applied sciences

MDPI

Article

A Multi-Layer Attention Knowledge Tracking Method with Self-Supervised Noise Tolerance

Haifeng Wang, Hao Liu, Yanling Ge * and Zhihao Yu *

School of Information Science and Engineering, Linyi University, Linyi 276000, China; wanghaifeng@lyu.edu.cn (H.W.); 210854002028@lyu.edu.cn (H.L.)
* Correspondence: geyanling@lyu.edu.cn (Y.G.); yuzhihao1@lyu.edu.cn (Z.Y.)

Abstract

The knowledge tracing method based on deep learning is used to assess learners' cognitive states, laying the foundation for personalized education. However, deep learning methods are inefficient when processing long-term series data and are prone to overfitting. To improve the accuracy of cognitive state prediction, we design a Multi-layer Attention Self-supervised Knowledge Tracing Method (MASKT) using self-supervised learning and the Transformer method. In the pre-training stage, MASKT uses a random forest method to filter out positive and negative correlation feature embeddings; then, it reuses noise-processed restoration tasks to extract more learnable features and enhance the learning ability of the model. The Transformer in MASKT not only solves the problem of long-term dependencies between input and output using an attention mechanism, but also has parallel computing capabilities that can effectively improve the learning efficiency of the prediction model. Finally, a multidimensional attention mechanism is integrated into cross-attention to further optimize prediction performance. The experimental results show that, compared with various knowledge tracing models on multiple datasets, MASKT's prediction performance remains 2 percentage points higher. Compared with the multidimensional attention mechanism of graph neural networks, MASKT's time efficiency is shortened by nearly 30%. Due to the improvement in prediction accuracy and performance, this method has broad application prospects in the field of cognitive diagnosis in intelligent education.

Keywords: deep learning; knowledge tracking; noise tolerance; feature selection; masked language model; multi-layer attention

Academic Editor: Yutaka Ishibashi

Received: 3 June 2025
Revised: 12 July 2025
Accepted: 29 July 2025
Published: 6 August 2025

Citation: Wang, H.; Liu, H.; Ge, Y.; Yu, Z. A Multi-Layer Attention Knowledge Tracking Method with Self-Supervised Noise Tolerance. *Appl. Sci.* **2025**, *15*, 8717. https://doi.org/10.3390/app15158717

1. Introduction

Knowledge tracing is an important branch of education, including predicting learners' answer behavior, classifying knowledge points, and analyzing cognitive states. With the recent developments in deep learning technology, its application in knowledge tracing has also begun to spread. In traditional methods, the Bayesian knowledge tracking model (BKT) is a commonly used method for tracking student skill status [1]. Before the popularization of deep learning, it was used as a basic model for improvement and innovation and has strong interpretability. The rapid development of deep learning methods has prompted researchers to apply Recursive Neural Networks (RNNs) in the field of intelligent education [2], with performance superior to traditional models such as BKT. Masked language models perform well in handling missing information and noisy data, and use upstream multi-task training to provide a reliable foundation for downstream applications. The Multi-layer Attention Self-supervised Knowledge Tracing Method uses masked language models to help models

in understanding language structure and information features, thereby improving the predictive performance of the model.

This paper proposes a MASKT model that combines self-supervised learning with multi-layer attention networks for knowledge tracing. Specifically, since the data in the current knowledge tracing field contain different features, random forest is used to filter out highly correlated features in the initial learning interaction sequence and classify them into positive and negative correlations to enhance the compatibility between the model and the dataset. The processed sequence is then input into the self-supervised masking task encoder for secondary processing. By performing three different noise restoration tasks, namely fragmentation, discrepancy, and reordering, the sequence structure information is destroyed to reduce the risk of model overfitting. The output sequence is then input as a learning sequence into two special attention encoders: a node encoder and a dimension encoder, which are used to capture potential connections in historical learning sequences in a hierarchical manner to ensure that dependency information at different granularity levels is modeled. This enables the more comprehensive capture of key features and improves model prediction performance. During the optimization process, the MASKT model incorporates noise processing and multi-level semantic information into learning interactions and knowledge state representations. In this way, MASKT balances the significance of knowledge state and prediction performance. The main contributions of this paper are as follows:

This paper designs a multi-level encoder based on attention to learn the dynamic input of masked language models, so as to more efficiently alleviate the long-dependency problem at different levels through hierarchical decomposition of sequences. Additionally, a random forest filtering strategy is designed to rank dataset attributes based on positive or negative correlations, ensuring the order of feature importance learning. This promotes MASKT's ability to learn more representative knowledge states and interactions. Extensive experiments on five public datasets demonstrate that MASKT outperforms other baseline models. Furthermore, MASKT adjusts parameters through ablation experiments to analyze the impacts of different noise sequences on model performance.

2. Related Work

Attention mechanisms are suitable for knowledge tracing tasks due to their ability to adaptively capture sequence dependencies. Knowledge tracing requires a thorough understanding of students' learning performance and knowledge state updates by combining past and current data. This has been extended to the Self-Attentive model for Knowledge Tracing (SAKT) and Attentive Knowledge Tracing (AKT). The application of self-attention mechanisms improves the interpretability and performance of the model. However, such models are constrained by the answer sequence and must strictly follow the time sequence. SAKT [3] was the first to introduce self-attention into knowledge tracing to capture contextual information for knowledge tracing tasks. Subsequently, AKT [4] improved the use of the attention mechanism, making tens of thousands of parameters interpretable. In order to further explore the decision-making process and interpretability of deep learning knowledge tracing, Sun et al. constructed a multi-layer attention network that uses graph attention neural networks and self-attention mechanisms to mine multidimensional and deep semantic associations. Regularization terms and trade-off factors were introduced into the loss function to improve the interpretability of the model and to achieve a quantitative assessment of the interpretability of the model [5].

In addition, researchers have attempted to optimize traditional structural models by changing deep learning models. Tian Zhejie et al. used a bidirectional encoder representation model combined with auxiliary information such as students' historical learning

performance to predict learners' knowledge states, analyzed learners' learning logs in detail, and explored the impact of auxiliary methods on knowledge states [6]. Mike Lewis et al. added noise removal methods to pre-trained sequence models and used noise to destroy text information. They trained a Bidirectional and Auto-Regressive Transformers (BART) model to reconstruct text enhancement models that understand limited information features, promoting the development of Transformer models for transfer tasks [7]. However, methods based on pre-trained embeddings or enhanced samples can only rely on end-to-end architectures, which are more effective for low-dimensional vector extraction than high-dimensional vector extraction, resulting in high-dimensional feature loss. Denoising methods focus on specific types of downstream tasks, have limited applicability, and cannot guarantee the learning effectiveness of vectors of different dimensions. Although Transformers have achieved great success in the field of knowledge tracing, there is still room for improvement. Transformers have a large number of parameters and layers, which require high computing resources. The self-supervised learning methods derived from this have achieved remarkable results in NLP tasks. Improvements such as reconstructing randomly masked text subsets, predicting masked tokens [8], and replacing token contexts [9] limit the model's functionality to specific tasks. To address the pre-training of word embedding vectors, Mnih et al. established a sequential language modeling objective [10], which was further extended by Peters et al. to a bidirectional language model to extract context-related features [11].

3. MASKT Model

This paper proposes a Multi-layer Attention Self-supervised Knowledge Tracing Method (MASKT) that modifies the Transformer architecture and uses self-supervised learning to extract similar historical information [12]. MASKT mainly consists of a knowledge encoder and an answer interaction encoder, which use a masked language model to perform pre-training tasks and apply the pre-trained model parameters to the downstream Transformer model [13]. It also integrates a double-layer attention network at the semantic and interaction levels to improve knowledge tracing performance.

As shown in Figure 1, MASKT includes a pre-training token detection component and a knowledge mastery accuracy prediction component. First, by calculating the positive and negative correlation coefficients of different features, 10 attributes with high positive and negative correlations are selected as auxiliary interaction embeddings to enhance the feature learning ability of the model. In the pre-training stage, MASKT uses self-supervised learning tasks to process student interaction sequences and enhance the semantic extraction ability of the embedding layer. The encoder performs the mask recovery task, learns contextual information, and outputs a sequence containing student cognitive state information. The decoder component uses multidimensional cross-attention to fuse information and optimize prediction accuracy.

3.1. Problem Definition

In the knowledge tracing process, the historical learning sequence of learners contains a series of important features. To enable the model to better learn the required features, each interaction sequence is defined as consisting of three parts, $c_t = (q_t, a_t, kc_t)$. Here, q_t denotes the question sequence, $q_t \in Q$. a_t denotes the answer sequence, $a_t \in (0, 1)$. kc_t denotes the knowledge point set, $kc_t \in K$, and each q_t is associated with a specific kc_t. Given the learning sequence $(c_1, c_2, \ldots c_t)$, the probability of answering the next question correctly at the next time step is predicted as follows:

$$\hat{y}_{t+1}(q_{t+1}) = p(a_{t+1}|c_1, c_2, \ldots c_t) \quad (1)$$

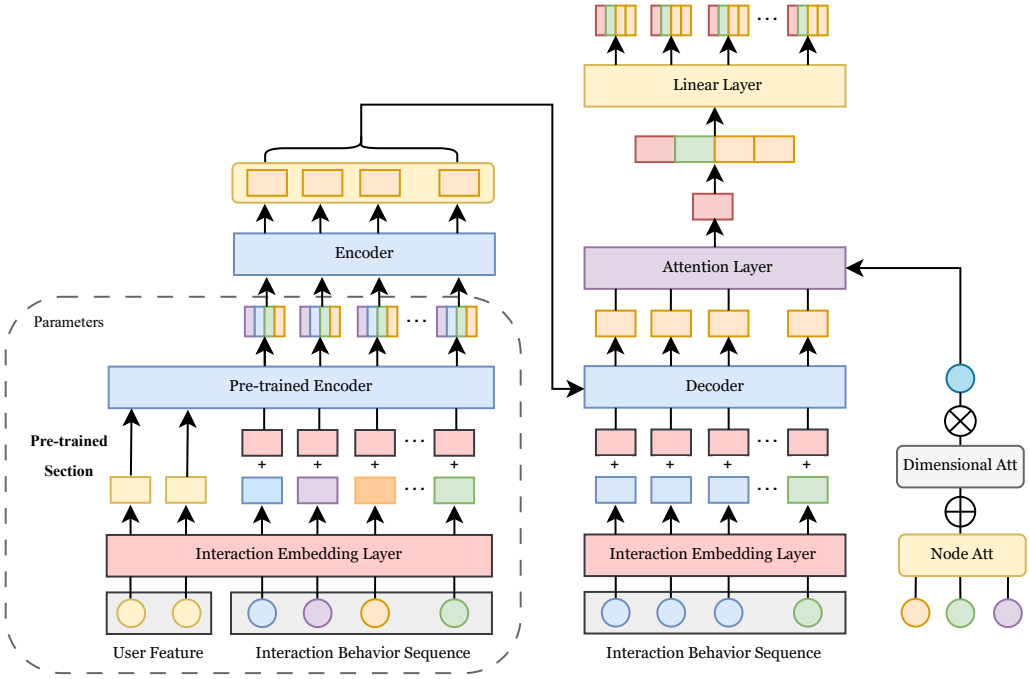

Figure 1. MASKT overall architecture.

