Computing and Informatics, Vol. 44, 2025, 1468–1494, doi: 10.31577/cai_2025_6_1468

BEHAVIOR-ENHANCED KNOWLEDGE TRACING MODEL IN INTELLIGENT TUTORING SYSTEMS

Si'ao Sun

School of Artificial Intelligence
Shandong Xiandai University
20288 Jing Shi East Road
250104 Jinan, China
e-mail: sun_siao@163.com

Liqing Qiu

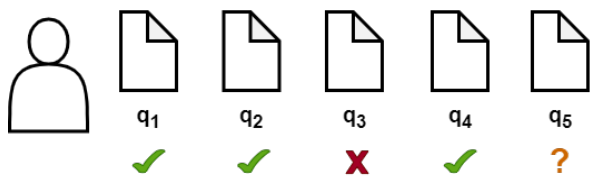
Institute of Computer Science and Engineering
Shandong University of Science and Technology
579 Qianwangang Road, Huangdao District
266590 Qingdao, China
e-mail: 2470772625@qq.com

Abstract. Knowledge tracing endeavors to track shifts in students' knowledge states throughout the learning process by utilizing students' work records, enabling the prediction of their responses to the next question. As knowledge tracing advances, more information can be mined from the datasets to improve the model. However, most existing models only consider the correctness of the question and the concept as input, ignoring much of the other information gathered by intelligent tutoring systems (ITS). Therefore, this paper presents a Behavior-Enhanced Knowledge Tracing Model (BEKT). The Behavior-Enhanced Knowledge Tracing Model first selects multi-dimensional features by calculating the mutual information. The mutual information value shows that the most relevant features to the prediction target are student behavior information, including the count of hints, attempts made, and time taken to answer questions. Then, the model proposes a cross-feature fusion method to change the students' knowledge mastery state. Finally, a multi-layer perceptron is applied to integrate student behavior features to enhance knowledge state representation. Compared to existing models, BEKT more accurately captures changes in knowledge state, improving model performance.

Keywords: Online education, intelligent system, knowledge tracing, attention mechanism, feature fusion

1 INTRODUCTION

Over the recent years, as online education platforms have made significant progress and become more widely accessible, more and more people have chosen online courses. Online education platforms can not only expand learning resources but also track the users' learning trajectories. To fully leverage the benefits of e-learning or online learning, the online education platform utilizes students' test records to reveal their knowledge mastery levels, recommend suitable topics, and offer personalized guidance [1]. Knowledge tracing is applied to solve this problem, which aims to build a model of students' knowledge state [2], to judge students' mastery of each knowledge point. Because it is difficult to directly obtain the changing state of knowledge, knowledge tracing models often employ an alternative solution that allows the model to predict the probability of getting the next question right. Figure 1 given the historical interaction records of students $S = (s_1, s_2, \ldots, s_t)$, forecast the probability of a student providing a correct response to a new exercise. Student interaction is defined as a question-response tuple $s_t = (q_t, r_t)$, where q_t is the number of the knowledge concept involved in the question answered by the student at time t, and r_t indicates whether the student's response is correct at time t. The answer is wrong when r_t is 0, and when r_t is 1, the answer is correct.



questions	concepts
q1	proportion
q2	division fractions
q3	volume sphere
q4	absolute value
q5	volume pyramid

Figure 1. The BEKT model's objective is to forecast the likelihood of a correct response to question q_5

Traditional knowledge tracing models usually take question-response pairs as input while ignoring other available information in the datasets. For instance, Deep Knowledge Tracing (DKT) [3] utilizes student interaction tuples as inputs to the recurrent neural network (RNN). To properly instruct an RNN using student interactions, a fundamental step involves converting these interactions into a sequence of fixed-length input vectors. The input vector is configured as a one-hot representation of the student $s_t = (q_t, r_t)$ for datasets with several unique exercises. Compressed sensing is utilized for large feature spaces to reduce the dimensionality of one-hot encodings. The RNN's hidden state functions as the student's state

of knowledge, while the RNN's output represents the predicted probability of the student correctly answering specific questions. Moreover, Self-Attentive Knowledge Tracing (SAKT) [4] utilizes student interaction tuples as inputs to compute attention weights. The interaction tuple $s_t = (q_t, r_t)$ is represented in the model as a scalar value $d_t = q_t + r_t \cdot E$, where E denotes the total number of exercises. Consequently, an element within the interaction sequence has a range of $2E$ possible values, whereas elements in the exercise sequence have E potential values.

With the continuous improvement of knowledge tracing models, several approaches have been proposed to leverage the features present in the datasets. For instance, Deep Knowledge Tracing with rich features (DKT-FE) [5] incorporates processed features with interactive embeddings. Additionally, both Deep Knowledge Tracing with decision trees (DKT-DT) [6] and Dynamic Key-Value Memory Networks with decision trees (DKVMN-DT) [7] utilize decision trees for preprocessing additional features. Furthermore, DKT + Forgetting [8] introduces time information to simulate the forgetting process during students' question-solving activities. Colearn [9] takes hint-taking behavior as a subtask of knowledge tracing. In conclusion, the effects of these improved models have been improved.

This article proposes the Behavior-Enhanced Knowledge Tracing Model (BEKT), inspired by the abovementioned advancements. The BEKT model aims to understand better how students' behaviors affect their knowledge state by incorporating student behavior features. First, the BEKT model utilizes the mutual information method to select features that exhibit the highest correlation with the prediction target. Through analysis, it is discovered that among the features with high mutual information values, including request prompt count, attempt count, and response time, all correspond to student behavior features. Second, the BEKT model takes into account how to handle the features. This model holds that students' knowledge state is affected by their behavior. Therefore, introduces a cross-feature approach that integrates student behavior features in a combined manner. Subsequently, the knowledge state fluctuations generated by these cross-features are captured using the tanh function. Lastly, the BEKT model employs a multi-layer perceptron (MLP) to fuse student behavior features. The fused features are then concatenated with the knowledge state to enhance its representation.

The BEKT model is evaluated on large-scale datasets collected from an online learning platform, and its performance is compared with other knowledge tracing models. The results show that the BEKT model is improved in tracing the student's knowledge states and predicting the student's future performance.

Our main contributions are summarized as follows:

1. By analyzing the datasets using the mutual information method, it is found that the impact of student behavior features on student responses is greater than the influence of question features on student responses.

2. This paper proposes a cross-feature method that integrates student behavior features. By utilizing this cross-feature method, students' knowledge states can be modified, resulting in more accurate and interpretable representations of the knowledge state.
3. By fusing student behavior features through multi-layer perceptron, feature dimension can be reduced without losing feature information as much as possible. The fused features and knowledge states are combined to strengthen the representation of knowledge states.

2 RELATED WORKS

There are two primary classifications among the current knowledge tracing models. The initial class comprises probability-centric models, exemplified by Bayesian Knowledge Tracing (BKT) [2]. The second class encompasses models rooted in deep learning, like Deep Knowledge Tracing (DKT) [3], Dynamic Key-Value Memory Networks (DKVMN) [10], and Attentive Knowledge Tracing (AKT) [11].

As a prominent knowledge tracing model, BKT employs the Hidden Markov Model (HMM) to adjust the knowledge state through an analysis of student's performance in exercises. BKT characterizes the potential knowledge state of students using binary variables, representing whether they have mastered a particular knowledge concept (KC) [12]. Despite the significant achievements of BKT within the domain of knowledge tracing, certain issues persist. First of all, the correspondence between variables and knowledge concepts is fuzzy. It cannot achieve one-to-one correspondence, and BKT cannot simulate a question requiring several knowledge concepts (KCS). Second, how each KC is modeled separately makes it impossible for BKT to capture the relationship between different KCS. Meanwhile, Probability-based knowledge tracing models often need input from domain experts to set parameters and probability distributions, limiting adaptability to different domains and requiring customization for specific contexts. Probability models often rely on certain assumptions, such as assuming independence of student knowledge states or fixed difficulty levels of items. These assumptions may not align perfectly with real-world scenarios, leading to potential biases in the predictions of the model.

