

AAKT: Enhancing Knowledge Tracing with Alternate Autoregressive Modeling

Hao Zhou, Wenge Rong, Jianfei Zhang, Qing Sun, Yuanxin Ouyang and Zhang Xiong

Abstract—Knowledge Tracing (KT) aims to predict students’ future performances based on their former exercises and additional information in educational settings. KT has received much attention since it provides personalized experiences in educational situations. Simultaneously, the autoregressive modeling on the sequence of former exercises has been proven effective for this task. One of the primary challenges in autoregressive modeling for Knowledge Tracing is effectively representing the anterior (pre-response) and posterior (post-response) states of learners across exercises. Existing methods often rely on complex model architectures to update learner states using question and response records, along with additional information. In this study, we demonstrate that these states can be directly represented through autoregressive encodings on a question-response alternate sequence, without the need for additional structures on the autoregressive model. We refer to this resulting framework as Alternate Autoregressive Knowledge Tracing (AAKT). Additionally, we incorporate supplementary system information, such as question-related skills, into our framework through an auxiliary task, and include extra exercise details, like response time, as additional inputs. Our proposed framework is implemented using advanced autoregressive technologies from Natural Language Generation (NLG) for both training and prediction. We conduct experiments on four real-world KT datasets and AAKT consistently outperforms all baseline models in terms of AUC, ACC, and RMSE. We believe that our proposed framework shows significant promise in the field of KT, owing to its simplicity in implementation and effectiveness in performance.

Index Terms—Educational Technology, Knowledge Tracing, Autoregressive Models, Transformer

I. INTRODUCTION

IT is imperative for intelligent tutoring systems and online learning platforms to effectively depict the evolving knowledge status of students. To address this requirement, the fundamental task of Knowledge Tracing (KT) has been introduced [1]. KT involves tracing the mastery level of individual students over time and generating predictions regarding the accuracy of their responses to diverse questions based on their knowledge status. This process entails collecting the historical exercise sequences of students, computing their mastery levels, and forecasting the likelihood of correct responses to subsequent exercises. By executing Knowledge Tracing, educators gain insights that facilitate the provision of

personalized learning materials, identification of weaknesses, and targeted exercise recommendations.

Conventional methodologies within this domain mainly include Bayesian Knowledge Tracing (BKT) employing Hidden Markov Models [1], [2], [3], and Deep Knowledge Tracing (DKT) utilizing Deep Neural Networks [4] along with its derivative models [5], [6]. While the initial Knowledge Tracing models, such as BKT, exhibit ease of implementation and enhanced interpretability, their efficacy is hindered by their inability to capture the dynamic nature of knowledge status due to oversimplified assumptions concerning the learning process.

The initial DKT models use RNNs, particularly LSTM and GRU architectures, to represent students’ knowledge status [4], [7], [8]. However, RNN-based models concentrate on nearby exercises, limiting the ability to model long-term dependencies. To address this limitation, Sequential Key-Value Memory Networks (SKVMN) [9] incorporate LSTM with hops to capture extended dependencies within exercise sequences. Despite this improvement, it does not fully overcome the challenge posed by the limited attention span.

The Transformer architecture [10], known for its effectiveness in sequence-to-sequence prediction tasks with the attention mechanism, influences certain KT models. These models integrate attention by incorporating a self-attention mechanism into Deep Knowledge Tracing (DKT) [11], [12], [13]. The key advantage of self-attention-based models lies in their ability to predict individual interactions by considering the entire historical exercise sequence. This addresses concentration imbalances and enhances the holistic perspective during predictions.