Existing Knowledge Tracing (KT) model studies embed other information from learners' responses in different ways, but they only implicitly use question-related information to obtain more diverse information representations. In MASKT, the paper introduces a new UF module to measure the importance of learners' auxiliary information. Correlation scores are calculated to filter out more effective auxiliary information, which is then concentrated and embedded as a new input vector. This is combined with the original interaction embedding c_t to enhance the relevance of the learner representation embedding. Let u_m denote the set of auxiliary information in the dataset, where $u_m \in U$ and m represents different attribute information.

3.2. Feature Embedding Screening

As an ensemble learning method, random forest consists of multiple decision trees and can effectively deal with nonlinear relationships between features and high-dimensional and redundant features. Compared with supervised methods such as linear discriminant analysis, random forest is more stable and robust in dealing with nonlinear feature relationships, and is particularly suitable for complex and diverse feature structures such as educational behavior data [14]. Compared with the "black box" feature extraction method of deep neural networks, random forest provides a clear ranking of feature importance, which can more intuitively understand which behavioral features (such as the number of prompts, the correctness of answering questions, etc.) play a key role in predicting learning performance, which is conducive to teaching intervention and strategy formulation [15]. Therefore, with the purpose of using the least feature vectors and carrying the most student information, the importance of the screened behavioral features is ranked as the pre-feature screening module of the self-supervised learning model, providing more refined and discriminative input features for subsequent modeling, and being able to more specifically retain features that have a significant impact on the prediction results.

Since the attributes contained in the learners' answers also have different characteristics, such as differences in the relevant attributes when the answer is correct or incorrect, we divide the questions into two levels. Learner–question pair: When the answer is correct, a bipartite graph is formed by the learners' correct answers in the dataset. Therefore,

we believe that there may be a correlation between the questions that the same learner answered correctly and incorrectly. The positive feedback correlation score is denoted as UF^+, representing the learner–question pair where the answer is incorrect, while the negative feedback correlation score is denoted as UF^-. The principle involves bootstrapping random sampling of the training set during training to form n datasets, generating N decision trees from these datasets, and using out-of-bag (OOB) data to calculate errors and evaluate the accuracy of the random forest. In a random forest, assuming there are N trees, the correlation of features is evaluated using Formula (2).

$$Imp = \frac{1}{N}|errOOB_1 - errOOB_2| \tag{2}$$

Imp indicates the relevance score of the feature, i indicates the number of features, N indicates the number of decision trees in the random forest, $errOOB_1$ indicates the out-of-bag error without interference, and $errOOB_2$ indicates the out-of-bag error with noise interference. The evaluation criterion for feature importance is whether the accuracy of the OOB data significantly decreases after noise is added. If the accuracy decreases, this indicates that the feature has a significant impact on the prediction results of the model and is marked as an important feature.

The above method is transferred and applied to the evaluation of behavioral features. The behavioral features obtained are sorted by importance. Before transferring them to the MASKT model, the learner's answer behavior features extracted using random forests are weighted using PCA (Principal Component Analysis). During the answering process, learners' interactions with different questions are also different, and there is a necessary relationship between interaction relevance and answer accuracy. Based on the learners' answer records, the relevance between answer interactions and answer results is quantified as follows:

$$Correlation(x_j)\begin{cases}\frac{\sum_{i=1}^{|N_j|}|e_{ij}=0|}{|N_j|}, |N_j| \\ 0.5, else\end{cases} \tag{3}$$

N_j is the set of learners who have answered the current question. e_{ij} is the response result of learner i to exercise j, where 1 or 0 indicates correct/incorrect. If the number of students who answered exercise e_j correctly is less than 10, the difficulty of the question is deemed to exceed the current ability range of the students, and the relevance is set to 0.5. There are also differences in the interactive behavior of learners during the answering process, so it is necessary to comprehensively consider the positive or negative correlation scores between learners' interactive behavior and their response results to screen for important auxiliary information:

$$UF^+(e_{ij}) = \sum_{i}^{|S_i|}\frac{\{a_{ij}=1\}}{|S_i|}u_{mt} \tag{4}$$

$$UF^-(e_{ij}) = \sum_{i}^{|S_i|}\frac{\{a_{ij}=0\}}{|S_i|}u_{mt} \tag{5}$$

UF^+ represents the correlation coefficient of learner i's correct answer to question e_j; UF^- represents the correlation coefficient of learner i's incorrect answer to question e_j; S_i represents the number of attempts of learner i to answer e_j; a_{ij} represents learner i's correct or incorrect answer to exercise e_j in the interaction at time t, with 1 or 0 indicating correct/incorrect answers.

Here, we used the sklearn.ensemble module to calculate the importance of behavioral features and compare the correlation coefficients between auxiliary information and responses. This helps to effectively screen features before model training, reducing model

complexity and load. Take the ASSISTments09 dataset as an example, with features shown in Table 1. We used random forest to screen behavioral features, retained the top 10 behavioral features according to the positive correlation coefficient, and observed the impact of the negative correlation coefficient, as shown in Figure 2. The blue part represents the positive correlation coefficient, and the yellow part represents the negative correlation coefficient. From the figure, we can see that attributes such as original and attempt_count have a relatively large effect on positive and negative feedback, which is presumed to be closely related to the answer sequence. Additionally, attributes such as hint_total and overlap_time have negative feedback coefficients greater than positive feedback, suggesting that when learners make mistakes, the number of attempts on a question increases, and the time spent answering also increases. The specific data are shown in Table 2.

Table 1. Characteristics and meanings of ASSISTments09 dataset.

Behavioral Characteristics	Meaning
order_id	Student ID
problem_id	Problem ID
original	Whether the student's original answer process was marked as correct or incorrect by the teacher
correct	Whether the student's answer is correct (1 for correct, 0 for incorrect)
attempt_count	Number of practice attempts recorded when the student answered the question
ms_first_response	Time from the start time to the student's first action (in milliseconds)
tutor_mode	Tutor mode
answer_type	Answer type
sequence_id	Sequence ID
student_class_id	Student class ID
problem_set_type	Problem set type
base_sequence_id	Base sequence ID
skill_id	Skill ID involved in the question
skill_name	Question name
teacher_id	Teacher ID
school_id	School ID
hint_count	Number of hints provided during answering
hint_total	Total number of attempts during answering
overlap_time	Time to complete the question (in milliseconds)
answer_id	Answer ID
answer_text	Answer text

Table 2. Correlation coefficients of positive and negative features.

Feature Name	Positive Correlation	Negative Correlation
original (binary problem)	0.32	0.25
attempt_count (total number of attempts)	0.21	0.29
ms_first_response (start time)	0.13	0.13
hint_count (number of hints)	0.13	0.18
correct (whether correct)	0.09	0.11
answer_type (answer type)	0.06	0.03
tutor_mode (tutor mode)	0.04	0.04
Position (position)	0.04	0.01
hint_total (total number of hints)	0.01	0.05
overlap_time (completion time)	0.01	0.09

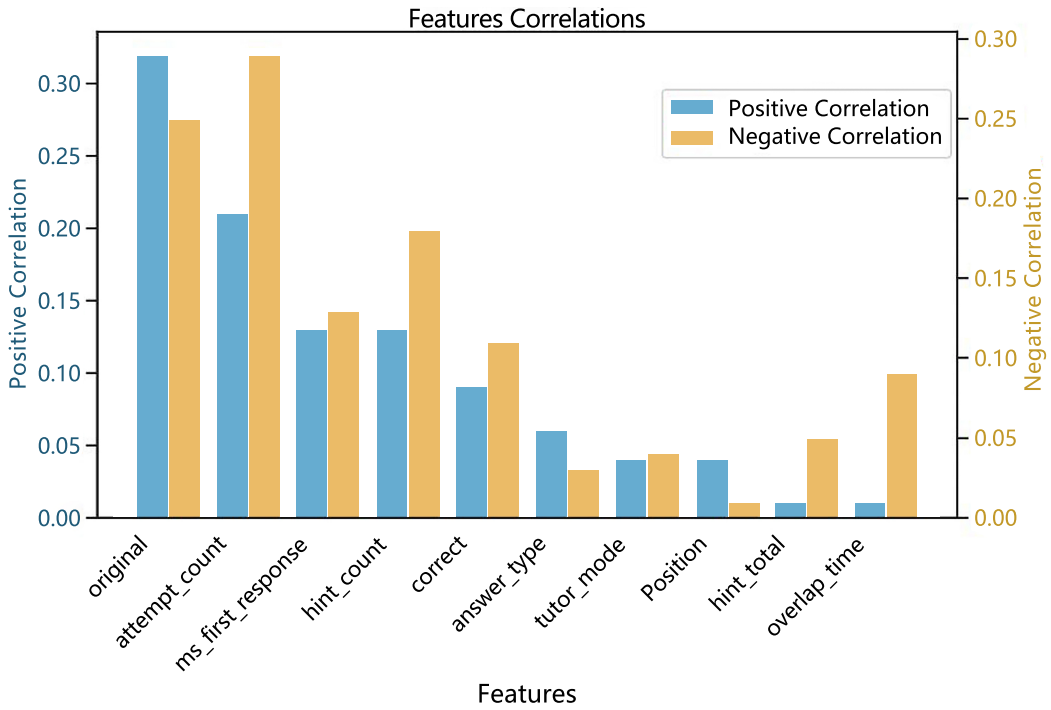

Figure 2. Positive and negative feature correlation screening.

After completing feature screening, vectors containing positive and negative correlation features are uniformly embedded into the model input sequence. Using a Multi-Layer Perceptron (MLP), strongly correlated auxiliary information-embedded sequences are output:

$$x_t = u_{mt} \otimes (\alpha \cdot (W_+[i] \cdot UF_t^+) \oplus \beta \cdot (W_-[i] \cdot UF_t^-)) \tag{6}$$

Among them, $\oplus$ and $\otimes$ are cascaded operations, $W_1 \in \mathbb{R}^{(d_c+M)\times d_k}$ is a weighting matrix used to adjust the influence of UF_t^+ and UF_t^- scores on the sequence. α and β are scaling factors used to regulate the influence of positive and negative scores.

3.3. MASKT Component

The MASKT model consists of two embedding layers and two encoders. Let the question embedding be $E^Q \in R^{q^*d}$, let the interaction feature embedding be $E^C \in R^{2q^*d_k}$, let q represent the number of questions, and let d_k represent the embedding dimension. The embedding layer generates initial interaction feature vectors. The interaction vectors x_i with strongly correlated auxiliary information are noise-processed to obtain different processing forms of branch interaction vectors, denoted as $e_i^\delta = \left(e_i^f, e_i^{d_p}, e_i^r\right)$. These represent fragmentation, discrepancy, and reordering noise interactions, respectively. The MASKT model aims to predict probabilities, and the model is represented as shown in Formula (7):

$$\hat{y}_{t+1}(q_{t+1}) = f(g^Q, g^C) \tag{7}$$

g^Q and g^C represent a question recognition function and an answer interaction recognition function, respectively. The learner combines the interaction sequence $(0, x_0, x_1 \ldots x_{n-1})$ as input and uses the interaction embedding layer to mask the input sequence to obtain the noisy interaction sequence $(0, e_0, [mask] \ldots e_{n-1})$. The noisy sequence is processed by the pre-training layer to complete the interaction sequence masking restoration task. The trained parameters contain all the features of the student's answers. The encoder outputs the hidden sequence, $(0, l_0, l_1 \ldots l_{n-1})$, the embedded interaction sequence, e_{i+1}^δ, and the timestamp, t_{i+1}. The decoder generates the hidden representation transformation block,

$e_i^{\theta\delta} = (e_i^{\theta f}, e_i^{\theta d}, e_i^{\theta r})$. The hidden features and the original data features are spliced together and enter the attention layer to update the weights. After passing through the linear layer, the correctness prediction sequence for the next time step is generated.