DKT pioneers the way in incorporating deep learning methods into the field of knowledge tracing, leveraging the versatility of Recurrent Neural Networks (RNNs) to overcome the constraints of skill separation and binary state assumption, thus enhancing the model's practicality. The model employs the RNN's hidden state as a summarization of past input sequences, assuming that this hidden state represents the level of comprehension for each concept. Subsequently, it analyzes the students' knowledge state and utilizes this information to predict their exercise outcomes. Because DKT is based on RNN, there is a long-term dependency problem, which makes DKT unable to utilize long sequences of inputs.

DKVMN is a variant of a memory enhancement neural network (MANN). This model builds upon the traditional neural network structure by adding a memory

module and corresponding read and write mechanism. DKVMN added static matrix and dynamic matrix as external memory, thus eliminating the relationship between trainable parameters and model memory ability. These modifications facilitate the modeling of long-distance dependencies. The DKVMN model, like other deep learning-based models, shares the common issue of being regarded as a black box model, making it challenging to interpret the reasoning behind its predictions. Explaining the inference process of student's learning progress and knowledge states is crucial for teachers and students to understand. However, the limited interpretability found in deep learning models might constrain their practical use within educational contexts.

AKT leverages the attention-mechanism-based transformer model with various novel and interpretable model components. The attention mechanism, as a self-explanatory model, partially addresses the issue of poor interpretability in deep learning. By employing a self-attention mechanism, the AKT model gains the ability to focus on various segments of the input sequence, effectively capturing long-range dependencies. AKT employs a unique monotone attention mechanism, wherein the scaled dot-product attention is modified by multiplying it with an exponential decay term based on relative time steps to compute attention weights. In addition, the Rasch model is used to regulate question and response embedding. It can capture variations among identical conceptual questions without necessitating an excessive number of parameters. However, AKT solely relies on exercise labels and correctness labels as inputs, disregarding crucial factors such as students' behavior features. While the AKT model possesses a certain level of interpretability, it overlooks the interpretability of knowledge states. This limitation can lead to inaccurate modeling of a student's knowledge state by AKT, consequently affecting its predictive performance for future exercise outcomes.

3 PROPOSED MODEL

This section will provide an elaborate overview of the BEKT framework, showcasing the overall architecture depicted in Figure 2. The BEKT model is composed of five components, which include an embedding layer, two self-focused encoders (one pertains to questions, while the other relates to acquiring knowledge), a knowledge retrieval module based on attention, a knowledge state fluctuation module, and a feedforward response prediction module. To provide better clarity, Table 1 presents an overview of the notations used in this study.

3.1 Embedding Layer

Some KT models address the issue of data sparsity by utilizing concepts to index questions. However, this approach fails to consider the distinctions between different questions that revolve around the same knowledge concept. To solve this problem, the Rasch model is employed by BEKT for crafting question embeddings as well as question-response embeddings.

Notation	Description
$S = (s_1, \ldots, s_t)$	The historical interaction records of students.
s_t	Student interaction at time t.
q_t	The number of the knowledge concepts involved in the question answered by the student at time t.
r_t	It indicates whether the student's response is correct at time t.
x_t	Question embedding.
y_t	Question-response embedding.
$w_{t,\tau}$	The monotonic attention weight.
$\hat{x}_t$	Context-aware question embedding.
$\hat{y}_{t-1}$	Context-aware question-response embedding.
h_t^{kr}	The knowledge state is output by the knowledge retriever.
x_i	The behavior features of students, $i = 1, 2, 3$.
v_t	Student interaction at time t.
h_t^c	Cross features produce knowledge state fluctuations.
h_t	The state of knowledge after being affected by cross features.
h_t^{mlp}	The knowledge state representation was enhanced using the MLP feature fusion method.
h_t^p	Input to the prediction layer.
f_{enc1}	Question encoder.
f_{enc2}	Knowledge encoder.
f_{kr}	Knowledge retriever.
f_{plus}	Add operation.

Table 1. Notations

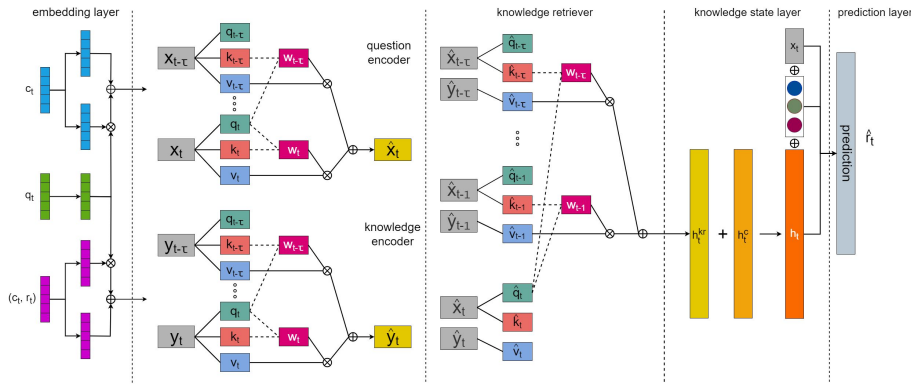


Figure 2. Architecture of the model under consideration. To enhance readability, dashed lines visually differentiate the model's five components.

Question Embedding. Modeling the question embedding x_t from concept c_t at time step t can be expressed as:

$$x_t = c_t + \mu_t \cdot d_t. \tag{1}$$

The $c_t \in R^d$ represents the concept embedding related to question q_t at time step t, reflecting the intrinsic concept representation. The $\mu_t \in R^d$ denotes the difficulty vector for question q_t which can distinguish between different questions with the same concept and $d_t \in R^d$ represents the vector that encapsulates the variations of questions related to the concept [4].

Question-Response Embedding. Construct the question-response embedding with concept and difficulty parameters. Question-response embedding y_t are constructed as follows:

$$y_t = e_t + \mu_t \cdot g_t. \tag{2}$$

The $e_t \in R^d$ corresponds to the embedding of the concept-response pair, and the vector $g_t \in R^d$ represents variations in different concepts and responses. The model employs an identical difficulty parameter μ_t for the question embedding and the question-response pair embedding. By utilizing question embedding, which contains richer information than concept embedding, the BEKT model's performance can be enhanced.

3.2 The Monotonic Attention Mechanism

The BEKT model considers that the question (or interaction) with a longer distance is less critical to the current question (or interaction). The question (or interaction) with a shorter distance is more important than the current question (or interaction). Therefore, BEKT uses a monotonic attention mechanism instead of a scale dot product attention mechanism. Monotonic attention adds an influencing factor measured by relative distance to the weight calculation process, and the influence factor gradually decreases with the distance increase, so that the distant element can get less attention. This method of calculating weights for monotonous attention is called context-aware.

The two encoders use a self-attention mechanism, and the knowledge retriever uses an attention mechanism. The question encoder and knowledge encoder use question embedding and question-response embedding as input to generate query, key, and value, respectively. The knowledge retriever uses the output of the question encoder as input to generate a query and a key while using the output of the knowledge encoder as input to generate a value. The following equations yield the query, key, and value:

$$Q = IW^Q, \quad K = IW^K, \quad V = IW^V, \tag{3}$$

where the matrices $W^Q \in R^{d\times d}$, $W^K \in R^{d\times d}$ and $W^V \in R^{d\times d}$ serve as projection matrices that transform input $I \in R^{N\times d}$ into the respective query $Q \in R^{N\times d}$, key

$K \in R^{N\times d}$ and value $V \in R^{N\times d}$ representations.

