsymbol{M_i^P} \tag{11}$$

where $\boldsymbol{M_i^P} \in \mathbb{R}^d$ represents the position information in interaction record sequence, and $\hat{\boldsymbol{y}}_{\boldsymbol{i}}$ is the position-encoded interaction vector. The model inputs $\hat{\boldsymbol{y}}_{\boldsymbol{1:t-1}}$ as queries, keys and values, and finally gets the historical interaction vector sequence $\boldsymbol{h_{1:t-1}}$:

$$\hat{\boldsymbol{h}}_{\boldsymbol{1:t-1}} = \text{MultiHead}(Q_{in} = \hat{\boldsymbol{y}}_{\boldsymbol{1:t-1}}, K_{in} = \hat{\boldsymbol{y}}_{\boldsymbol{1:t-1}}, V_{in} = \hat{\boldsymbol{y}}_{\boldsymbol{1:t-1}}) \tag{12}$$

$$\boldsymbol{h_{1:t-1}} = \text{FFN}(\hat{\boldsymbol{h}}_{\boldsymbol{1:t-1}}) \tag{13}$$

For the exercise vector sequence $\boldsymbol{x_{1:t-1}}$, the model inputs $\boldsymbol{x_{1:t-1}}$ as queries, keys and values, and gets the historical exercise vector sequence $\boldsymbol{l_{1:t-1}}$.

$$\hat{\boldsymbol{l}}_{\boldsymbol{1:t-1}} = \text{MultiHead}(Q_{in} = \boldsymbol{x_{1:t-1}}, K_{in} = \boldsymbol{x_{1:t-1}}, V_{in} = \boldsymbol{x_{1:t-1}}) \tag{14}$$

$$\boldsymbol{l_{1:t-1}} = \text{FFN}(\hat{\boldsymbol{l}}_{\boldsymbol{1:t-1}}) \tag{15}$$

3.3.3. Prediction Layer

After the historical attention layer, MMAKT utilizes the historical sequences of exercises and interactions to calculate the prediction vector $\boldsymbol{f_t}$ in prediction layer. The prediction layer inputs the target exercise $\boldsymbol{x_t}$ as query, the historical exercise sequence $\boldsymbol{l_{1:t-1}}$ as keys, the historical interaction sequence $\boldsymbol{h_{1:t-1}}$ as values, and outputs the prediction vector $\boldsymbol{f_t}$:

$$\hat{\boldsymbol{f}}_{\boldsymbol{t}} = \text{MultiHead}(Q_{in} = \boldsymbol{x_t}, K_{in} = \boldsymbol{l_{1:t-1}}, V_{in} = \boldsymbol{h_{1:t-1}}) \tag{16}$$

$$\boldsymbol{f_t} = \text{FFN}(\hat{\boldsymbol{f}}_{\boldsymbol{t}}) \tag{17}$$

It is noteworthy that we must not use the response at timestamp t to predict the response of $\boldsymbol{x_t}$. So, we can only use $\boldsymbol{h_{1:t-1}}$ and $\boldsymbol{l_{1:t-1}}$ sequence before timestamp $i-1$. Compared with the multi-head attention layers in the historical attention layer, the predication layer's Q_{in}, K_{in}, V_{in} are totally different because MMAKT must trace the attention between the target exercise and the historical exercises in order to get the student's historical proficiency from the historical interaction sequence.

Then, MMATK converts $\boldsymbol{f_t}$ to the response o_t via a sigmoid activated fully connected layer.

$$o_t = \text{sigmoid}(\boldsymbol{f_t W_p} + \boldsymbol{b_p}) \tag{18}$$

where $\boldsymbol{W_p} \in \mathbb{R}^d$, $\boldsymbol{b_p} \in \mathbb{R}$ are parameters in FC layer.

Finally, all the trainable parameters in MMAKT can be learned by minimizing the cross-entropy loss between o_t and r_t.

$$\mathcal{L} = -\sum_{i \in |S|} (r_t \log(o_t) + (1 - r_t) \log(1 - o_t)) \tag{19}$$

4. Experiments

4.1. Datasets and Evaluation Metric

MMAKT is evaluated on four real-world datasets. ASSIST2009 and ASSIST2017 contain exercises' problem information, and the others not. The ASSISTments datasets were collected from an online tutoring platform. The Statics2011 dataset was collected from a college-level engineering course on statics.

ASSISTments2009: Dataset contains 4151 students, 110 concepts, 16891 problems and 325637 interactions.