3.4. Self-Supervised Sequence Tasks

Inspired by the BART model noise-reduction autoencoder method, MASKT uses multiple noise sequences to assist model training [16]. Noise tasks include input sequence filling, deletion, reordering, etc. They are divided into the following three aspects, as shown in Figure 3: F noise, D noise, and R noise. The original sequence of the student is $E = (0, e_0, e_1 \ldots e_{n-1})$, $n = 8$, and the maximum length is 16.

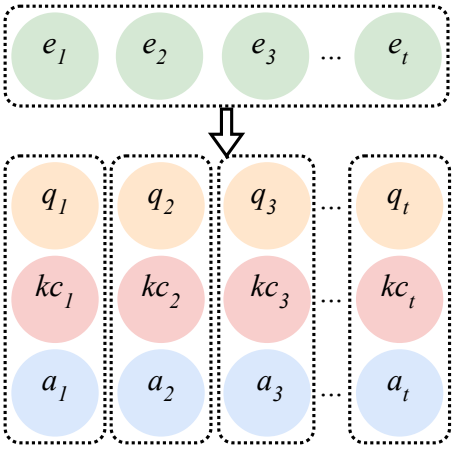

Figure 3. Original sequence structure of student interaction.

In real-world student response scenarios, learner interaction sequences typically consist of a question sequence, an answer sequence, and a set of knowledge points. Considering that post-learned behavioral factors influence answer outcomes in real-world response processes, this hinders accurate analysis of learning behavior and knowledge mastery. Therefore, this paper proposes a simulation noise to emulate error-prone response behaviors in real-world scenarios. Three main types of noise are defined and discussed:

F noise (Fragmentation Noise: This type of noise refers to a situation in which the question sequence, q_t, and the knowledge point set, kc_t, are complete in the interaction sequence tuple, but part of the answer sequence, a_t, is missing. This noise is used to simulate situations where the recording of student answers is incomplete due to reasons such as students leaving the answer midway, network connection problems, or incomplete data recording, as shown in Figure 4. The answer sequences are then uniformly adjusted in length using two methods. The first method sets a sequence length threshold, padding short sequences with zeros and splitting long sequences. The second method uses mask-marked insertion. Filling positions are marked specifically, and invalid bits in the sequences are filtered out and recorded as $e_i^{\delta f} = (e_0^{\delta f}, e_1^{\delta f} \ldots e_n^{\delta f})$.

D noise (discrepancy noise): This type of noise simulates situations in interactive sequence tuples where the problem sequences q_i and kc_i are the same at different time steps i and j, but the corresponding answer sequences, a_i, are inconsistent, as shown in Figure 5. This simulates situations where students' understanding and responses to the same question change over time or where they make mistakes in their answers. This noise makes it difficult to analyze the consistency of students' knowledge mastery. For this noise, it is possible to use the [mask] to randomly select one sequence from the two unequal sequences for marking. This prevents the model from over-relying on certain features and reduces the risk of overfitting. This is denoted as $e_i^{\delta d} = (e_0^{\delta d}, e_1^{\delta d} \ldots e_n^{\delta d})$.

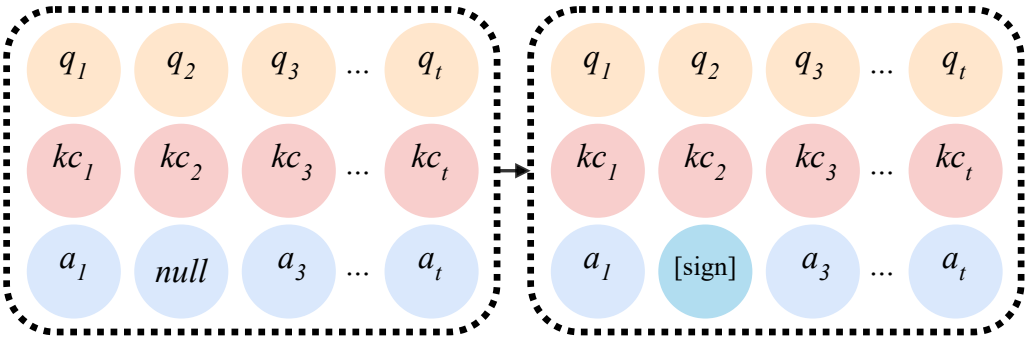

Figure 4. F noise sequence structure.

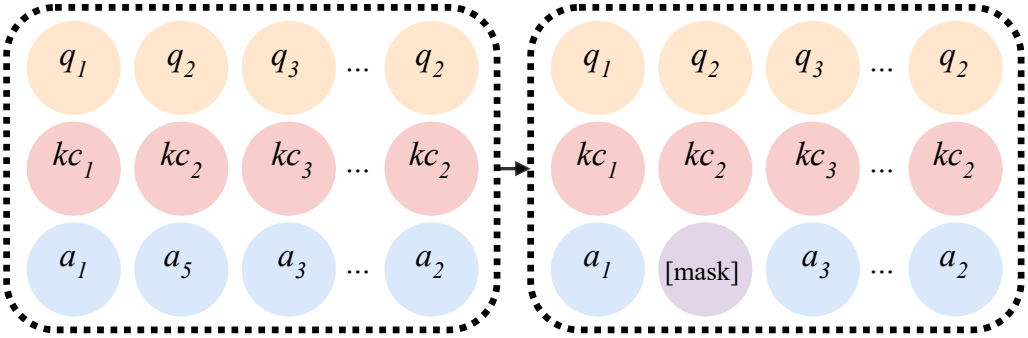

Figure 5. D noise sequence structure.

R noise (reordering noise): In learner interaction sequences, some learner response sequences are misaligned with the question sequences, causing confusion in the overall interaction sequence. This noise simulates synchronization issues during data collection or sorting errors during data processing. R noise severely affects the time-series analysis of student responses, thereby affecting the accuracy of modeling student learning paths and knowledge mastery. As shown in Figure 6, according to the minimum length of sequence q_t, assuming that the minimum length is $n = 4$, we mark the beginning and end of each question sequence q_t, and re-check whether all grouped question sequences q_t and answer sequences a_t correspond to each other. We set the hyperparameter ω such that when the matching of q_t and a_t is less than the ω threshold. Judgment sequence endings are misplaced in such a way that a_t is less than q_t, recorded as if it were a defective mismatch. Then, the starting marker position is deleted, and a subsequent a_t left shift is added. A length of a_t, less than q_t, is recognized as an incremental mismatch, which is subsequently supplemented by an a_t right shift. This approach ensures that most of the sequences are correct, and reconstructing the sequences improves the efficiency of relative position recognition of contextual information and enhances the model characteristic learning ability, denoted as $e_i^{\delta r} = (e_0^{\delta r}, e_1^{\delta r} \ldots e_n^{\delta r})$.

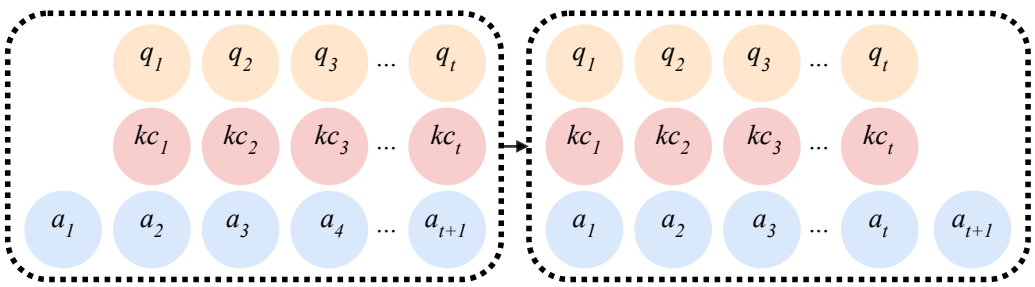

Figure 6. R noise sequence structure.

3.5. Introducing of Dynamic Masking Mechanism

The input sequence completes the mask conversion in the pre-training stage of MASKT to generate a static mask. However, a sequence corresponds to only one mask form, which reduces the data reuse efficiency and wastes a large amount of feature information in the data. The introduction of dynamic masking technology has changed this situation. In each training round, the position and direction of the mask are calculated in real time, so that the same sequence has different mask modes in different rounds, greatly improving the data reuse rate. The experiment compared the training results of static masking and dynamic masking and found that the two have similar performance, but dynamic masking has a more efficient advantage. Therefore, dynamic masking was selected in subsequent experiments. Figure 7 shows that dynamic masking achieves training data augmentation by changing the mask position of the same sequence in different epochs, thereby improving the model training effect.

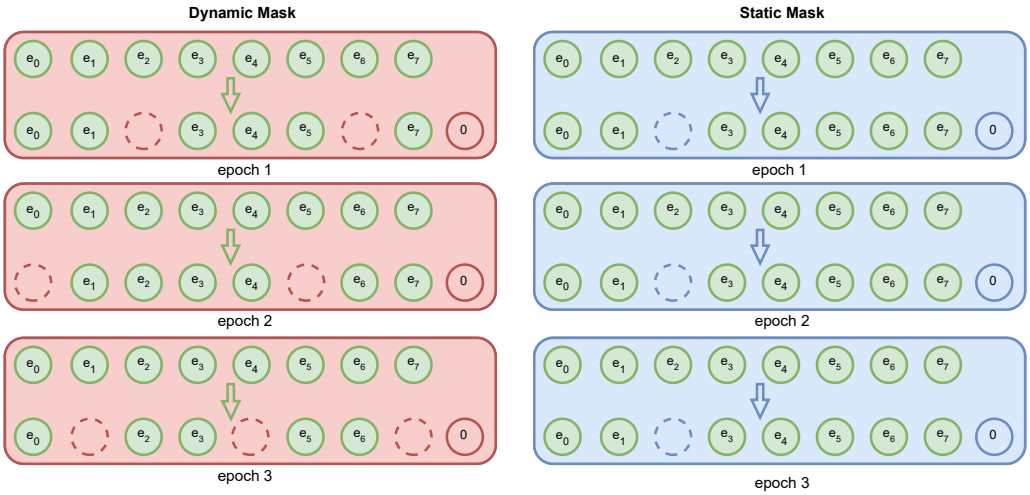

Figure 7. Comparison of dynamic/static masking mechanisms.

The core advantage of dynamic masking over static masking lies in its ability to achieve coordinated optimization of data utility and model generalization through a periodic mask reconstruction mechanism. During the pre-training stage of MASKT, the static masking strategy predefines a fixed mask pattern for each sequence, resulting in a one-to-one correspondence between sequences and mask patterns. This severely restricts the reuse efficiency of data features. In contrast, dynamic masking generates differentiated mask positions and orientations in real time during each training cycle (epoch), as illustrated in Figure 4. This mechanism enables a single sequence to be represented from multiple perspectives, yielding two key benefits:

1. Equivalent data augmentation: By randomly transforming the mask patterns during training, dynamic masking effectively expands the diversity of training samples, pushing data utilization closer to its theoretical upper bound.
2. Feature learning smoothing: By forcing the model to decouple its dependence on fixed contextual information, dynamic masking promotes the formation of robust feature representations, significantly reduces the risk of overfitting, and lowers RMSE in noise testing scenarios.