The computation of the monotonic attention weights $w_{t,\tau}$ employs the softmax function, expressed as follows:

$$w_{t,\tau} = \text{softmax}\left(\text{mask}\left(e^{-\theta\Delta_{t\tau}}\frac{{Q_t}^T K_\tau}{\sqrt{d}}\right)\right), \tag{4}$$

$$\text{Attention} = w_{t,\tau}\cdot V, \tag{5}$$

where θ is the learnable hyperparameter, $e^{-\theta\Delta_{t\tau}}$ is the influence factor, and d is the dimension of the matrix, which is divided by $\sqrt{d}$ to avoid too small softmax gradient due to too large inner product so that the gradient in the training process is more stable. The distance measure $\Delta_{t\tau}$ represents the distance between time steps t and τ, and the positional encoding determines it. Mask() indicates the mask operation to mask the data of subsequent events and prevent information leakage.

Attention is calculated n times, and the results are combined, known as a multi-head self-attention process. This enables the model to focus on information from various representation subspaces, enhancing generalization and extending the ability to focus on different locations.

$$head_i = \text{Attention}_i, \tag{6}$$

$$\text{MHA}(Q,K,V) = \text{cat}(head_1, head_2, \ldots, head_n)W^{out}. \tag{7}$$

3.3 Question Encoder and Knowledge Encoder

The question encoder encoder will use original question $\{x_1,\ldots,x_t\}$ as input and uses a monotone attention mechanism to output context-aware questions embedded $\{\hat{x}_1,\ldots,\hat{x}_t\}$.

$$\hat{x}_t = f_{enc1}(x_1,\ldots,x_t), \tag{8}$$

where $\hat{x}_{t-\tau}$ represents the context-aware embedding that takes into account previous questions.

Knowledge encoders embed the original question-response $\{y_1,\ldots,y_{t-1}\}$ as input, using the monotonic attention mechanism to output the actual knowledge obtained $\{\hat{y}_1,\ldots,\hat{y}_{t-1}\}$.

$$\hat{y}_{t-1} = f_{enc2}(y_1,\ldots,y_{t-1}), \tag{9}$$

where $\hat{y}_{t-\tau}$ represents the context-aware embedding of past student responses.

Context-aware embedding indicates that learners' comprehension and learning while answering questions depend on the individual learner. When two learners have distinct historical records, their understanding of the same question and the knowledge acquired through practice may differ.

3.4 Knowledge Retriever

The knowledge retriever utilizes a monotonic attention approach to fetch previously acquired relevant knowledge, which pertains to the current question. It inputs the above two embeddings and generates the knowledge state that has been retrieved.

$$h_t^{kr} = f_{kr}(\hat{x}_1, \ldots, \hat{x}_t, \ldots, \hat{y}_1, \ldots, \hat{y}_{t-1}), \tag{10}$$

where h_t^{kr} represents the student's knowledge state at time t, which is also the context-aware knowledge state.

The BEKT model aims to predict the likelihood of a student providing accurate responses at time t, grounded in their knowledge state. The BEKT model postulates that the student's behavior can lead to fluctuations in their knowledge state. Therefore, several improvements are made to the context-aware knowledge state to capture these changes. The details of the improvements made to the initial knowledge state are described in detail in the Knowledge State Layer, as discussed in Section 3.5.

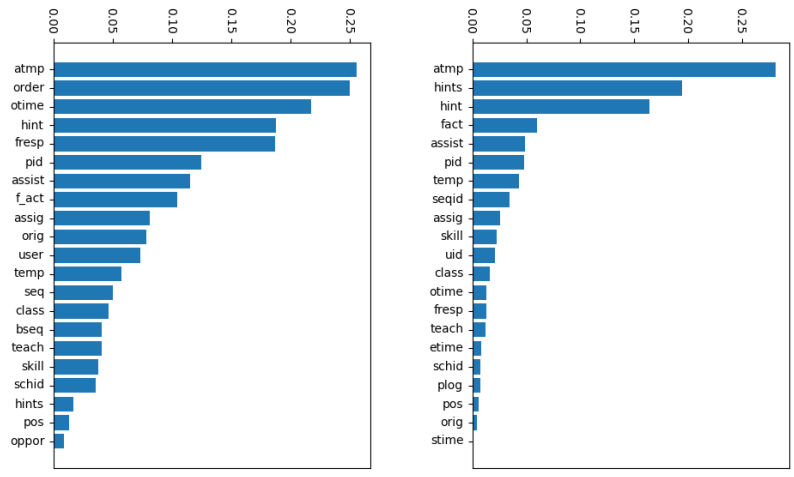
3.5 Knowledge State Layer

At the knowledge state layer, BEKT first analyzes the features of the datasets using mutual information to get the student behavior features that are most relevant to the response to the question. A cross-feature method is proposed to splice student behavior features and apply the cross-feature to the knowledge state. Then, a method of blending features with a multi-layer perceptron is proposed to combine the knowledge states affected by students' behavior with the fused features to enhance the representation of knowledge states.

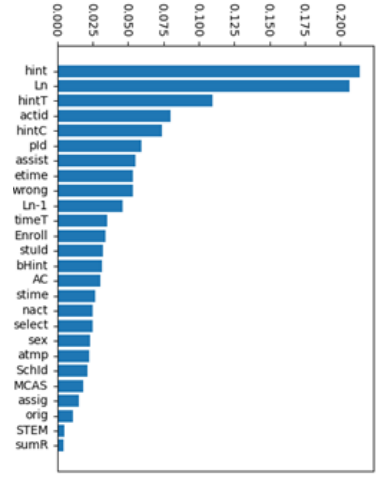
3.5.1 Mutual Information

The mutual information method [13] is a conventional approach for feature selection. It evaluates the importance of features by measuring the mutual information between features and target variables. Then it selects the feature possessing the greatest mutual information value as the vital feature. Because the BEKT model aims to forecast the likelihood of providing a correct answer to the following question, the feature of the answer to the question is taken as the target variable, and mutual information is computed with other features in the datasets in turn. To visually depict the magnitude of mutual information values and identify the most correlated features, the features with the highest correlation are arranged in descending order based on their mutual information values and represented in a bar chart. The equation utilized to compute mutual information is

$$I(x;y) = \sum \sum p(x,y) \log \frac{p(x,y)}{p(x)p(y)}, \tag{11}$$



a) assistment2009 b) assistment2012



c) assistment2017

Figure 3. On the vertical axis lie the features present in the datasets, while the horizontal axis denotes the correlation between each feature and the correct one. The bar chart is sorted by correlation from most excellent to least minor.

where x is correct, and y is the feature in the datasets. Figure 3 shows the feature importance ordering in datasets. It can be observed from Figure 3 that the correlation of student behavioral features is higher than that of exercise features. Therefore, the three student behavioral features, request prompt count, attempt count, and response time, are chosen as a set of selected features.

3.5.2 Cross Feature

Crossover is an approach used to combine two or more features into a single one, representing the concurrent performance of these features [14]. The BEKT model proposed the cross features v_t combined with students' behaviors as follows:

$$v_t = \frac{ti_t}{ave\_ti_t + hint_t + at_t + \varepsilon}, \tag{12}$$

where ave_ti_t indicates the average time to do the task, $hint_t$ indicates the count of prompt requests, at_t represents the count of attempts, and $\varepsilon = 0.001$ is meant to prevent a denominator of 0.