Despite advancements in DNNs, current KT models still exhibit limitations which can be attributed to three factors. Firstly, some prior knowledge tracing models encode question sequence and history interaction sequences separately [11], [14], [12], [15]. These models initially encode the information of history interactions and the next question into hidden states separately. This late fusion method neglects shallow interactions and combines the two types of information only after they have been extracted into highly abstract states, resulting in a loss of bilateral information. Additionally, empirical findings reveal that, despite having more parameters, the self-attention-based AKT model fails to outperform the RNN-based DKT model on certain datasets [12]. AKT still adheres to the paradigm of late fusion, where the “Knowledge Retriever” models history interactions individually. Furthermore, the monotonic attention mechanism leaves history responses unattended, resulting in the failure to effectively integrate

Manuscript received xx xxxx xxxx; revised xx xxxx xxxx and xx xxx xxxx; accepted xx xxxx xxxx. This work was supported by the 2024 National Basic Subject Talent Training Plan 2.0 of China. (Corresponding author: Wenge Rong.)

H. Zhou, W. Rong, J. Zhang, Q. Sun, Y. Ouyang, and Z. Xiong are with the School of Computer Science and Engineering, Beihang University, China. (e-mail: h.zhou@buaa.edu.cn, w.rong@buaa.edu.cn, zhangjf@buaa.edu.cn, sunqing@buaa.edu.cn, oyyx@buaa.edu.cn, xiongz@buaa.edu.cn).

Digital Object Identifier 10.1109/TLT.xxxx.xxxxxxx

information from both questions and responses.

Secondly, current KT models aiming to merge information pertaining to both questions and skills, exhibit a shortfall in meticulous fusion treatment. Notably, some models opt for a simplistic concatenation or addition of embedding vectors without considering hierarchical distinction between questions and skills [14], [16]. It is important that questions and skills occupy disparate levels of abstraction and hierarchy within students' learning trajectories. Skills invariably hold a higher position than questions, as questions are derived from skills. Treating them as interchangeable overlooks the hierarchical disparity inherent in their relationship.

Lastly, current KT models fail to utilize datasets optimally. Each element within a given exercise sequence is used only once during training [4], [7], [11]. Dividing subsequences in a consecutive manner may overlook those with diverse distributions in the context of knowledge tracing. KT models learn how to effectively capture students' knowledge states via different subsequences of history interactions. Therefore, introducing overlaps within subsequences leads to enhanced model performance. Furthermore, many previous KT models merely utilize the information of questions or skills [4], [8], [12]. However, certain dataset components, including antecedent information known prior to responding and subsequent information discernible only after responding, remain untapped. These overlooked elements have utility in extracting students' knowledge statuses and predicting future interactions.

Considering the aforementioned limitations of current KT models, this paper introduces the Alternate Autoregressive Knowledge Tracing model (AAKT) as a viable solution. An illustrative example of AAKT is shown in Fig. 1.

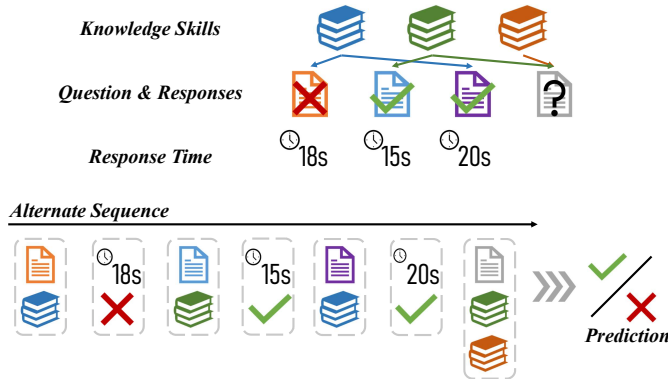


Fig. 1. An overview of AAKT. Every element in students' history interactions is split into anterior information and posterior information. Subsequently, they are organized in an alternate manner, facilitating autoregressive modeling.

The AAKT model adopts an innovative approach by implementing alternate sequence construction, wherein elements from each question sequence and its corresponding response sequence are interleaved. This technique is augmented by the incorporation of sliding windows, enhancing dataset volume and bolstering predictive performance. Integration of information related to questions and skills is accomplished through an auxiliary task coupled with the computation of auxiliary loss. Additionally, the embedding process incorporates supplementary details, such as the time spent on answering each question,

further enriching the model's capacity for comprehensive knowledge tracing.