The efficiency gain in training convergence speed stems from the real-time computation paradigm, which eliminates the need to pre-store mask matrices. Meanwhile, improvements in generalization performance are attributed to the anti-memory effect, introduced by periodic perturbations at the epoch level. Overall, dynamic masking serves

as a preferred strategy for time-series data modeling, offering a more efficient learning framework for knowledge tracing tasks and enhancing both robustness and adaptability.

3.6. Noise Sequence Restoration Task

The goal of the MASKT noise restoration task is to generate the masked parts of the original sequence through decoding. Drawing on the BART model, an autoregressive decoder is used to achieve this goal. The sequence reconstruction process uses cross-entropy loss. For token filling and token deletion, conventional cross-entropy loss is used. For token random, an additional term is added to penalize incorrect prediction order. By adjusting the value of λ, the impact of reordering errors on the total loss is controlled. The aim is to improve the accuracy of the model's token value predictions while maintaining their correct order.

$$L = -\sum_{i=1}^{N}\left[(1-m_i)(1-d_i)y_i log(\hat{y}_i) + \lambda \sum_{j=1}^{N} s_{ij} \cdot f(\hat{y}_i, \hat{y}_j)\right] \tag{8}$$

In this context, N represents the total number of tokens in the sequence, $\hat{y}_i$ denotes the predicted output for the ith token, y_i denotes the actual value of the ith token, and m_i indicates whether the ith token is masked. If it is, it is 1; otherwise, it is 0. d_i indicates whether the i-th token has been deleted. If it is, it is 1; otherwise, it is 0. s_{ij} indicates whether the i-th token has been swapped with the j-th token. If it is, it is 1; otherwise, it is 0. λ represents the weight parameter used to balance the cross-entropy loss and the penalty for sequence errors. The $f(\hat{y}_i, \hat{y}_j)$ function is used to calculate the prediction error of swapping the positions of tokens i and j.

4. Multi-Layer Cross-Attention Embedding Module

Most existing knowledge tracing methods pursue high accuracy in predicting student performance, neglecting the consistency between changes in students' knowledge states and the learning process [17]. When decoding, cross-attention in the model fully captures the relevant information in the input sequence to obtain a better context representation. However, cross-attention requires different levels of granularity of information for different tasks. Therefore, in order to improve the decoder's ability to interpret the encoder's input in the MASKT model, we introduce a node-dimension level cross-attention.

4.1. Node-Dimension Level Attention

In this paper, the interaction sequences after noise processing are further processed into node vectors and dimension vectors to fully explore the sequence correlation after different processing methods in hidden features, and then extract the spatial features in the learning process [9]. The embedding process of adjacent vector dimensions is shown in Figure 8.

After the embedding vector e_i^{δ} is processed, embedding matrix E_i^{δ} is obtained. After inputting noise encoder, noise feature vectors l_i^{δ} and $l_i^{\theta} = (l_i^f, l_i^{d_p}, l_i^r)$ are obtained. Define δ as the pair of adjacent interaction nodes (i, j) that correspond to each other. Here, i represents the central interaction, j represents the adjacent interaction, and p represents the current epoch number. Use the attention mechanism to learn the normalized weight α_{ij}^{δ} of different adjacent interaction nodes j for the current node i.

$$\alpha_{ij}^{\delta} = attention\left(l_i^{\theta\delta}, l_j^{\theta\delta}, \delta\right) \tag{9}$$

where attention denotes the computation process of attention weight, the feature vectors of the center interaction node i and the neighboring interactions j are spliced separately, and

the weight values are obtained after a nonlinear transformation. The computation process normalized by SoftMax function is as follows:

$$\alpha_{ij}^{\theta\delta} = \frac{e^{\omega_{ij}}}{\sum e^{\omega_{ik}^{\theta}}} \tag{10}$$

$$\omega_{ik}^{\theta} = \sigma(v_{\theta}^{T} \bullet [l_i^{\theta} \oplus l_k^{\theta}]) \tag{11}$$

where σ represents the activation function, $\oplus$ acts as a concatenation vector, $\theta = (f, d, r)$ and v_{θ}^{T} are attention vectors after noise processing, and $k \in N_i^{\theta}$ and N_i^{θ} represent the set of adjacent interacting nodes i. For adjacent interactions δ, the weights α_{ij}^{δ} aggregate the feature vectors of adjacent interacting nodes, and the node embedding is e_i^{δ}.

$$e_i^{\theta\delta} = \sigma\left(\sum \alpha_{ij}^{\theta\delta} l_i^{\theta\delta}\right) \tag{12}$$

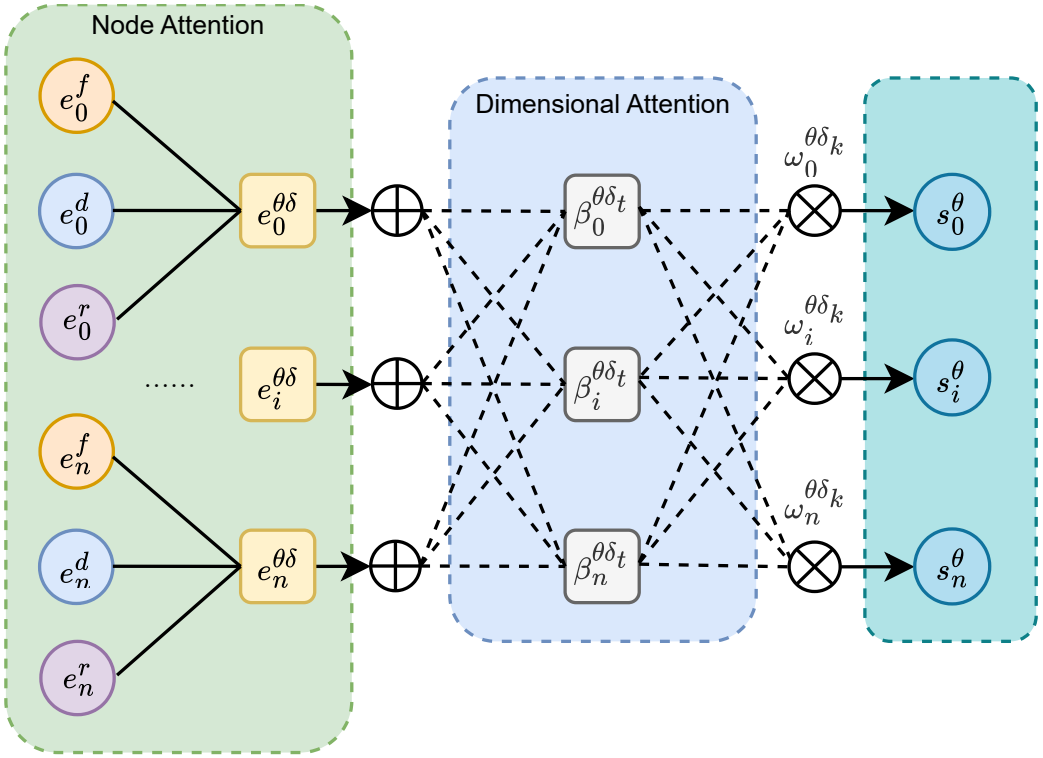

Figure 8. Node dimension vector embedding graph.

The adjacent interactive node set $\{\delta_1, \delta_2, \ldots \delta_t\}$ is used to obtain the different dimensional embeddings $\{e_i^{\theta\delta_1}, e_i^{\theta\delta_2}, \ldots e_i^{\theta\delta_t}\}$ for question i. Finally, the question embeddings are fused with the different dimensional question embeddings. Different dimensions have different weights. The features of questions with different dimensions are embedded according to attention as input, and the normalized weight of the question i is calculated based on the attention of the dimension:

$$\beta_i^{\theta\delta_t} = attention\left(e_i^{f\delta_t}, \quad e_i^{d\delta_t}, \quad e_i^{r\delta_t}\right) \tag{13}$$

Among them, attention refers to the process of calculating the attention weights for different dimensions. The feature embeddings extracted from different semantic features are subjected to linear transformations. The title embeddings are multiplied by the attention

vector v_d to obtain the weights, which are then normalized using the softmax function to obtain the final embeddings.

$$\beta_i^{\theta\delta_j} = \frac{e^{\omega_i^{\theta\delta_j}}}{\Sigma_{k=1}^{p} e^{\omega_i^{\theta\delta_k}}} \tag{14}$$

$$\omega_i^{\theta\delta_k} = v_d^T \bullet tanh\left(We_i^{\theta\delta_t} + b\right) \tag{15}$$

Here, v_d^T is the dimension attention vector, W is the weight matrix, b is the bias vector, and $\beta_i^{\theta\delta_j}$ represents the degree of interaction between different dimensions; the higher the degree of interaction, the more similar the content. The attention weight topic $\beta_i^{\theta\delta_j}$ is weighted and summed across different dimension embeddings to obtain the final topic embedding vector s_i^θ.

$$s_i^\theta = \sum_{j=1}^{t} \beta_i^{\theta\delta_j} e_i^{\theta\delta_j} \tag{16}$$

The embedding encoding is represented as $E \in R^{m\times d}$. In the Transformer, the dimensions of the unified embedding encoding and the embedding position information are obtained to form the complete input sequence z_i.

$$z_i = x_i \oplus p_i \oplus s_i^\theta \tag{17}$$

where $x_i \in E$ is the student interaction sequence embedding, and p_i is the position information embedding matrix. The position encoding matrix p_i is alternately generated using cosine and sine functions.

4.2. Historical Answer Sequence Attention

The self-attention mechanism provides a foundation for explaining model prediction results by generating weights based on the model's self-learning. In this paper, the self-attention mechanism is used to model the learner's answer sequence while introducing the relative position information of the answer interaction. In the traditional model, the question-embedding vector at time step t is denoted as s_t^θ, and the student's answer situation a_t is incorporated into s_t^θ to obtain the embedding vector r_t^θ. The position embedding matrix P is introduced to locate the relative position of the answer interaction vector, and the interaction vector after adding the position matrix is denoted as $r_t^{\dot{\theta}} = r_t^\theta + P_t$, where P_t represents the position embedding at the t-th time step. By calculating the correlation weights between the historical answer records of the current question, the historical interaction vectors are aggregated to obtain the question embedding information s_t^θ at the current time step, which is mapped to the query vector (q). The historical interaction vectors r_t^θ are mapped to the key vector (k) and value vector (v), and the attention weights γ_i between the current question and the historical answer sequences at time step i are calculated.

$$r_t^\theta = \begin{cases} s_t^\theta \oplus 0, & a_t = 1 \\ 0 \oplus s_t^\theta, & a_t = 0 \end{cases} \tag{18}$$

$$\gamma_i = \frac{e^{\omega_i}}{\sum_{k=1}^{t-1} e^{\omega_k}} \tag{19}$$

$$\omega_i = \frac{\left(W^Q e_t^\theta\right)^T \cdot W^K r_t^{\dot{\theta}}}{\sqrt{d}} \tag{20}$$

$$o_\tau = \sum_{i=1}^{t-1} \gamma_i W^V r_t^{\dot{\theta}} \tag{21}$$

Here, $W^Q \in R^{d^*d}$ denotes the mapping matrix of q, and $W^K \in R^{2d*d}$ denotes the mapping matrix of k. Finally, the historical interaction vectors are summed up according to the attention weights to obtain the learning state vectors related to the current answer in the attention module. $W^W \in R^{d^*d}$ denotes the mapping matrix of v.