ASSISTments2015: Dataset contains 19840 students, 100 concepts, 942816 interactions.

ASSISTments2017: Dataset contains 1709 students, 102 concepts, 3162 problems and 942816 interactions.

Statics2011: Dataset contains 333 students, 1223 concepts, 189297 interactions.

The Area Under Curve (AUC) and the Accuracy (ACC) are used for evaluation metrics. AUC is defined as the area under the receiver operating characteristics curve, representing the predictive performance of the model. Higher AUC and ACC values indicate better performance of the model.

4.2. Baseline Methods and Approach

We compare MMAKT against the baseline methods, DKT, DKVMN and SAKT. The parameters are set as follows:

DKT: Hyperparameters are set following Piech et al. (2015). The embedding dimension is 50, and the hidden dimension is 200 in RNN. The learning rate is 0.001 with Adam optimizer.

DKVMN: Hyperparameters are set following Zhang et al. (2017). The memory size is 50, and memory dimension is 200. The learning rate is 0.001 with Adam optimizer.

SAKT: Hyperparameters are set following Pandey and Karypis (2019). The embedding dimension is 50. The learning rate is 0.001 with Adam optimizer.

MMAKT: For different four datasets, the memory sizes of $\boldsymbol{M^k}$ and $\boldsymbol{M^v}$ are set according to the number of concepts in each dataset. The batch size is 30, and the learning rate is 0.01 with Adam optimizer. Hyperparameter d is determined by comparing AUC values. The results of testing are shown in Table 1. We can find in the table that the AUC values are higher than the others when $d = 24$. It can be seen that when d is set too low, the performance of the model decreases; when d is set too high, there are too many parameters in the model, which easily lead to overfitting. So, d should be chosen according to the result of experiments.

Table 1: AUC results with different d

ASSIST2009		ASSIST2015		ASSIST2017		STATICS2011	
d	AUC	d	AUC	d	AUC	d	AUC
16	0.750 1	16	0.708 6	16	0.716 7	16	0.817 1
24	**0.764 4**	**24**	**0.723 1**	**24**	**0.724 8**	**24**	**0.818 6**
32	0.751 9	32	0.704 2	32	0.720 4	32	0.817 9

4.3. Results and Discussion

In this paper, for each dataset, 20% learners are used as the test set, 60% are used as the training set and 20% are used as the validation set to adjust hyperparameters and early stop.

Table 2 shows the results of AUCs and ACCs of MMAKT and the other 3 baselines on 4 datasets. The results show that MMAKT generally outperforms the other 3 models. SAKT performs the worst among models because it embeds exercises and interactions only in terms of knowledge concepts, and neglects the student's knowledge proficiency and the historical relationship of exercises and interactions. DKT uses latent states to simulate the student's proficiency, which cannot model each concept directly, making lower performance

than DKVMN. Both DKT and DKVMN have the weakness on the expression of historical relationship, which leads their inferior to MMAKT. MMAKT not only restructures the vectors of exercises and interactions with neural CD and DKVMN, but also considers the historical relationship of exercises and interactions to enrich the information of the prediction vector, making better performance than the other models.

Table 2: Prediction results of models

Model	ASSIST2009		ASSIST2015		ASSIST2017		STATICS2011	
	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC
DKT	0.736 4	**0.736 0**	0.700 7	0.744 3	0.710 4	0.688 3	0.786 8	0.780 9
DKVMN	0.745 9	0.726 8	0.717 4	0.747 4	0.679 8	0.670 6	0.785 7	0.803 8
SAKT	0.723 7	0.705 9	0.668 1	0.750 1	0.652 3	0.659 7	0.793 1	0.796 7
MMAKT	**0.764 4**	0.722 0			**0.724 8**	**0.696 6**		
MMAKT-N	0.750 0	0.718 7	**0.723 1**	**0.754 1**	0.681 4	0.669 9	**0.818 6**	**0.809 1**

In addition, we also conduct experiments without problem information with MMAKT-N. The results show the performance of MMAKT-N is slightly lower than MMAKT, but still higher than the other three models.

Figure 5 shows the AUC plots of MMAKT and the other 3 methods on the ASSIST2009's training set and validation set in 50 epochs. The results show that SAKT and DKT have obviously overfitting problem. DKVMN performs well in preventing overfitting, but its AUC performance is inferior to that of MMAKT because it only makes predictions based on the student's knowledge states without taking into account the historical relationship. MMAKT not only has excellent performance on the validation set, but also remains stable with training, and does not appear the same situation as DKT and SAKT, where AUC decreases due to overfitting. All above indicate the effectiveness of MMAKT.