In AAKT, a distinctive approach is employed to rearrange and combine each question sequence and its corresponding response sequence. This involves interleaving question indices and responses in an alternate order, deliberately segregating anterior and posterior information and thereby facilitating an autoregressive prediction process. The resulting rearranged sequences are partitioned into subsequences with overlaps using sliding windows. Then these subsequences are amalgamated with additional information, such as the time spent on answering questions, to generate embedding vectors. These vectors undergo an auxiliary task, ensuring their inclusion of hierarchical information pertaining to the corresponding skill(s).

The modified vectors are then fed into an autoregressive transformer, specifically the GPT-J (without pre-trained weight) instance in this paper. The output of the autoregressive transformer, representing the hidden knowledge status of students, is further transformed into prediction results via feed-forward networks. The total loss of the model is calculated as a combination of binary cross-entropy loss and Kullback-Leibler divergence generated from the auxiliary task.

The contribution of our paper can be summarized as follows:

1. **Autoregressive Perspective of Knowledge Tracing:** In this paper, we propose a novel perspective on knowledge tracing. We posit that knowledge tracing can be regarded as generative, aligning with the setting of autoregressive models. In the generative context, the model takes students' historical interactions as input and generates the most probable representation of the next step in the hidden state space. Subsequently, the representation can be extracted into the probability of correctly answering the next question.

2. **Alternate Sequence Construction:** Our work stands out as the first by introducing the innovative use of alternate sequences to integrate question sequences and response sequences. This novel methodology fosters autoregressive modeling, ensuring an automatic and effective interaction between these two critical types of information.

3. **Skill-Enriched Embeddings:** We introduce skill-related information into question embeddings through an auxiliary task. This approach acknowledges the hierarchical distinction between questions and skills within learning trajectory, showcasing superiority over conventional addition or concatenation strategies of embedding fusion.

II. RELATED WORK

Corbett and Anderson introduced Bayesian Knowledge Tracing (BKT) as the initial knowledge tracing model, utilizing binary variables to predict a student's mastery or non-mastery of skills within a set [1]. BKT relies on assumptions about variable distributions, including response accuracy and question difficulty [17], [18]. The model considers four factors (i.e., initial knowledge status, learning rate, guessing probability, and slipping probability) to predict knowledge status based on the most recent response. Subsequent research expanded BKT by introducing factors like problem difficulty or individual

learning ability [19], [20]. Additionally, Knowledge Tracing Machine (KTM) [21] employs factorization machines [22] to enhance the interpretability and performance of BKT. However, the interpretability of BKT models relies on simplified assumptions, limiting their ability to capture dynamic knowledge status and provide comprehensive explanations of students' learning processes, thereby negatively impacting overall model performance.

To address the limitations of traditional KT models, Piech et al. [4] pioneered the integration of deep neural networks with knowledge tracing, introducing the DKT model. DKT employed Long Short-Term Memory recurrent neural networks (LSTM) [23] to model students' knowledge status, surpassing the performance of BKT and its extensions. Seeking enhanced prediction consistency in exercise sequences, DKT+ [7] improved upon DKT by introducing regularization on the LSTM's hidden state, smoothing the evolution of knowledge status. Subsequent to DKT, researchers delved further into the application of neural networks in knowledge tracing. Zhang et al. [8] proposed a Dynamic Key-Value Memory Network (DKVMN) employing external memory matrices to store knowledge memory and update corresponding mastery levels. Building on DKVMN, Abdelrahman and Wang [9] introduced a modified LSTM, hop-LSTM, in their model. Guo et al. [16] utilized adversarial training in Adversarial Knowledge Tracing (ATKT) to enhance model robustness and mitigate overfitting in small datasets. Wang et al. [24] integrated educational priors and RNN in a KT model to improve interpretability.