5. Fusion Encoder

The main function of MASKT's knowledge encoder and response interaction encoder is to receive noise tasks and vectors processed by multidimensional attention, and then perform training to help the model mine the learner's knowledge state. The formulas for the knowledge encoder and interaction encoder are as follows:

$$En_i^Q = g_i^Q(x_i, m, t) \tag{22}$$

$$En_j^C = g_j^C(s_i^\theta, o_t, m, t) \tag{23}$$

where m is the position mask after that time step, and the attention weights are cleared to zero. t represents the current time step, $i \in (0, t+1)$, $j \in (0, t)$. The question encoder includes the question information for the next time step. Set a fusion encoder to combine the output results of the question and interaction encoders as a new prediction component. Q: g_i^Q, K: g_i^Q, and V: g_i^C correspond to Q, K, and V in the attention module. Among them, Q represents the query, K represents the keyword, and V represents the value. All three are derived from the input vector transformation. A fully connected layer is added to extract the representations of questions and interactions, and a nonlinear transformation is performed to project them into a high-dimensional space; finally, the prediction probability $\hat{y}_{t+1}$ is output through a sigmoid function. The prediction results are as follows:

$$z_{t+1} = f(\mathrm{Q}, \mathrm{K}, \mathrm{V}) \tag{24}$$

$$\hat{y}_{t+1} = \sigma(W_2 \cdot Gelu(W_1 \cdot z_t + b_1) + b_2) \tag{25}$$

where W_1, W_2, b_1 and b_2 are trainable parameters, and z_{t+1} is the output vector.

5.1. Loss Function

Yeung et al. pointed out that reconstruction problems are prone to occur in conventional Deep Knowledge Tracing (DKT) predictions, leading to wave-like changes in learners' knowledge state predictions, which can mislead knowledge state explanations [18]. To promote the model's effective convergence to the global optimum, a binary cross-entropy loss function is used to measure the distance between the model's predicted probability distribution and the actual labels. The performance is evaluated by comparing the predicted probability of correctly answering the next knowledge point with the actual probability of the knowledge point. The loss function of the MASKT model is defined as y_t and $\hat{y}_t$, as follows:

$$\mathrm{L}_p = -\sum_t (y_t \cdot log\hat{y}_t + (1 - y_t) log(1 - \hat{y}_t)) \tag{26}$$

For each sample, t, the model calculates the cross-entropy between the predicted probability distribution $\hat{y}_t$ and the true label y_t. There is a correlation between historical answer records and prediction results. A regularization term is introduced to balance the growth rate of model parameters and improve the model's generalization ability.

Given the prediction task of the model predicting the t-th time step, the historical answer sequence is $(a_1, a_2 \ldots a_{t-1})$, and the correlation weight of the first $t-1$ time step is $(\gamma_1, \gamma_2, \ldots, \gamma_{t-1})$. The feature screening relevance is calculated by adjusting the feature correlation coefficient weight value, and the feature vector e_i^{δ} of the historical record is calculated by the learnable parameter $w_i \in R^d$. The feature correlation weight is calculated to obtain the feature correlation score ϕ_i of the i-th historical record:

$$\phi_i = \sigma(w^T e_i^{\delta}) \tag{27}$$

The feature correlation weight ϕ_i is fused with the original attention weight $\beta_i^{\theta\delta_t}$ to generate the historical feature correlation composite weight η_i:

$$\eta_i = \frac{\beta_i \cdot \phi_i}{\sum_{j=1}^{t-1} \beta_j \phi_j} \tag{28}$$

Considering the correlation between the current question and the historical answer record, the overall historical answer situation is recorded as a_i, and the historical feature correlation weight value is defined as $\dot{s}_t$:

$$\dot{s}_t = \sum_{i=1}^{t-1} \eta_i a_i \tag{29}$$

At the current time step t, the smaller the difference between $\dot{s}_t$ and the predicted value $\hat{y}_t$, the higher the correlation of the prediction result. The root mean square deviation of the model prediction value $\hat{y}_t$ and the weighted value s_t of the overall historical answer results is used as the loss function. The hyperparameter λ is added to balance the model prediction performance and correlation, and the loss function is as follows:

$$\mathrm{L}_s = \frac{\sqrt{\sum_{i \in B} (\hat{y}_t - \dot{s}_t)^2}}{|B|} \tag{30}$$

$$\mathrm{L} = (1-\lambda)\mathrm{L}_p + \lambda \mathrm{L}_s \tag{31}$$

where B represents all interactions in a batch, and $|\ B\ |$ represents the sum of all sequence lengths in a batch, i.e., the total number of interactions.

5.2. Relevance Index

In order to further quantify the relevance of the model, the basic idea of balancing the prediction results and feature relevance is proposed, and a feature relevance measurement indicator is proposed: relevance (Balance). The relevance is intended to measure the extent to which the model promotes the prediction results in terms of feature screening features. First, if the difference between the prediction result $\hat{y}_t$ at time step t and the weighted value $\dot{s}_t$ of the overall relevance of historical answers is less than or equal to the specified balance factor λ, then the prediction result is considered to have high relevance; otherwise, it is considered to be irrelevant. The relevance is further defined as the proportion of relevant prediction results in all prediction results. Therefore, the calculation of relevance is as follows:

$$I(t) = \begin{cases} 1, & \text{if } |\dot{s}_t - y_t| \leq \lambda \\ 0, & \text{otherwise} \end{cases} \tag{32}$$

$$Balance_{\lambda} = \frac{1}{n} \sum_{t=1}^{n} I(t) \tag{33}$$

The larger the relevance value, the higher the promotion effect of the model based on key features on the model.

To further investigate the relationship between the regularization factor λ and feature relevance balance in MASKT, the relevance under fixed λ values for different models is presented in a heatmap, as shown in Figure 9. When comparing the correlation differences of multiple models under different λ values across various datasets, MASKT demonstrates global optimality: it consistently ranks first in performance across all datasets and all λ values (0.0–0.50), with the highest peak value (e.g., 98.2 in ASSIST15), and is the only model that maintains a correlation above 97.0 in multiple high-λ scenarios ($\lambda \geq 0.40$). Its robustness to performance decay is also higher than in other models. Under high $\lambda = 0.50$ scenarios, its average performance remains at 96.5, significantly outperforming the unprocessed model MSKT (94.9), and improving by over 17 points compared to traditional models like Dynamic Key-Value Memory Networks (DKVMN), whose average correlation value decays to 79.0. Parameter adaptability: In the critical λ range of 0.25–0.40, the average performance improvement reaches 9.2%, with stability fluctuations within 1.4 points, indicating that its balancing mechanism can adaptively adjust knowledge representation weights.

Compared to traditional models, DKT and IRT, which rely on simple recurrent networks or statistical methods, may struggle to capture long-term sequence dependencies. For example, in ASSIST12, DKT achieves only 9.6 at $\lambda = 0.05$, resulting in weak performance at low λ and overfitting at high λ. In terms of attention mechanisms, SAKT and AKT introduce attention but do not optimize multi-scale feature fusion. For example, in EdNet, SAKT fluctuates by 20.3 points at $\lambda = 0.20$–0.30, causing local performance oscillations. DKVMN and Sequential Key-Value Memory Networks (SKVMN) experience knowledge update conflicts at high λ values; for instance, DKVMN in ASSIST09 drops to 66.9 at $\lambda = 0.50$, reflecting the sensitivity of the memory module to balancing parameters, and leading to degradation in the memory network.

Compared to traditional models, such as DKT and IRT, which rely on simple recurrent networks or statistical methods, these approaches often struggle to capture long-term sequence dependencies. For example, in the ASSIST12 dataset, DKT achieves only 9.6 at $\lambda = 0.05$, resulting in weak performance under low λ and overfitting under high λ conditions. In terms of attention mechanisms, although SAKT and AKT introduce attention structures, they do not optimize multi-scale feature fusion. For instance, in the EdNet dataset, SAKT exhibits fluctuations of 20.3 points when λ varies from 0.20 to 0.30, leading to local performance instability. Moreover, DKVMN and SKVMN suffer from knowledge update conflicts under high λ values. For example, in the ASSIST09 dataset, DKVMN's performance drops to 66.9 at $\lambda = 0.50$, reflecting the sensitivity of memory modules to balance parameters, which ultimately leads to degradation in memory network performance.

To further observe how the trade-off factor λ in MASKT affects prediction performance and feature correlation, the AUC and correlation, corresponding to different λ values, are presented in a scatter plot, as shown in Figure 10. As illustrated in Figure 10, as the regularization parameter λ increases (from 0 to 0.5), the feature correlation (BA) across all datasets significantly improves (BA $<$ 40% at low λ values and $>$92% at high λ values), whereas the AUC exhibits an increasing trend initially, followed by a slight decline. The optimal λ range is concentrated between 0.3 and 0.4. For example, in the Algebra05 dataset, the model achieves optimal performance at $\lambda = 0.4$, with a subsequent slight decrease in AUC; this is possibly due to the clear structure of the questions, the strong logical consistency of knowledge points, the low data noise, and the model's ability to capture effective features. In the ASSIST09 dataset, MASKT reaches peak performance for both

metrics (AUC = 0.7794, BA = 99.5%) at $\lambda = 0.3$, with performance declining as λ further increases. This phenomenon may be attributed to skewed data distribution, where over-balancing leads to overfitting of the majority class and reduced generalization ability. On the ASSIST12 dataset, MASKT demonstrates a sudden AUC increase to 0.7680 at $\lambda = 0.3$, which is likely due to the high sparsity of interaction behaviors. In this case, λ adjustment reveals the inherent trade-off between accuracy and robustness. For the ASSIST15 dataset, MASKT maintains a BA consistently above 97.8% within the λ range of 0.3–0.4, representing the highest correlation among all datasets. However, the corresponding AUC remains notably low (AUC $\leq$ 0.7453), which may result from the impact of high annotation noise—allowing the model to achieve superficial feature balance while still exhibiting relatively low accuracy and discriminative capability. Finally, in the EdNet dataset, the model reaches its AUC peak (0.7714) at $\lambda = 0.35$, after which both AUC and BA decline as λ increases. This may be explained by the large dataset size and high diversity, where appropriate λ tuning can achieve a trade-off between correlation and performance, while excessive balancing dilutes information from difficult or minority samples.

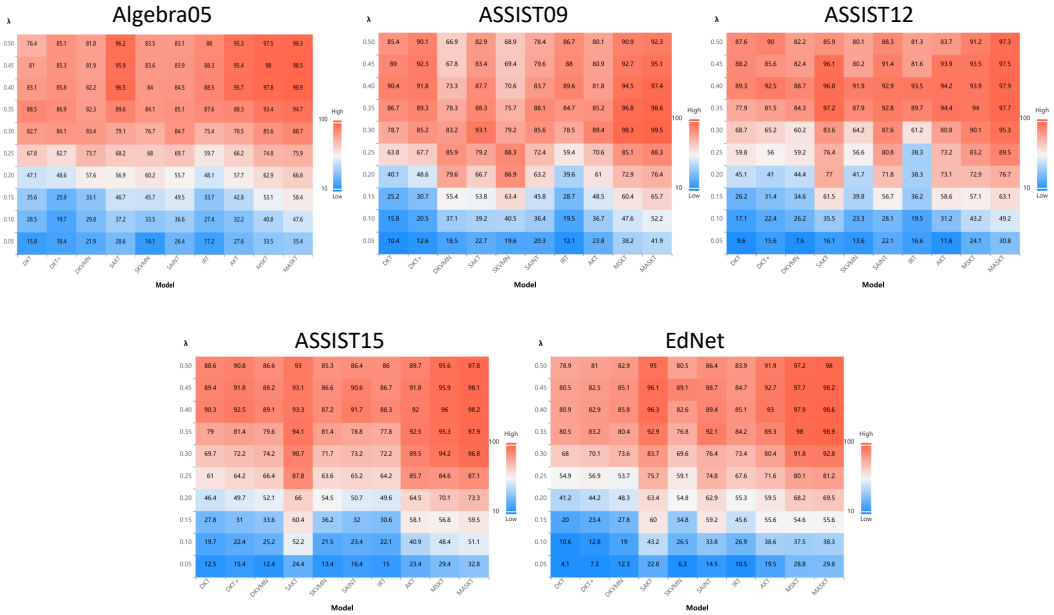

Figure 9. Different feature correlation values of multiple models on different datasets.