The tanh function is used to limit the fluctuation range of the knowledge state generated by student behaviors to $[-1, 1]$. Set the threshold α to the sum of the mean attempts and the mean request prompts.

$$h_t^c = \begin{cases} \tanh(v_t), & 0 \le ave\_hint_t + ave\_at_t < \alpha, \\ -\tanh(v_t), & ave\_hint_t + ave\_at_t \ge \alpha, \end{cases} \tag{13}$$

$$\tanh(y) = \frac{e^y - e^{-y}}{e^y + e^{-y}}. \tag{14}$$

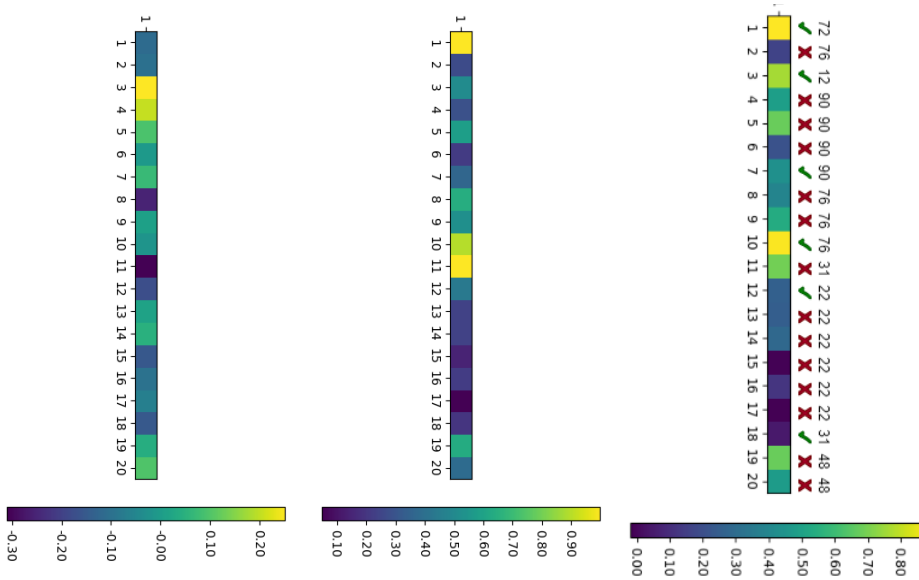
The knowledge state layer holds that students' knowledge state is affected by students' behavior. The knowledge state fluctuates under the influence of students' behavior when doing questions to get the updated knowledge state h_t.

$$h_t = f_{plus}(h_t^{kr}, h_t^c). \tag{15}$$

The behavioral features of students show good interpretability. When the number of requests and attempts exceeds a certain threshold, students are considered not to have mastered the knowledge point. In such cases, their knowledge state will decline. On the contrary, if students can gain from doing questions, their knowledge state will rise. When the time for students to do a question exceeds the average time, it is considered that students cannot solve the question, and the knowledge state decreases, whereas the knowledge state increases. Figure 4 illustrates the evolution of the knowledge state.

Taking questions 12–14 as an example from Figure 4 demonstrates how student behavior features influence the knowledge state. Given that questions 12–14 cover the same concept, where question 12 is answered correctly, and questions 13

and 14 are answered incorrectly. Figure 4 a) shows that the color becomes lighter from question 12 to question 14, which contradicts the actual answering situation. However, Figure 4 b) reveals that the color gradually becomes darker from question 12 to question 14. This implies that the knowledge state change caused by student behavior indicates that the student has not mastered the specific concept. Therefore, considering both the context-aware knowledge state and the knowledge state change caused by student behavior, as shown in Figure 4 c), it is evident that question 12 is answered correctly while questions 13 and 14 are answered incorrectly.



a) The knowledge retrieval system outputs the knowledge state h_t^{kr}. It is also a context-aware knowledge state.

b) The knowledge state fluctuations resulting from student behaviors, captured through cross-feature analysis and denoted as h_t^c.

c) The outcome of the knowledge state layer, symbolizing the student's ultimate degree of knowledge mastery, denoted as h_t.

Figure 4. The fluctuation in a student's knowledge state tracked by BEKT. The top section indicates a student's performance across a sequence of 20 questions and the corresponding concepts for each question. In the heatmap, lighter colors indicate a higher level of mastery of the knowledge by the students.

3.5.3 Multi-Layer Perceptron

The knowledge state encompasses the description of a student's level of knowledge, while student behavior features capture the description of a student's actions. To leverage different levels and types of information, this section proposes a method to integrate the knowledge state and student behavior features.

A Multi-Layer Perceptron (MLP) constitutes a feedforward neural network, comprising input, hidden, and output layers. Utilizing a MLP for feature fusion is significant as it enables the nonlinear transformation and combination of multiple input features, capturing more comprehensive and expressive feature representations. Compared with simply splicing multiple features into a long vector, MLP can better discover the correlation and importance among features and learn higher-level feature representation.

The BEKT model fuses student behavior features through MLP:

$$l_1 = (x_1 \oplus x_2 \oplus x_3)W_1 + b_1, \tag{16}$$

$$h_1 = \mathrm{ReLU}(l_1), \tag{17}$$

$$l_2 = h_1 W_2 + b_2, \tag{18}$$

where $W_1 \in R^{d_{input} \times d_{hidden}}$ and $W_2 \in R^{d_{hidden} \times d_{output}}$ are weight matrices, $b_1 \in R^{d_{hidden}}$ and $b_2 \in R^{d_{ouput}}$ are bias vectors. Let $output = l_2$ represent the fused features, and x represents the features of the student. The feature fusion process is depicted in Figure 5.

The features fused by the multi-layer perceptron and the knowledge state of student are spliced to enhance the knowledge state's representation.

$$h_t^{mlp} = \mathrm{concat}(h_t, output). \tag{19}$$

The integration of the student's cognitive state and behavior features in the variable h_t^{mlp} aids the model in better understanding the student's learning situation, thereby enhancing the accuracy of predictions.

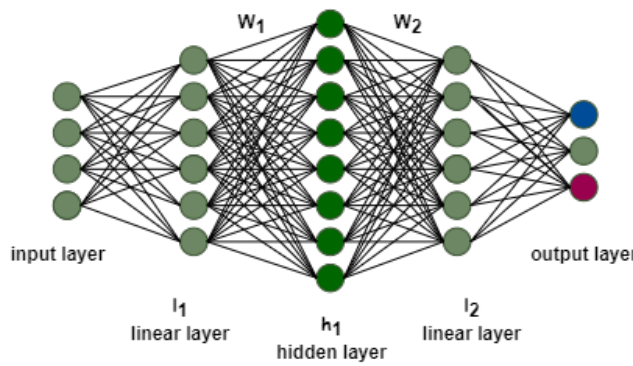


Figure 5. The process of multi-layer perceptron fusion of student behavior features. The output layer represents three features in three colors.

3.6 Prediction Layer

The response prediction layer takes as input a vector that includes the output of the knowledge state layer, the embedding of the current question x_t, and the features combined through the MLP. This input passes through a densely connected network, generating the predicted probability $\hat{r_t} \in [0, 1]$ of learners' correct answer to the current question through the sigmoid function.

$$h_t^p = \text{concat}(h_t^{mlp}, x_t) = \text{concat}(h_t, output, x_t), \tag{20}$$

$$\hat{r_t} = \sigma(h_t^p), \tag{21}$$

$$\text{Sigmoid}(x) = \frac{1}{1 + e^{-x}}. \tag{22}$$

The BEKT model is trained using the cross-entropy loss, which computes the difference between the actual response r_t and the forecasted value $\hat{r_t}$. The objective is to minimize this loss, and the optimization is performed using the Adam optimizer.

$$l = -\sum_t \left(r_t \log \hat{r_t} + (1 - r_t) \log(1 - \hat{r_t})\right). \tag{23}$$

4 EXPERIMENTS

In this segment, a comprehensive set of experiments is presented on three real-world datasets to evaluate the performance of the BEKT model thoroughly. The initial stage of this study involves the introduction of three real-world datasets. Subsequently, a comparative experiment is carried out to showcase the accuracy (ACC) of the BEKT model. Furthermore, ablation experiments are performed to provide empirical evidence of the effectiveness of the proposed improvements in the BEKT model.

4.1 Datasets and Baseline Models

4.1.1 Datasets

This paper employs three real-world datasets provided by the online tutoring platform ASSISTments to verify the accuracy of the BEKT model. Table 2 summarizes general information about each dataset. The detailed information is as follows:

1. ASSISTments2009[1]: This dataset consists of 123 concepts and 17 751 questions. The dataset is called the skill builder dataset, also known as the Mastery Learning dataset. In this dataset, a skill is considered to be mastered by students when they satisfy a specific condition (often set as continuously answering three

[1] `https://sites.google.com/site/assistmentsdata/home/assistment-2009-2010-data/skill-builder-data-2009-2010`

questions correctly), and once mastered, no further questions are recommended to the students for that particular skill [15].