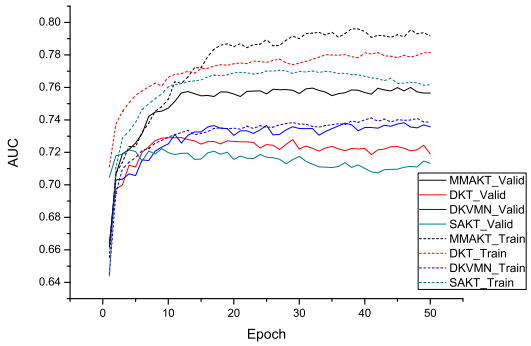

Figure 5: Performance on ASSIST2009 training set and validation set.

4.4. Visualizing Parameters

We choose a student in ASSIST2009 randomly, and visualize learning records over a period time. In Figure 6, the x-axis represents the student's interaction records, where q_t is exercise's concept, and r_t is student's response in (q_t, r_t). The y-axis represents four concepts which these records contain. From the data we can see that after the student's correct answer about concept 31 at timestamp 4, the student's knowledge proficiency improved correspondingly, and at timestamp 11, the student's knowledge proficiency declined after the incorrect answer. This indicates that MMAKT can dynamically update students' knowledge proficiency according to students' answers.

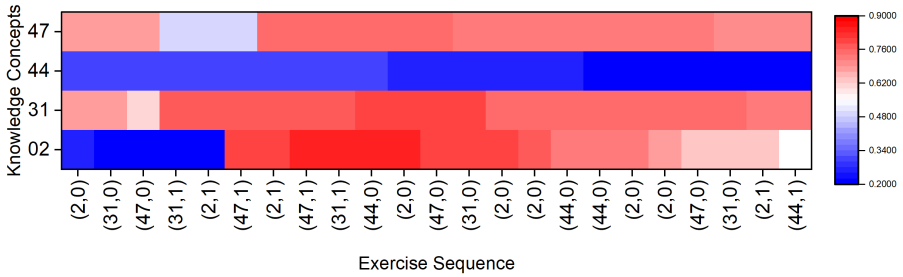

Figure 6: Knowledge proficiency output result of MMAKT.

Then we choose records about the concepts 2 at timestamp 1,5,7 and 11. And the problem information p_t is added to form tuple (q_t, p_t, r_t). The plots of exercise vectors $\boldsymbol{x}$, interaction vectors $\boldsymbol{y}$ in Multi-factor DKVMN layer are shown in Figure 7. We can see the comparison of exercise vectors and interaction vectors between MMAKT and DKT. The x-axis represents the hyperparameter d, and the y-axis represents interaction' records. In plot X, though all the records are about concept 2, vectors are totally different since their problem information is different. On the contrary, DKT's exercise vectors Q which are embedded from q_t are all the same with each other. In plot Y, we can see the interaction vectors $\boldsymbol{y_t}$ which are calculated by equation 4 not only contain the information of $\boldsymbol{x_t}$, but also that of $\boldsymbol{m_t}$, yet the interaction vectors $\boldsymbol{QA}$ in DKT are still unchangeable.

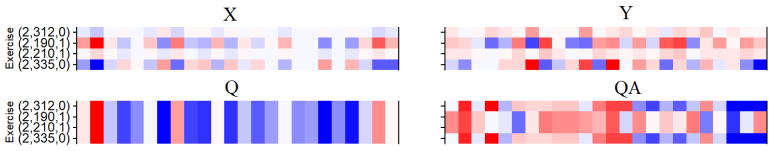

Figure 7: Comparison between embedding vectors in MMAKT and DKT.

There are 3 multi-head attention networks in the historical attention layers and the prediction layer. We take the attention weights of the first 10 exercises to make illustration. In Figure 8, the x-axis represents the past exercises, and the y-axis represents the target exercises. In the historical layer, target exercises are same as past exercises, but in prediction

layer, target exercises surpass past exercises one timestamp because we cannot use exercises' their own answers to predict responses of themselves.