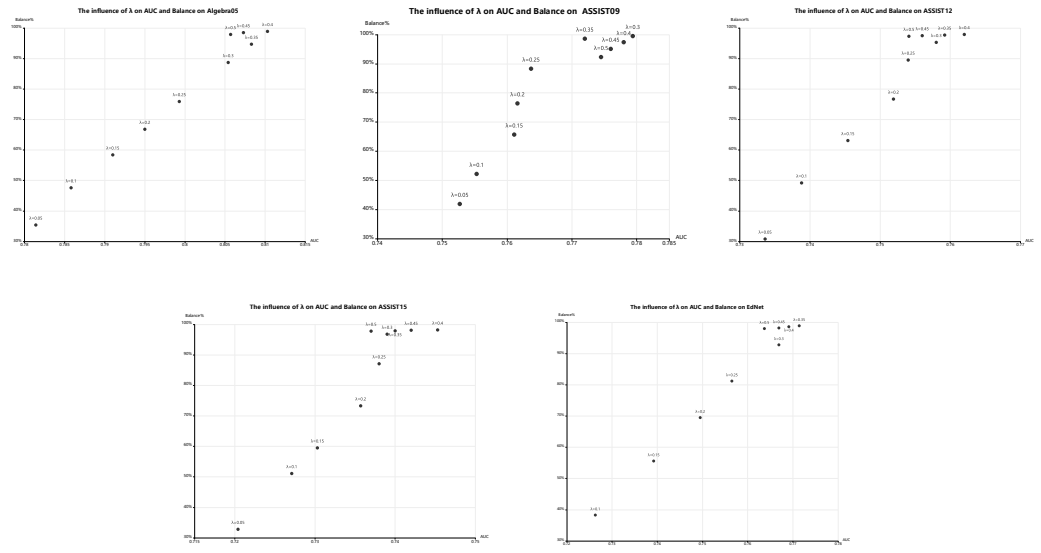

Figure 10. The impact of AUC and balance of λ on multiple datasets.

6. Experiment

6.1. Experimental Setup

In the experiment 20% of the student sequences were randomly selected as a test set to evaluate the model and the remaining 80% of the dataset was used for 5-fold cross-validation. ADAM was chosen as the optimizer to train the model. The maximum value of the training period was set to 300 and an early-stopping strategy was used to shorten the training process. The embedding dimension, hidden state dimension, and two-dimensionality of the prediction layer were set to [64, 128], the learning rate and dropout were set to [0.001, 0.0001] and [0.05, 0.1, 0.3, 0.5], respectively, the number of blocks and attention headers were set to [1, 2, 4] and [4, 8], the batch size was 128, and the seed was set to [42, 3407]. The model was implemented in PyTorch 2.1.0 and training was performed on an NVIDIA RTX 3080 GPU device (NVIDIA Corporation, Santa Clara, CA, USA). Similar to all existing DKT studies, AUC was used as the primary evaluation metric and RMSE as a secondary metric.

6.2. Dataset

The dataset uses four public datasets commonly used in knowledge tracking models to validate MASKT; the statistics of the dataset are shown in Table 3.

Table 3. Dataset-related information.

Dataset	Students	Knowledge	Interface
Algebra05 [19]	574	436	607,026
ASSIST2009 [20]	4151	110	325,673
ASSIST2012 [21]	27,485	265	53,065
ASSIST2015 [22]	19,840	100	683,801
EdNet [23]	784,309	13,169	131,317,236

Algebra05 [19]—the KDD Cup 2010 EDM Challenge dataset with 809,694 interactions from 574 students on 210,710 problems; ASSIST2009 [20]—from the ASSISTMENTS online tutoring system, which, after removing duplicate records, contains 4151 interactions from 4151 students on 110 problems; ASSIST2012 [21]—in this dataset, each question is related to only one skill, but a skill still corresponds to several questions (after the same data processing as ASSIST09, it has 2,709,436 exercises, 27,485 students, 265 skills, and 53,065 questions); ASSIST2015 [22]—contains 683,801 interactions of 19,840 students on 100 questions; EdNet [23]—a large-scale hierarchical dataset of students' interactions on 100 questions collected by the Santa Artificial Intelligence Guided Learning system, which collects a large-scale hierarchical student activity dataset containing 131,317,236 interactions from 784,309 students.

6.3. Comparison Experiment

As shown in Table 4, in order to further evaluate and compare the prediction performance of each model on different datasets, the optimal AUC values were compared on five datasets commonly used in the knowledge tracking domain. The Separated Self-AttentIve Neural Knowledge Tracing (SAINT) and AKT and SAKT models perform relatively better than the other methods on each dataset, and it is hypothesized that these methods may have deeper self-attention layers and more complex hidden layer structures to improve the predictive performance of the models at the cost of interpretability. The MSKT method is an unfeatured filtered model, and a significant gap between the predictive performance and the overall model can be observed. MASKT method uses parameters to balance the weight of the noise function in the pre-training to optimize the model compre-

hension and achieve the purpose of improving the prediction performance. The optimal prediction performance is more than 0.02 higher compared to other methods.

Table 4. Comparison of AUC values ± confidence intervals and paired *t*-tests under different datasets.

Method	Algebra05		ASSIST09		ASSIST12		ASSIST15		EdNet	
	Mean ± Std	*t* *p*	Mean ± Std	*t* *p*	Mean ± Std	*t* *p*	Mean ± Std	*t* *p*	Mean ± Std	*t* *p*
DKT	0.7638 ± 0.0167	7.38 0.0020	0.7631 ± 0.0214	1.54 0.1986	0.7504 ± 0.0189	1.23 0.2877	0.7271 ± 0.0246	1.50 0.2079	0.7012 ± 0.0221	6.18 0.0035
DKT+	0.7394 ± 0.0193	7.41 0.0017	0.7599 ± 0.0174	2.17 0.0962	0.7541 ± 0.0188	0.90 0.4203	0.7371 ± 0.0205	0.84 0.4475	0.7254 ± 0.0234	3.90 0.0176
DKVMN	0.7653 ± 0.0205	5.23 0.0063	0.7621 ± 0.0187	1.82 0.1446	0.7473 ± 0.0206	1.47 0.2149	0.7268 ± 0.0212	1.69 0.1666	0.6857 ± 0.0253	6.65 0.0027
SAKT	0.7645 ± 0.0196	5.06 0.0072	0.7571 ± 0.0192	2.21 0.0918	0.7491 ± 0.0233	1.16 0.3105	0.7240 ± 0.0207	1.99 0.1176	0.6976 ± 0.0276	5.49 0.0054
SKVMN	0.7521 ± 0.0231	5.25 0.0063	0.7432 ± 0.0245	2.90 0.0441	0.7594 ± 0.0291	0.19 0.8591	0.7384 ± 0.0198	0.72 0.5125	0.7164 ± 0.0258	4.37 0.0120
SAINT	0.7751 ± 0.0157	4.47 0.0111	0.7653 ± 0.0179	1.59 0.1879	0.7584 ± 0.0266	0.28 0.7951	0.7421 ± 0.0304	0.22 0.8377	0.7411 ± 0.0229	2.69 0.0545
IRT	0.7032 ± 0.0267	8.91 0.0009	0.7254 ± 0.0325	3.44 0.0260	0.7051 ± 0.0291	3.58 0.0183	0.7079 ± 0.0273	2.68 0.0551	0.6940 ± 0.0316	5.06 0.0072
AKT	0.7677 ± 0.0145	6.22 0.0034	0.7679 ± 0.0157	1.48 0.2133	0.7532 ± 0.0196	0.94 0.4015	0.7381 ± 0.0201	0.74 0.4995	0.7649 ± 0.0189	0.70 0.5230
MSKT	0.8024 ± 0.0135	1.25 0.2796	0.7548 ± 0.0147	3.05 0.0380	0.7546 ± 0.0175	0.87 0.4352	0.7269 ± 0.0191	1.84 0.1399	0.7603 ± 0.0186	1.17 0.3071
MASKT	0.8103 ± 0.0098		0.7794 ± 0.0124		0.7620 ± 0.0168		0.7453 ± 0.0159		0.7714 ± 0.0152	

As shown in Table 5, the root mean square error (RMSE) is used to measure the difference between the predicted and true values and to assess the model prediction accuracy. The lower the RMSE value within a certain range, the more accurate the prediction value is. The RMSE values of the comparison dataset in Table 5 are basically consistent with the prediction performance in Table 4. It is consistent with the feature that, the better the prediction performance, the lower the error.

Table 5. Comparison of RMSE values ± confidence intervals and paired *t*-tests under different datasets.

Method	Algebra05		ASSIST09		ASSIST12		ASSIST15		EdNet	
	Mean ± Std	*t* *p*	Mean ± Std	*t* *p*	Mean ± Std	*t* *p*	Mean ± Std	*t* *p*	Mean ± Std	*t* *p*
DKT	0.4113 ± 0.0218	2.51 0.0665	0.4372 ± 0.0226	0.32 0.7660	0.4236 ± 0.0207	2.41 0.0740	0.4275 ± 0.0233	0.81 0.4640	0.4428 ± 0.0255	1.83 0.1409
DKT+	0.4068 ± 0.0209	2.16 0.0976	0.4339 ± 0.0217	0.02 0.9860	0.4182 ± 0.0238	1.67 0.1702	0.4231 ± 0.0199	0.47 0.6635	0.4387 ± 0.0201	1.88 0.1339
DKVMN	0.4026 ± 0.0210	1.67 0.1700	0.4357 ± 0.0235	0.18 0.8674	0.4153 ± 0.0224	1.51 0.2064	0.4305 ± 0.0253	0.98 0.3840	0.4327 ± 0.0272	0.94 0.3990
SAKT	0.3972 ± 0.0128	1.65 0.1749	0.4361 ± 0.0151	0.31 0.7711	0.4091 ± 0.0144	1.33 0.2552	0.4196 ± 0.0176	0.13 0.9040	0.4165 ± 0.0162	−1.51 0.6370
SKVMN	0.4034 ± 0.0218	1.70 0.1647	0.4419 ± 0.0234	0.73 0.5053	0.4194 ± 0.0257	1.64 0.1768	0.4251 ± 0.0222	0.62 0.5680	0.4268 ± 0.0269	0.48 0.6560
SAINT	0.3997 ± 0.0196	1.47 0.2156	0.4342 ± 0.0213	0.05 0.9658	0.4035 ± 0.0207	0.41 0.7032	0.4287 ± 0.0198	1.04 0.3570	0.4178 ± 0.0241	−1.24 0.8240
IRT	0.4205 ± 0.0233	3.13 0.0354	0.4653 ± 0.0248	2.64 0.0578	0.4251 ± 0.0268	2.00 0.1161	0.4419 ± 0.0231	2.05 0.1100	0.4596 ± 0.0202	3.89 0.0178
AKT	0.3984 ± 0.0136	1.78 0.1493	0.4324 ± 0.0149	−1.17 0.8750	0.4106 ± 0.0127	1.69 0.1673	0.4207 ± 0.0166	0.27 0.8030	0.4169 ± 0.0152	−1.49 0.6490
MSKT	0.3896 ± 0.0096	0.58 0.5934	0.4359 ± 0.0125	0.32 0.7665	0.4163 ± 0.0117	2.70 0.0540	0.4245 ± 0.0158	0.75 0.4970	0.4257 ± 0.0131	0.76 0.4880
MASKT	0.3865 ± 0.0083		0.4337 ± 0.0106		0.3994 ± 0.0092		0.4185 ± 0.0113		0.4206 ± 0.0098	

As shown in Table 6, comparing the ablation experiment prediction performance using the MASKT method, the single-noise-processing method performed slightly lower than the combined MASKT method overall across the datasets. The performance of MASKT-F was closest to the full method performance across multiple datasets. It is hypothesized that this may be due to the fact that the features before sequence filling are preserved and the MASKT method is able to capture valid features in the filled features. The MASKT-D

method is slightly shorter compared to the other two methods, possibly due to the loss of features due to sequence deletion affecting its performance.