2. ASSISTments2012[2]: This dataset consists of 216 concepts and 54 057 questions. The data is collected from skill-builder problem sets, in which students engage in similar exercises to attain mastery. This dataset includes affect predictions alongside the student performance data [16]. The ASSISTments 2012 dataset is characterized by its large size among publicly available datasets.
3. ASSISTments2017[3]: This dataset consists of 102 concepts and 3 162 questions. This dataset was obtained from a longitudinal study that monitored students' usage of the ASSISTments blended learning platform throughout their middle school years from 2004 to 2007. Compared to ASSISTments2012, this dataset encompasses a significantly longer student learning sequence. This dataset is employed in the 2017 ASSISTments data mining competition [16].

Dataset	ASSISTments2009	ASSISTments2012	ASSISTments2017
skills	123	254	102
problems	17 751	34 714	3 162
students	4 163	25 331	1 709
records	401 757	1 048 574	942 817

Table 2. The key characteristics of the datasets

4.1.2 Baseline Models

This article proposes a novel knowledge tracing model called BEKT, which leverages student behavior to modify their level of knowledge mastery. The proposed model considers the student behavior characteristics collected by Intelligent Tutoring Systems (ITS) and the changes in student knowledge state, often overlooked by most existing models.

Firstly, students exhibit different knowledge states when attempting different questions. Second, different student behaviors result in different variations in the knowledge state. Therefore, the primary objective of the BEKT model is to capture variations in knowledge states, thereby improving the predictive accuracy of the model. Finally, the obtained knowledge state is further enhanced by incorporating student behavior features.

To further confirm the effectiveness of the proposed approach, this paper contrasts the model against the subsequent models.

1. Deep Knowledge Tracing (DKT) [3] investigates the effectiveness of employing Recurrent Neural Networks (RNNs) for modeling student learning. By leverag-

[2] 2012-13-school-data-with-affect: `https://sites.google.com/site/assistmentsdata/2012-13-school-data-with-affect`

[3] `https://sites.google.com/view/assistmentsdatamining`

ing neural networks, significant enhancements in prediction performance have been observed across various knowledge tracing datasets.

2. Dynamic Key-Value Memory Networks (DKVMN) [10] leverage the connections among fundamental concepts to directly track a student's level of mastery for each individual concept. DKVMN comprises a fixed matrix referred to as the "key", used to retain knowledge concepts, and a flexible matrix named the "value", responsible for preserving and revising the proficiency levels of the associated concepts.
3. Convolutional Knowledge Tracing (CKT) [17] incorporates sliding windows within the CNN architecture to capture students' individualized learning rates throughout the learning process. For personalized prior knowledge, CKT measures it by relying on the learning interactions present in students' work records.
4. Attentive Knowledge tracing (AKT) [11] incorporates a unique monotonic attention approach, which connects a student's past responses to answer questions with their future responses. Additionally, the Rasch model is utilized to regulate the question and response embeddings, capturing individual differences between questions related to the same concept without extensive parameter usage.
5. The Sequential Self-Attentive model for Knowledge Tracing (SSAKT) [18] employs Multi-dimensional Item Response Theory (MIRT) to harness question information, allowing it to comprehend the associations between questions and concepts. Additionally, SSAKT incorporates a self-attention layer to capture the interdependencies among questions. In contrast to conventional self-attention networks, SSAKT's self-attention component employs Long Short-Term Memory networks (LSTM) for positional encoding. Furthermore, SSAKT incorporates a context module to capture contextual information, enhancing its overall performance.
6. Learning Process-consistent Knowledge Tracing (LPKT) [19] is a methodology that directly captures students' learning process to track their level of knowledge. It formalizes the learning process using a tuple of question, response time, and answer. LPKT extensively measures the learning gain and its diversity by considering factors such as the difference between current and prior learning cells, the time interval between them, and the students' related knowledge state.
7. Difficulty Matching Knowledge Tracing (DIMKT) [20]. The DIMKT model introduces the concept of difficulty level into the representation of questions. To establish a connection between the student's knowledge state and the level of difficulty in the questions encountered during practice.
8. Knowledge tracing based on a heterogeneous information network (HINKT) [21] obtains implicit relationships among questions or concepts through an examination of the interactions between students and the entities of questions and concepts. It marks the first knowledge tracing model to incorporate three entities simultaneously in constructing a heterogeneous information network.

4.2 Hyperparameter Experiment of BEKT

The BEKT model incorporates several hyperparameters that are crucial in determining the model's prediction performance. These hyperparameters include the number of layers in the model, the batch size used during training, the dropout rate, and the learning rate. This section conducts a sequence of experiments to examine how these hyperparameters impact the predictive performance of the model.

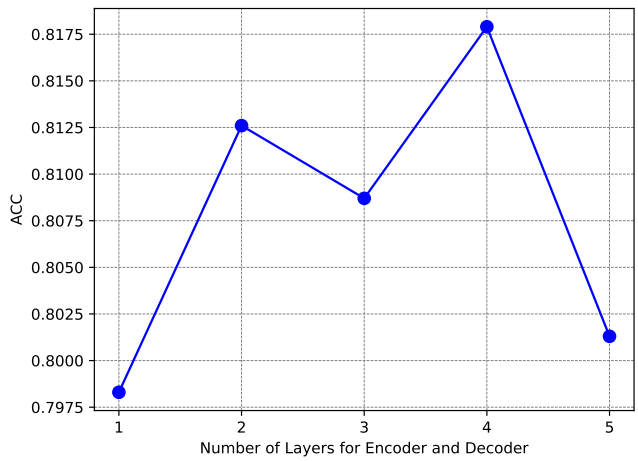


Figure 6. The ACC exhibits variations across distinct numbers of layers

4.2.1 Number of Layers for Encoder and Decoder

In this section, an experiment is conducted to determine the optimal number of layers for the encoder and decoder in the BEKT model. During the experiment on the ASSISTments2009 dataset, the number of layers for encoder and decoder $\psi = 1$, $batch\_size = 24$, dropout rate $\lambda = 0.1$, learning rate $\xi = 1\text{e}{-5}$. The number of layers ψ is adjusted within the range of 1 to 5, and the resulting changes in prediction accuracy are shown in Figure 6. While the model attains its highest accuracy with a layer number ψ of 4, it requires longer training time and more computational resources. When the number of layers ψ is set to 2, the training speed is faster, and performance improvement remains. Considering the training speed, and hardware requirements, setting the number of layers ψ to 2 is considered more appropriate for the model.

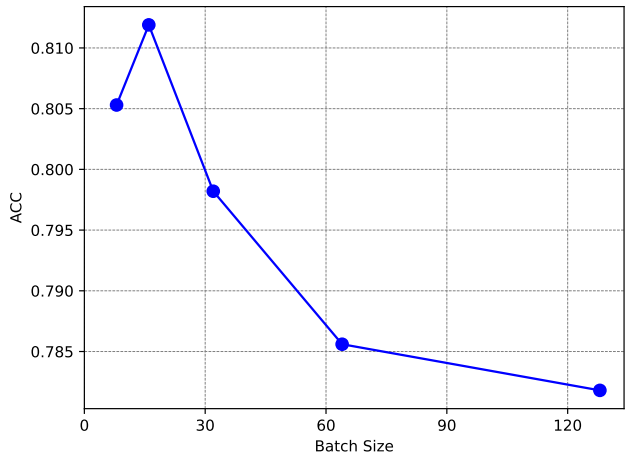


Figure 7. The ACC exhibits variations across distinct *batch_size*

4.2.2 Batch Size

In this part, the experimental procedure for determining the appropriate *batch_size* hyperparameter is presented in detail. The experiments are performed using the ASSISTments2009 dataset, with the number of layers $\psi = 2$, dropout rate $\lambda = 0.1$, and learning rate $\xi = 1e{-}5$. The *batch_size* is set to values of 2^n, where n ranges from 4 to 8 with an interval of 1. Figure illustrates the correlation between the prediction accuracy and the *batch_size*. The results indicate that when n is within the range of 4 to 8, increasing the *batch_size* results in a gradual decline in prediction accuracy. However, in Section 4.2.1, when the *batch_size* value was randomly initialized to 24, the model achieved the highest ACC. Therefore, the *batch_size* is ultimately set to 24 for this model.