In Figure 8, though two historical attention layers correspond to the same exercises, their attention weights are different. Take the exercise 7 as example, in the exercise historical attention layer, the weights between the exercise 1 and 6 are significantly higher than others. Nevertheless, in the interaction historical attention layer, the exercise 7 pays more attention on exercise 3 and 5. The reason is $\boldsymbol{y_t}$ contains the student's knowledge proficiency besides $\boldsymbol{x_t}$. For the prediction layer, attention weights are between $\boldsymbol{x_t}$ and $\boldsymbol{l_{t-1}}$ to make prediction based on the historical vectors. Compared with the method in SAKT, whose weights are between knowledge concept q_t and interaction (q_t, a_t), MMAKT is more realistic for students' learning, and the experiments show that MMAKT performs better.

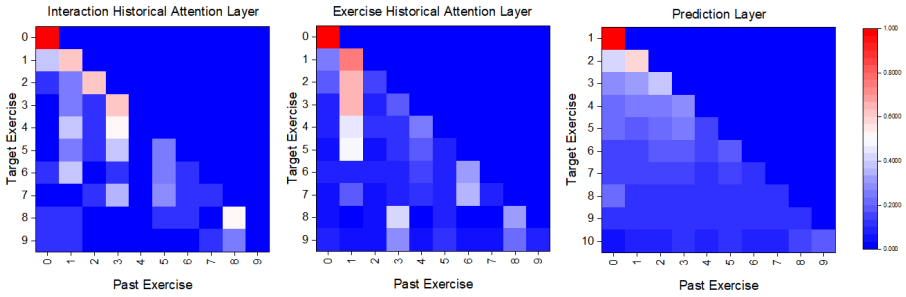

Figure 8: Attention weights in Historical Attention Layer and Prediction Layer.

5. Conclusions and Future Works

In this paper, we propose MMAKT which refers Neural CD to construct exercise vectors and interaction vectors, and traces the student's proficiency with DKVMN. Besides, MMAKT uses attention mechanisms to enhance the representation of historical information and predicts the student's future response as well. The experiments show that our model outperforms DKT, DKVMN and SAKT.

In the future research, we will explore the following aspects:

1. For the embedding, we will consider adding other realistic factors, such as the priori, posteriori relation of concepts and students' effectiveness of learning.

2. For the attention mechanism, we will consider adding more factors, such as time interval, etc.

Acknowledgments

Foundation item: National Natural Science Foundation of China (U1811261)

References

Ghodai Abdelrahman and Qing Wang. Knowledge tracing with sequential key-value memory networks. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 175–184, 2019.

Fangzhe Ai, Yishuai Chen, Yuchun Guo, Yongxiang Zhao, Zhenzhu Wang, Guowei Fu, and Guangyan Wang. Concept-aware deep knowledge tracing and exercise recommendation in an online learning system. *International Educational Data Mining Society*, 2019.

Ryan SJD Baker and Kalina Yacef. The state of educational data mining in 2009: A review and future visions. *JEDM— Journal of Educational Data Mining*, 1(1):3–17, 2009.

Albert T Corbett and John R Anderson. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278, 1994.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 197–206. IEEE, 2018.

Shalini Pandey and George Karypis. A self-attentive model for knowledge tracing. *arXiv preprint arXiv:1907.06837*, 2019.

Chris Piech, Jonathan Spencer, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas Guibas, and Jascha Sohl-Dickstein. Deep knowledge tracing. *arXiv preprint arXiv:1506.05908*, 2015.

Georg Rasch. *Probabilistic models for some intelligence and attainment tests.* ERIC, 1993.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pages 1842–1850. PMLR, 2016.

Xia Sun, Xu Zhao, Yuan Ma, Xinrui Yuan, Feijuan He, and Jun Feng. Muti-behavior features based knowledge tracking using decision tree improved dkvmn. In *Proceedings of the ACM Turing Celebration Conference-China*, pages 1–6, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

Fei Wang, Qi Liu, Enhong Chen, Zhenya Huang, Yuying Chen, Yu Yin, Zai Huang, and Shijin Wang. Neural cognitive diagnosis for intelligent education systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6153–6161, 2020.

Chun-Kit Yeung and Dit-Yan Yeung. Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*, pages 1–10, 2018.

Michael V Yudelson, Kenneth R Koedinger, and Geoffrey J Gordon. Individualized bayesian knowledge tracing models. In *International conference on artificial intelligence in education*, pages 171–180. Springer, 2013.

Jiani Zhang, Xingjian Shi, Irwin King, and Dit-Yan Yeung. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th international conference on World Wide Web*, pages 765–774, 2017.

Shuai Zhang, Yi Tay, Lina Yao, Aixin Sun, and Jake An. Next item recommendation with self-attentive metric learning. In *Thirty-Third AAAI Conference on Artificial Intelligence*, volume 9, 2019.