Table 6. Comparison of Best AUC values in ablation experiments.

Method	Algebra05	ASSIST09	ASSIST12	ASSIST15	EdNet
MASKT-F	0.8047	0.7751	0.7341	0.7451	0.7654
MASKT-D	0.7992	0.7796	0.7473	0.7337	0.7631
MASKT-R	0.7949	0.7841	0.7482	0.7401	0.7573
MASKT	0.8103	0.7944	0.7620	0.7553	0.7714

As shown in Table 7, based on the comparative ablation experiments of the MASKT method, a dynamic masking mechanism is added to the MASKT-D single-noise-processing method, which is denoted as MD-Dynamic. Compared with the static masking method, the dynamic masking method enlarges each epoch by 10 times on the basis of the original one. However, too many epoch iterations bring the risk of overfitting, so an early-stopping strategy is added to the dynamic masking mechanism. Through the experimental results, it is observed that the prediction performance of MD-Dynamic is slightly stronger than that of the MASKT-D single-noise-processing method, and it can reach the same effect as the complete MASKT method on some datasets. Based on this method, the dynamic masking mechanism is migrated to the full MASKT method, and the experimental results demonstrate that the dynamic MASKT fine-tuning method obtains better performance in the finite space of the relevant datasets in the experiments.

Table 7. Comparison of best AUC values in static/dynamic mask experiments.

Method	Algebra05	ASSIST09	ASSIST12	ASSIST15	EdNet
MASKT-D	0.7992	0.7796	0.7473	0.7337	0.7631
MD-Dynamic	0.8095	0.7953	0.7579	0.7426	0.7694
MASKT	0.8103	0.7944	0.7620	0.7553	0.7714
MASKT-DY	0.8196	0.8109	0.7673	0.7518	0.7786

6.4. Performance Evaluation

The MASKT model introduces a multi-layer attention network to find the link between topic–skill relationships, represents topic and skill information models, and overall performs better than using a single skill information or topic information model. Figure 11 shows the results of the overall performance evaluation of the MASKT model. Experiments using both AUC and RMSE for more comprehensive comparisons reveal significant differences in performance on most datasets. Examples of the results are presented here: Figure 11a shows on the Algebra05 dataset; the SAINT model exhibits a superior prediction performance, presumably because Transformer captures the complex relationship between exercises and answers more effectively. The SKVMN model outperforms SAINT and other datasets on the ASSIST2012 dataset, possibly because the SKVMN model has enhanced time-series capabilities and loop modeling to better capture the dependencies between knowledge states. In addition the AKT model's introduction of problem–skill relationships in the Transformer architecture allows the model to capture learning features well, and is in the second or third position overall. Finally, the MASKT model employs a specific data cleaning model that introduces an effective self-supervised learning method and hierarchical attention mechanism to guide the model to enhance knowledge feature extraction and knowledge state representation. The comparative metrics based on AUC and RMSE MASKT consistently outperforms all the models, and is able to maintain a stable performance beyond 0.2 percentage points on the AUC of all the models.

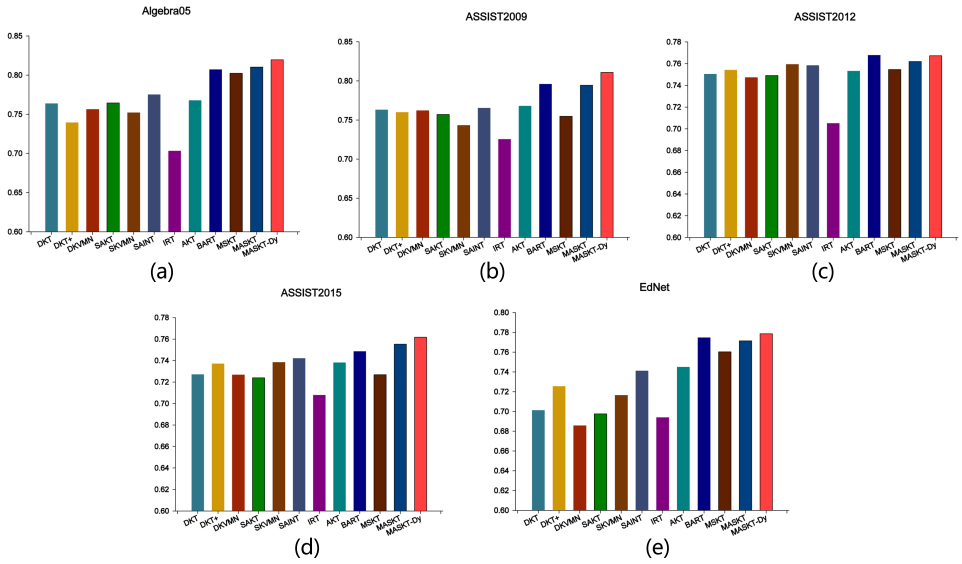

Figure 11. Comparison of AUC performance of MASKT model dataset. (**a**) Performance on the Algebra05 dataset; (**b**) Performance on the ASSIST2009 dataset; (**c**) Performance on the ASSIST2012 dataset; (**d**) Performance on the ASSIST2015 dataset; (**e**) Performance on the EdNet dataset.

The ablation experiment performance comparison shown in Figure 12 is designed to explore the optimal λ value by adjusting the regularization proportion λ within the loss function. For newly introduced datasets, conclusions are drawn by tuning the λ parameter within the MASKT model and observing its impact on prediction performance. As shown in Figure 12, across various datasets, once λ exceeds a specific threshold, the overall model performance experiences a notable decline. The core objective of the MASKT model is to balance prediction performance and sequence feature correlation, thereby enhancing both interpretability and robustness. To verify this balancing mechanism, comparative experiments are conducted among single-noise-processing methods (filling, delete, random), the full MASKT model with multi-noise processing, and a dynamic training strategy (denoted as Dynamic) serving as a baseline.

Figure 12a demonstrates that the MASKT model achieves relatively superior performance, highlighting the effectiveness of noise restoration on this dataset. However, the random noise method proves less effective. At higher λ values, the amplified validation loss weight exposes the shortcomings of the random method, which lacks a consistent processing strategy and thus suffers from conflict-driven degradation. In contrast, MASKT utilizes a dynamic weighting mechanism within its multi-noise-processing framework to suppress performance fluctuations; nevertheless, for $\lambda > 0.5$, slight overfitting still occurs. As illustrated in Figure 12b, on the ASSIST09 dataset, optimal AUC performance is achieved for both the filling and delete single-noise-processing methods as well as the complete MASKT model at $\lambda = 0.4$, with AUC improvements of approximately 5% to 7%. At this stage, the dynamic–random method significantly outperforms other single processing approaches, with an AUC gain of roughly 3%. This may be attributed to the dominance of filling noise and delete noise in this dataset, combined with a uniform noise distribution, making the dynamic strategy well-suited to the noise characteristics under the optimal λ.

In Figure 12c, for the ASSIST12 dataset, when $\lambda = 0.5$, the random method exhibits a temporary improvement in performance, with the AUC increasing by 2%. However, MASKT subsequently shows an unstable downward trend, with the AUC decreasing by approximately 4%. This fluctuation stems from the high proportion of random noise present within the dataset. At $\lambda = 0.5$, the random method overemphasizes sequence

correlations, revealing its short-term adaptability. As λ increases further, the excessive dynamic shifting of sequences induced by random noise reduces the overall robustness of MASKT. As shown in Figure 12d, for the ASSIST15 dataset, when $\lambda > 0.5$, the random method exhibits a wavelike decline, with AUC oscillations of approximately $\pm 3\%$, while the MASKT model maintains relatively stable performance throughout.

In summary, the random method demonstrates lower overall stability compared to other noise-processing techniques. Its performance fluctuations align with established cognitive patterns: random strategies may exhibit localized effectiveness in simple or uniformly noisy environments, but struggle to adapt in complex, dynamic real-world scenarios where noise distributions are heterogeneous. Fundamentally, performance fluctuations are driven by the interplay of dataset-specific noise heterogeneity, loss weight sensitivity, and the robustness of the noise-processing mechanism. Future research will focus on introducing quantifiable noise distribution metrics to optimize the selection of λ thresholds, further enhancing both performance and model interpretability.

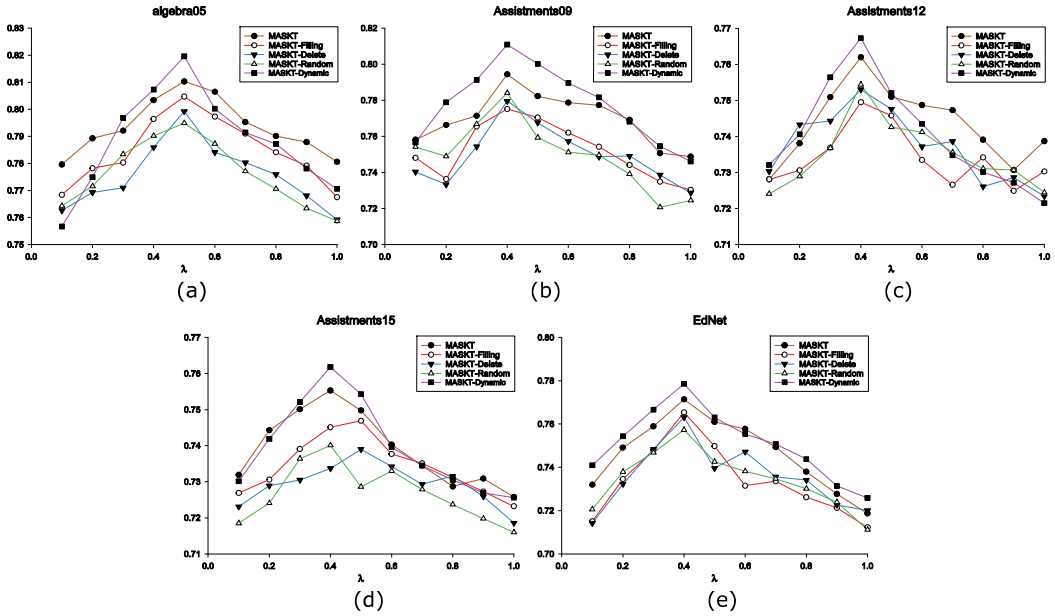

Figure 12. Comparison chart of model ablation experimental performance. (**a**) Ablation experiment results on the Algebra05 dataset (**b**) Ablation experiment results on the ASSIST09 dataset (**c**) Ablation experiment results on the ASSIST12 dataset (**d**) Ablation experiment results on the ASSIST15 dataset (**e**) Ablation experiment results on the EdNet dataset.