4.2.3 Dropout Rate

In this section, the experiments carried out to determine the appropriate dropout rate. The experiments are conducted on the ASSISTments2009 dataset, with the number of layers $\psi = 2$, *batch_size* $= 24$, and learning rate $\xi = 1e{-}5$. The dropout rate varied within the range $[0.1, 0.5]$. Figure 8 illustrates the variation of ACC with different dropout rates. The graph indicates a decline in ACC as the dropout rate increases within the interval of $[0.1, 0.5]$. Therefore, a dropout rate of 0.1 is set in the model.

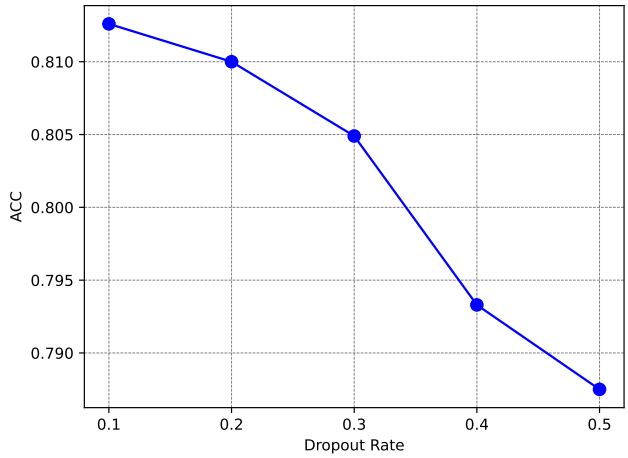


Figure 8. The ACC exhibits variations across distinct dropout rate values

4.2.4 Learning Rate

This section provides a detailed experimental procedure for determining the initial learning rate. The experiments are carried out using the ASSISTments2009 dataset, with the number of layers $\psi = 2$, $batch\_size = 24$, and dropout rate $\lambda = 0.1$. The

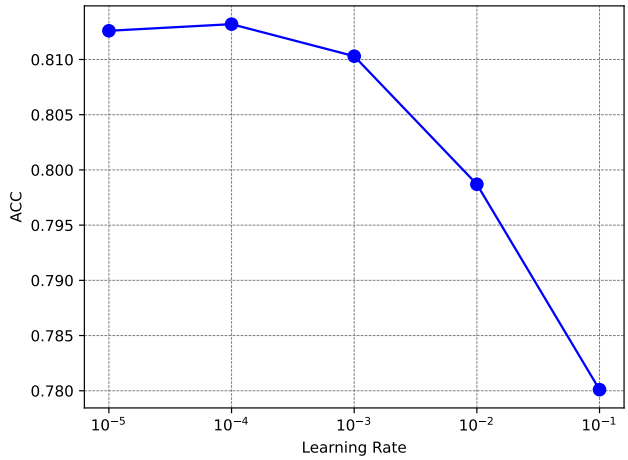


Figure 9. The ACC exhibits variations across distinct learning rate values

Figure 10. The loss when the learning rate $\xi = 0.0001$

initial learning rate ξ is varied within the given range $[1e{-}5, 0.1]$, with one order of magnitude 0.1 increments. In the model, the learning rate is fixed during training. Figure 9 illustrates the variation of ACC under different learning rates.

From Figure 9, it is evident that the ACC shows an increasing trend within the range $[1e{-}5, 1e{-}4]$ of the learning rate and a decreasing trend within the range $[0.0001, 0.1]$. Although the learning rate ξ of $1e{-}4$ achieved the highest prediction accuracy, it led to early overfitting during training. To prevent overfitting, an early stopping mechanism was employed in the experiment. Specifically, when the difference between the current training epoch and the epoch at which the best result was achieved exceeds 10, the early stopping mechanism is triggered, leading to the termination of the training process. As depicted in Figure 10, it becomes evident that when utilizing a learning rate ξ of 0.0001, the model halted training after a mere 49 epochs. Based on these observations, the model's initial learning rate is established at $1e{-}5$.

4.3 Experimental Comparison of Different Methods

This section compares different methods for capturing knowledge states to obtain more accurate ones.

4.3.1 Comparison of Attention Mechanisms

Inspired by the temporal aspect of student behavior features [22, 23], transforming monotonic attention mechanism to temporal interval attention. An influence factor based on the time interval is incorporated into calculating attention weight. This factor gradually decreases as the time interval increases, resulting in less attention

given to distant questions, thus simulating the phenomenon of forgetting among students [24].

$$w_{i,j} = \text{softmax}\left(\text{mask}\left(e^{-\theta\cdot\Delta_{ij}}\frac{Q_i^T K_j}{\sqrt{d}}\right)\right). \tag{24}$$

The variable Δ_{ij} represents the time interval between question i and question j, rather than a distance based on positional encoding. The students' interaction records are sorted according to the end time, and the interval time is defined as the duration between consecutive end times. Figure 11 shows the temporal information.

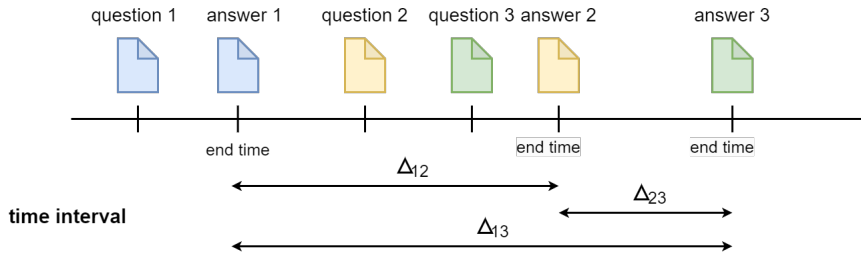


Figure 11. The time interval information. The time axis represents the order of response records, sorted by their end times. The response records for each question are indicated with the same color.

The model that replaces monotonic attention with time interval attention is referred to as BEKT-SMT (Substituting Monotonic attention with Time interval attention). The performance of BEKT-SMT was evaluated on the Assistments2012 and Assistments2017 datasets, as the Assistments2009 dataset lacked the feature of question completion time. The experimental results of BEKT-SMT are showcased within Table 3. To assess the model's performance and facilitate comparison with existing models, the area under the curve (AUC) is utilized as a standard metric in the field of knowledge tracing. According to the experimental findings, it can be observed that replacing the monotonic attention mechanism with time-based attention slightly enhances the BEKT model's performance.

BEKT-SMT	ASSISTments2012	ASSISTments2017
AUC	0.7621	0.7704
ACC	0.7933	0.7160

Table 3. Results of the BEKT-SMT model

The analysis of experimental results reveals the possible reasons for the limited improvement in model performance. The monotonic attention mechanism determines attention weights by computing the distances between time steps. This mechanism is typically suitable for sequential data where the intervals between adjacent time steps are equal or have a clear temporal order. It allows for a decreasing

weighted consideration of previous information over time, implicitly simulating the forgetting process. The time-based interval attention places greater emphasis on distinctions between time intervals. It recognizes that different time intervals may have varying importance and relevance. It can more accurately capture crucial information in time series data by considering factors such as the length, relative position, and distribution of time intervals. And the time interval-based attention mechanism better simulates forgetting, its significance is similar to monotonic attention. Therefore, the experimental results did not show significant improvements. Additionally, the time interval-based attention does not consider the inter-question correlation, where the impact between questions covering the same concept is more significant than the impact between unrelated questions. Considering that the article focuses not on forgetting, the BEKT model still utilizes the monotonic attention mechanism.

BEKT-NT	ASSISTments2009	ASSISTments2012	ASSISTments2017
AUC	0.8311	0.7527	0.7662
ACC	0.8054	0.7865	0.7144

Table 4. Results of the BEKT-NT model

4.3.2 Comparison of Cross-Feature Methods

According to the BEKT model, student behavior is believed to induce fluctuations in the knowledge state. To address this, a method called cross-features has been proposed to alter the knowledge state. The experiment aimed to explore the necessity of incorporating the tanh function in restricting the range of knowledge state fluctuations within the cross-features method. The approach that does not utilize the tanh function is referred to as BEKT-NT. The effectiveness of the tanh function was validated on three real-world datasets separately. The outcomes of the experiments are showcased in Table 4.

Through an analysis of the experimental outcomes, it was found that applying the tanh function to constrain the fluctuation range of the knowledge state is necessary for the influence caused by student behavior to become more significant, avoiding neglect the original knowledge state.

4.4 Main Results

This section offers a detailed account of the experimental procedure and presents an exhaustive analysis of the obtained results. The experimental evaluations were performed on three distinct datasets, namely assistments2009, assistments2012, and assistments2017. All samples were randomly allocated into separate training and test sets during the experimental process. To guarantee the resilience of the experimental outcomes, a five-fold cross-validation approach was utilized. Table 5 presents each model's AUC, ACC, and Loss results on the three datasets. Furthermore, the

benchmark scores were replicated in the experiment, ensuring consistency with the established reference scores.

Model	ASSISTments2009		ASSISTments2012		ASSISTments2017	
	AUC	ACC	AUC	ACC	AUC	ACC
DKT	0.7382	0.7016	0.6130	0.6464	0.7087	0.6778
DKVMN	0.7406	0.7359	0.6312	0.6531	0.7166	0.6864
CKT	0.7411	0.7382	0.6357	0.6578	0.7263	0.6921
AKT	0.7563	0.7413	0.6462	0.7034	0.7342	0.7059
SSAKT	0.7591	0.7466	0.6340	0.6756	0.7407	0.7067
LPKT	–	–	0.7387	0.7109	0.7469	0.7105
DIMKT	0.7880	0.7652	0.7461	0.7146	0.7503	0.7148
HINKT	0.8193	0.7802	–	–	0.7512	**0.7200**
BEKT	**0.8397**	**0.8126**	**0.7610**	**0.7942**	**0.7696**	0.7157

Table 5. Performance prediction outcomes of the comparative models

Drawing insights from the information showcased in Table 5, our proposed model showcases superior performance when contrasted with the comparison algorithms in knowledge tracing prediction across the three datasets. Each model's performance is slightly higher on the assistments2009 dataset compared to the other two datasets. This result may be attributed to the relatively smaller sample size in the assistments2009 dataset than the others. Compared to the AKT model, BEKT exhibits an average improvement of 7.78 % in AUC and 5.73 % in ACC across the three datasets. This result demonstrates the effectiveness of incorporating students' behavior features in knowledge tracing, highlighting the positive impact of their integration.

The experimental findings outlined above can be explained by the incorporation of knowledge states influenced by features of student behavior in the BEKT model. When seeking helpful information in the dataset, the BEKT model focuses on the question information and considers the student information. Through analysis, it is found that student behavior features have the most pronounced influence on the prediction target. The model first utilizes student behavior features to modify the student's knowledge state. Then, it enhances the representation of knowledge states using student behavior features. The precision of predictions rises in tandem with the accuracy of students' knowledge states.

4.5 Ablation Study

This section carries out an ablation study to assess the impact of each component in the BEKT model. We evaluate two variants of the BEKT model, with each variant excluding one element from the original configuration. The particulars of these variants are outlined below:

- BEKT-RM (Remove MLP) solely relies on cross-features without incorporating MLP. In BEKT-RM, the MLP-fused features are not utilized to strengthen the

representation of the knowledge state. Instead, the focus is on utilizing cross-features, which introduce fluctuations in the knowledge state.

- BEKT-RC (Remove Cross-features) exclusively utilizes MLP while excluding cross-features. The experimental outcomes of distinct models on the ASSISTments 2009, ASSISTments 2012, and ASSISTments 2017 datasets are summarized in Table 6.

According to Table 6, it is evident that the proposed components of the model contribute effectively to improving prediction accuracy. First, through a comparison of the experimental outcomes between the BEKT-RM model and the BEKT model, it is evident that considering the influence of student behavior on knowledge state leads to a significant improvement in KT prediction accuracy. The BEKT-RM model achieves an average decrease of 2.25 % in AUC and 1.13 % in ACC across the three real-world datasets. Second, through a comparison of the experimental outcomes between the BEKT-RC model and the BEKT model, it can be observed that enhancing the representation of the knowledge state through the fusion of features using MLP also enhances the model's performance. The BEKT-RC model achieves an average decrease of 1.59 % in AUC and 0.26 % in ACC across the three real-world datasets.

Model	ASSISTments2009		ASSISTments2012		ASSISTments2017	
	AUC	ACC	AUC	ACC	AUC	ACC
BEKT-RM	0.8016	0.7996	0.7468	0.7830	0.7545	0.7061
BEKT-RC	0.8154	0.8106	0.7537	0.7917	0.7535	0.7124
BEKT	0.8397	0.8126	0.7610	0.7942	0.7696	0.7157

Table 6. Ablation studies

5 CONCLUSION AND DIRECTIONS FOR FUTURE WORK

In this study, a behavior-enhanced knowledge tracing model called BEKT is proposed. The model analyzes the impact of multi-dimensional features in the dataset on knowledge states. The model first employs mutual information analysis to identify the close correlation between student behavior features and the prediction target. Specifically, the request prompts count, attempt count, and response time are identified as significant student behavior features. The influence of these three features on prediction accuracy is further investigated, revealing their ability to change student knowledge states and interpretability. A cross-feature approach is proposed to capture the changes in student knowledge states. Finally, the student behavior features are fused through an MLP, combining the fused features with the knowledge states to enhance their representation. The BEKT model demonstrates favorable prediction performance on three real-world datasets.

In the future, we will continue to explore how to leverage the abundant information collected by Intelligent Tutoring Systems (ITS) to improve model prediction

accuracy. Furthermore, the Rasch model merely distinguishes concepts and items based on parameters. We can employ graphical representations to explicitly differentiate the relationships between concepts and questions, thereby altering the model's embeddings. Moreover, we can also investigate the relationships between questions and questions and concepts and concepts to adjust the weights of attention mechanisms. These research directions aim to refine the model and optimize its performance by incorporating more nuanced and comprehensive information.

REFERENCES

[1] Le, C. V.—Pardos, Z. A.—Meyer, S. D.—Thorp, R.: Communication at Scale in a MOOC Using Predictive Engagement Analytics. In: Penstein Rosé, C., Martínez-Maldonado, R., Hoppe, H. U. et al. (Eds.): Artificial Intelligence in Education (AIED 2018). Springer, Cham, Lecture Notes in Computer Science, Vol. 10947, 2018, pp. 239–252, doi: 10.1007/978-3-319-93843-1_18.

[2] Corbett, A. T.—Anderson, J. R.: Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge. User Modeling and User-Adapted Interaction, Vol. 4, 1994, No. 4, pp. 253–278, doi: 10.1007/BF01099821.

[3] Piech, C.—Bassen, J.—Huang, J.—Ganguli, S.—Sahami, M.—Guibas, L. J.—Sohl-Dickstein, J.: Deep Knowledge Tracing. In: Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., Garnett, R. (Eds.): Advances in Neural Information Processing Systems 28 (NIPS 2015). Curran Associates, Inc., 2015, pp. 505–513, https://proceedings.neurips.cc/paper_files/paper/2015/file/bac9162b47c56fc8a4d2a519803d51b3-Paper.pdf.