The MASKT model extends the topic embedding module based on the SAKT model to further improve the prediction performance. The overall performance is slightly better than other topic embedding models, and in some datasets, the performance is slightly worse than other models, such as DHKT [24], which is richer in mining features and more comprehensive in topic embedding by introducing semantic dimensional multi-layer attention correlation compared with single skill dimension correlation. MASKT performs best in the ASSIST09 and ASSIST12 datasets, and the difference with the optimal model in other datasets is about 0.2 percentage points, which indicates that MASKT has a high prediction performance while improving the interpretability of the model.

6.5. Comparison to BART

In order to compare the training effectiveness of MASKT and the current popular large language model BART, the prediction performance AUC was compared on five datasets with the BART model [7], MASKT-MA without the introduction of attentional processing, and MASKT with the introduction of the multidimensional attention mechanism, respec-

tively. As shown in Table 8, on the Algebra05 dataset, the AUC performance of the BART model is smaller than that of the MASKT performance. On the ASSIST2009 dataset, MASKT outperforms the BART and MASKT-MA models. On the ASSIST2012 dataset, MASKT and BART are essentially equal. For the ASSIST2015 dataset, MASKT also performs best. On the EdNet dataset, the performance is still the highest of the three methods. Thus, it can be seen that, without the introduction of the multidimensional attention mechanism, there is a slight gap between MASKT and the BART model in terms of prediction performance. However, there is a further improvement in the prediction performance of the model by introducing the multidimensional attention mechanism, which suggests that there is a positive impact of the multidimensional attention mechanism on the model's performance on these specific tasks.

Table 8. Comparison of AUC values with BART.

Method	Algebra05	ASSIST09	ASSIST12	ASSIST15	EdNet
BART	0.8071	0.7959	0.7678	0.7486	0.7747
MASKT-MA	0.8047	0.7944	0.7620	0.7518	0.7714
MASKT	0.8071	0.7959	0.7678	0.7486	0.7747

7. Conclusions

In exploring new preprocessing methods within the field of smart education and online learning, this study proposes an innovative approach to understanding the relationships between questions and skills across different dimensions and time steps, with the goal of optimizing the knowledge tracing process. The proposed method employs a Transformer-based bidirectional encoder combined with a data noise-processing module to construct self-supervised signals that reflect the characteristics of learners' historical behaviors. Through a multi-layer attention self-supervised learning framework, the model effectively captures more informative knowledge state representations, thereby enhancing its ability to predict learner performance.

In addition, this study introduces a random forest algorithm at the pre-training feature selection stage to filter feature embeddings with positive and negative correlations, aiming to further improve prediction accuracy. Three noise-processing strategies—F noise, D noise, and R noise—are designed to simulate potential erroneous behaviors exhibited by learners during the answering process under real-world conditions. The model's learning process is subsequently optimized through restoration tasks based on these simulated errors. Furthermore, a multidimensional attention mechanism is integrated into the cross-attention structure, leading to significant improvements in the model's predictive performance. Experimental results on public benchmark datasets demonstrate that the proposed method outperforms existing models in terms of both predictive accuracy and representation quality. Ablation studies are conducted to evaluate the individual contributions of each component and to examine the overall impact of self-supervised tasks on the model's performance. The MASKT framework replaces the conventional pre-training stage with a masked language modeling (MLM) strategy and utilizes a multi-layer attention network to enhance the model's capacity for understanding multidimensional semantic associations, thereby improving overall robustness. Future research will focus on refining the design of noise-processing ratios to further improve prediction accuracy and model interpretability. This research provides effective technical support for the advancement of smart education systems and the implementation of personalized learning solutions. Nevertheless, the MASKT method also presents certain limitations. The integration of a multi-layer attention mechanism with self-supervised learning tasks significantly increases computational complexity and memory consumption, particularly as the length of the learning history

sequence grows. Moreover, the training time is substantially longer compared to simpler knowledge tracing models, such as item response theory (IRT) or Bayesian knowledge tracing (BKT). Although the introduction of the random forest algorithm enhances feature selection by identifying important embeddings, the overall decision-making process of the model—especially the complex interactions within the deep multi-layer attention network—remains a "black box." The lack of intuitive interpretability regarding internal mechanisms may limit educators' trust in and understanding of the model's predictive outputs. Improving the interpretability of the MASKT model is identified as a crucial direction for future research.

Author Contributions: Conceptualization, H.W. and H.L.; methodology, H.L.; software, H.L.; validation, Y.G.; formal analysis, Y.G.; investigation, Y.G.; resources, H.W.; data curation, Y.G.; writing—original draft preparation, H.W.; writing—review and editing, Y.G.; visualization, H.W.; supervision, Z.Y.; project administration, Z.Y.; funding acquisition, H.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Shandong Provincial Natural Science Foundation, grant number ZR2023MF090; the Innovation Capacity Improvement Project for Technology-based SMEs in Shandong Province, grant number 2023TSGC0449; the Youth Innovation Team Development Plan for Universities in Shandong Province, grant number 2021QCYY003; and the Undergraduate Teaching Reform Research Project of Shandong Province, grant number Z2024301, M2022035.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available in edudata at https://edudata.readthedocs.io/en/latest/tutorial/zh/DataSet.html (accessed on 31 July 2025).

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

MASKT	Multi-layer Attention Self Supervised Knowledge Tracking Method
BKT	Bayesian knowledge tracking model
SAKT	self-attentive model for knowledge tracing
AKT	Attentive Knowledge Tracing.
PCA	Principal Component Analysis.

References

1. Yudelson, M.V.; Koedinger, K.R.; Gordon, G.J. Individualized bayesian knowledge tracing models. In Proceedings of the Artificial Intelligence in Education: 16th International Conference, AIED 2013, Memphis, TN, USA, 9–13 July 2013; pp. 171–180.
2. Piech, C.; Bassen, J.; Huang, J.; Ganguli, S.; Sahami, M.; Guibas, L.J.; Sohl-Dickstein, J. Deep knowledge tracing. In Proceedings of the 29th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 505–513.
3. Pandey, S.; Karypis, G. A self-attentive model for knowledge tracing. *arXiv* **2019**, arXiv:1907.06837. [CrossRef]
4. Ghosh, A.; Heffernan, N.; Lan, A.S. Context-aware attentive knowledge tracing. In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Virtual, CA USA, 6–10 July 2020; pp. 2330–2339.
5. Sun, J.; Zhou, J.; Liu, S.; He, F.; Tang, Y. Hierarchical Attention Network Based Interpretable Knowledge Tracing. *J. Comput. Res. Dev.* **2021**, *58*, 2630–2644. [CrossRef]
6. Tian, Z.; Zheng, G.; Flanagan, B. BEKT: Deep knowledge tracing with bidirectional encoder representations from transformers. In Proceedings of the International Conference on Computers in Education, Virtual, 22–26 November 2021.
7. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; Zettlemoyer, L. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv* **2019**, arXiv:1910.13461.

8. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.R.; Le, Q.V. Xlnet: Generalized autoregressive pretraining for language understanding. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 5753–5763.
9. Dong, L.; Yang, N.; Wang, W.; Wei, F.; Liu, X.; Wang, Y.; Gao, J.; Zhou, M.; Hon, H.W. Unified language model pre-training for natural language understanding and generation. In Proceedings of the 33rd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 13063–13075.
10. Mnih, A.; Hinton, G.E. A scalable hierarchical distributed language model. In Proceedings of the 22nd International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 8–10 December 2008; pp. 1081–1088.
11. Peters, M.E.; Neumann, M.; Zettlemoyer, L.; Yih, W.T. Dissecting contextual word embeddings: Architecture and representation. *arXiv* **2018**, arXiv:1808.08949. [CrossRef]
12. Sun, F.; Liu, J.; Wu, J.; Pei, C.; Lin, X.; Ou, W.; Jiang, P. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, Beijing, China, 3–7 November 2019; pp. 1441–1450.
13. Lee, W.; Chun, J.; Lee, Y.; Park, K.; Park, S. Contrastive learning for knowledge tracing. In Proceedings of the ACM Web Conference 2022, Lyon, France, 25–29 April 2022; pp. 2330–2338.
14. Lin, S.; Tian, H. Short-term metro passenger flow prediction based on random forest and LSTM. In Proceedings of the 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chongqing, China, 12–14 June 2020; Volume 1, pp. 2520–2526.
15. Zhu, Y.; Duan, J.; Li, Y.; Wu, T. Image classification method of cashmere and wool based on the multi-feature selection and random forest method. *Text. Res. J.* **2022**, *92*, 1012–1025. [CrossRef]
16. Liu, Z.; Liu, Q.; Chen, J.; Huang, S.; Gao, B.; Luo, W.; Weng, J. Enhancing deep knowledge tracing with auxiliary tasks. In Proceedings of the ACM Web Conference 2023, Austin, TX, USA, 30 April–4 May 2023; pp. 4178–4187.
17. Shen, S.; Liu, Q.; Chen, E.; Huang, Z.; Huang, W.; Yin, Y.; Su, Y.; Wang, S. Learning process-consistent knowledge tracing. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Singapore, 14–18 August 2021; pp. 1452–1460.
18. Yeung, C.K.; Yeung, D.Y. Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In Proceedings of the Fifth Annual ACM Conference on Learning at Scale, London, UK, 26–28 June 2018; pp. 1–10.
19. Stamper, J.; Pardos, Z.A. The 2010 KDD Cup Competition Dataset: Engaging the machine learning community in predictive learning analytics. *J. Learn. Anal.* **2016**, *3*, 312–316. [CrossRef]
20. Feng, M.; Heffernan, N.; Koedinger, K. Addressing the assessment challenge with an online system that tutors as it assesses. *User Model. User-Adapt. Interact.* **2009**, *19*, 243–266. [CrossRef]
21. Koedinger, K.R.; Baker, R.S.; Cunningham, K.; Skogsholm, A.; Leber, B.; Stamper, J. A data repository for the EDM community: The PSLC DataShop. *Handb. Educ. Data Min.* **2010**, *43*, 43–56.
22. King, D.R. Production implementation of recurrent neural networks in adaptive instructional systems. In Proceedings of the Adaptive Instructional Systems: Second International Conference, AIS 2020, Held as Part of the 22nd HCI International Conference, HCII 2020, Copenhagen, Denmark, 19–24 July 2020; pp. 350–361.
23. Choi, Y.; Lee, Y.; Shin, D.; Cho, J.; Park, S.; Lee, S.; Baek, J.; Bae, C.; Kim, B.; Heo, J. Ednet: A large-scale hierarchical dataset in education. In Proceedings of the Artificial Intelligence in Education: 21st International Conference, AIED 2020, Ifrane, Morocco, 6–10 July 2020; pp. 69–73.
24. Wang, T.; Ma, F.; Gao, J. Deep hierarchical knowledge tracing. In Proceedings of the 12th International Conference on Educational Data Mining, Montréal, QC, Canada, 2–5 July 2019.