[4] Pandey, S.—Karypis, G.: A Self-Attentive Model for Knowledge Tracing. In: Lynch, C. F., Merceron, A., Desmarais, M., Nkambou, R. (Eds.): Proceedings of the 12th International Conference on Educational Data Mining (EDM 2019). 2019, pp. 384–389, doi: 10.48550/arXiv.1907.06837.

[5] Zhang, L.—Xiong, X.—Zhao, S.—Botelho, A.—Heffernan, N. T.: Incorporating Rich Features into Deep Knowledge Tracing. Proceedings of the Fourth (2017) ACM Conference on Learning @ Scale (L@S '17), 2017, pp. 169–172, doi: 10.1145/3051457.3053976.

[6] Yang, H.—Cheung, L. P.: Implicit Heterogeneous Features Embedding in Deep Knowledge Tracing. Vol. 10, 2018, doi: 10.1007/s12559-017-9522-0.

[7] Sun, X.—Zhao, X.—Ma, Y.—Yuan, X.—He, F.—Feng, J.: Muti-Behavior Features Based Knowledge Tracking Using Decision Tree Improved DKVMN. Proceedings of the ACM Turing Celebration Conference – China (ACM TURC '19), 2019, doi: 10.1145/3321408.3322847.

[8] Nagatani, K.—Zhang, Q.—Sato, M.—Chen, Y. Y.—Chen, F.—Ohkuma, T.: Augmenting Knowledge Tracing by Considering Forgetting Behavior. The World Wide Web Conference (WWW '19), 2019, pp. 3101–3107, doi: 10.1145/3308558.3313565.

[9] Chaudhry, R.—Singh, H.—Dogga, P.—Saini, S. K.: Modeling Hint-Taking Behavior and Knowledge State of Students with Multi-Task Learning. Proceedings

of the 11th International Conference on Educational Data Mining (EDM), 2018, pp. 21–31, https://files.eric.ed.gov/fulltext/ED593119.pdf.

[10] Zhang, J.—Shi, X.—King, I.—Yeung, D. Y.: Dynamic Key-Value Memory Networks for Knowledge Tracing. Proceedings of the 26th International Conference on World Wide Web (WWW '17), 2017, pp. 765–774, doi: 10.1145/3038912.3052580.

[11] Ghosh, A.—Heffernan, N.—Lan, A. S.: Context-Aware Attentive Knowledge Tracing. Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '20), 2020, pp. 2330–2339, doi: 10.1145/3394486.3403282.

[12] Pardos, Z. A.—Heffernan, N. T.: Modeling Individualization in a Bayesian Networks Implementation of Knowledge Tracing. In: De Bra, P., Kobsa, A., Chin, D. (Eds.): User Modeling, Adaptation, and Personalization (UMAP 2010). Springer, Berlin, Heidelberg, Lecture Notes in Computer Science, Vol. 6075, 2010, pp. 255–266, doi: 10.1007/978-3-642-13470-8_24.

[13] Belghazi, M. I.—Baratin, A.—Rajeshwar, S.—Ozair, S.—Bengio, Y.—Courville, A.—Hjelm, D.: Mutual Information Neural Estimation. In: Dy, J., Krause, A. (Eds.): Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research (PMLR), Vol. 80, 2018, pp. 531–540, https://proceedings.mlr.press/v80/belghazi18a.html.

[14] Sun, X.—Zhao, X.—Li, B.—Ma, Y.—Sutcliffe, R.—Feng, J.: Dynamic Key-Value Memory Networks with Rich Features for Knowledge Tracing. IEEE Transactions on Cybernetics, Vol. 52, 2022, No. 8, pp. 8239–8245, doi: 10.1109/TCYB.2021.3051028.

[15] Feng, M.—Heffernan, N.—Koedinger, K.: Addressing the Assessment Challenge with an Online System That Tutors as It Assesses. User Modeling and User-Adapted Interaction, Vol. 19, 2009, No. 3, pp. 243–266, doi: 10.1007/s11257-009-9063-7.

[16] Patikorn, T.—Baker, R. S.—Heffernan, N. T.: ASSISTments Longitudinal Data Mining Competition Special Issue: A Preface. Journal of Educational Data Mining, Vol. 12.

[17] Shen, S.—Liu, Q.—Chen, E.—Wu, H.—Huang, Z.—Zhao, W.—Su, Y.—Ma, H.—Wang, S.: Convolutional Knowledge Tracing: Modeling Individualization in Student Learning Process. Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '20), 2020, pp. 1857–1860, doi: 10.1145/3397271.3401288.

[18] Zhang, X.—Zhang, J.—Lin, N.—Yang, X.: Sequential Self-Attentive Model for Knowledge Tracing. In: Farkaš, I., Masulli, P., Otte, S., Wermter, S. (Eds.): Artificial Neural Networks and Machine Learning – ICANN 2021. Springer, Cham, Lecture Notes in Computer Science, Vol. 12891, 2021, pp. 318–330, doi: 10.1007/978-3-030-86362-3_26.

[19] Shen, S.—Liu, Q.—Chen, E.—Huang, Z.—Huang, W.—Yin, Y.—Su, Y.—Wang, S.: Learning Process-Consistent Knowledge Tracing. Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining (KDD '21), 2021, pp. 1452–1460, doi: 10.1145/3447548.3467237.

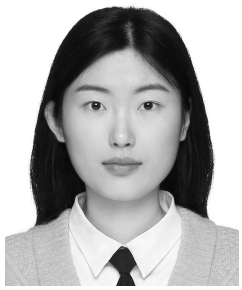
[20] Shen, S.—Huang, Z.—Liu, Q.—Su, Y.—Wang, S.—Chen, E.: Assessing Student's Dynamic Knowledge State by Exploring the Question Difficulty Effect. Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22), 2022, pp. 427–437, doi: 10.1145/3477495.3531939.

[21] Xu, J.—Huang, X.—Xiao, T.—Lv, P.: Improving Knowledge Tracing via a Heterogeneous Information Network Enhanced by Student Interactions. Expert Systems with Applications, Vol. 232, 2023, Art. No. 120853, doi: 10.1016/j.eswa.2023.120853.

[22] Wang, C.—Ma, W.—Zhang, M.—Lv, C.—Wan, F.—Lin, H.—Tang, T.—Liu, Y.—Ma, S.: Temporal Cross-Effects in Knowledge Tracing. Proceedings of the 14th ACM International Conference on Web Search and Data Mining (WSDM '21), 2021, pp. 517–525, doi: 10.1145/3437963.3441802.

[23] Sun, J.—Zou, R.—Liang, R.—Gao, L.—Liu, S.—Li, Q.—Zhang, K.—Jiang, L.: Ensemble Knowledge Tracing: Modeling Interactions in Learning Process. Expert Systems with Applications, Vol. 207, 2022, Art. No. 117680, doi: 10.1016/j.eswa.2022.117680.

[24] Liu, S.—Zou, R.—Sun, J.—Zhang, K.—Jiang, L.—Zhou, D.—Yang, J.: A Hierarchical Memory Network for Knowledge Tracing. Expert Systems with Applications, Vol. 177, 2021, Art. No. 114935, doi: 10.1016/j.eswa.2021.114935.



Si'ao Sun received her M.Sc. degree from the Shandong University of Science and Technology. She currently holds a position as a Full-Time Teacher in the School of Artificial Intelligence at Shandong Xiandai University. Her current research interests include time series analysis, deep learning and intelligent education.

Liqing Qiu is Associate Professor at the Shandong University of Science and Technology. She received her Ph.D. degree in computer software and theory from Beihang University, Beijing, P.R. China. Her current research interests include data mining, social networks and deep learning.