Building on the success of attention mechanisms and transformers [10] in natural language processing and computer vision, self-attention-based knowledge tracing models have emerged. Pandey and Karypis [11] were pioneers in introducing the self-attention mechanism to Knowledge Tracing models (SAKT). This innovation enables SAKT to capture long-term dependencies within historical exercises. Pu et al. [15] proposed a KT model based on the vanilla transformer. Additionally, Ghosh et al. [12] presented a context-aware attentive KT model featuring monotonic attention and an exponential decay mechanism. More recently, Cui et al. [25] introduced a multi-relational transformer designed to facilitate fine-grained interaction modeling between question-response pairs.

Despite notable advancements, self-attention-based KT models encounter challenges in effectively integrating information from questions and responses. While various design features attempt to model both types of information, the interactive relationship between them is partly neglected. In SAKT [11], history interactions, including questions and responses, and future questions are encoded into different sequences, leading to inconsistencies in semantic spaces. AKT [12] calculates self-attention within question sequences and response sequences separately, employing a knowledge retriever to partially combine the two types of information. However, the information from responses does not contribute to calculating attention weights in knowledge retrievers, resulting in inadequate information interaction. MRT-KT [25] establishes multiple relations between different exercises based on their question skill sets and response results. However, it

omits including response information as encoded input due to the lack of an effective autoregressive sequence construction method.

In response to the identified limitations of self-attention-based KT models, we advocate for better integration of questions and responses. Subsequently, we conducted an analysis of the causal relationships inherent in knowledge tracing tasks. Our proposal introduces an alternative autoregressive modeling approach based on a novel sequential representation that encompasses both question and response information.

III. METHODOLOGY

In this section, we present the details of our AAKT framework. The overall architecture is depicted in Fig. 2. The initial segment encompasses **Alternate Sequence Construction**, a process that amalgamates the question sequence with the question and response sequences to form a distinctive sequence. This is succeeded by the **Sliding Window** mechanism, which selects subsequences from the alternate sequence for both training and prediction purposes. The second segment introduces **Additional Information** into the sequential embeddings, capturing inter-sequence dependencies through an autoregressive transformer. This component predicts responses to each question in the sequence, employing Binary Cross Entropy (BCE) loss calculated based on the prediction outcomes and response label sequences. The third and final segment conducts **Auxiliary Task** and focuses on compelling the question embeddings to assimilate skill information. This is achieved by predicting the skill distribution for each question and subsequently minimizing the Kullback–Leibler divergence with the actual skill distribution.

A. Problem Formulation

The notations needed to formulate the target of knowledge tracing task and the principle of AAKT are listed in Table I.

TABLE I
NOTATIONS USED FOR PROBLEM FORMULATION

Notation	Description
$\mathcal{Q} = \{q_1, q_2, \dots\}$	Question set
$\mathcal{T} = \{t_1, t_2, \dots\}$	Skill set
q_i	The i -th question of \mathcal{Q}
t_i	The i -th skill of \mathcal{T}
$\mathcal{I} = \{\text{Ex}_1, \text{Ex}_2, \dots\}$	The exercise sequence of a student
$\tilde{q}_i \in \mathcal{Q}$	The question in \mathcal{I} at step i
$\tilde{t}_i \in \mathcal{T}$	The skill set in \mathcal{I} at step i
$\tilde{a}_i \in \mathbb{R}$	The posterior information in \mathcal{I} at step i
$\tilde{c}_i \in \{0, 1\}$	The correctness in \mathcal{I} at step i
$\text{Ex}_j = (\tilde{q}_j, \tilde{t}_j, \tilde{a}_j, \tilde{c}_j) \in \mathcal{I}$	The element of \mathcal{I} at step j
L_{\max}	Max sequence length
$r_o \in [0, 1]$	Overlap ratio
L_{batch}	Batch size

Given the exercise sequence \mathcal{I} of a certain student, knowledge tracing seeks to monitor the student's evolving knowledge status by predicting the correctness of future questions. Therefore, suppose that the model has already known the data of $\mathcal{I}' = \{(\tilde{q}_1, \tilde{t}_1, \tilde{a}_1, \tilde{c}_1), \dots, (\tilde{q}_m, \tilde{t}_m, \tilde{a}_m, \tilde{c}_m)\} \in \mathcal{I}$, it should then predict the probability of answering the next question

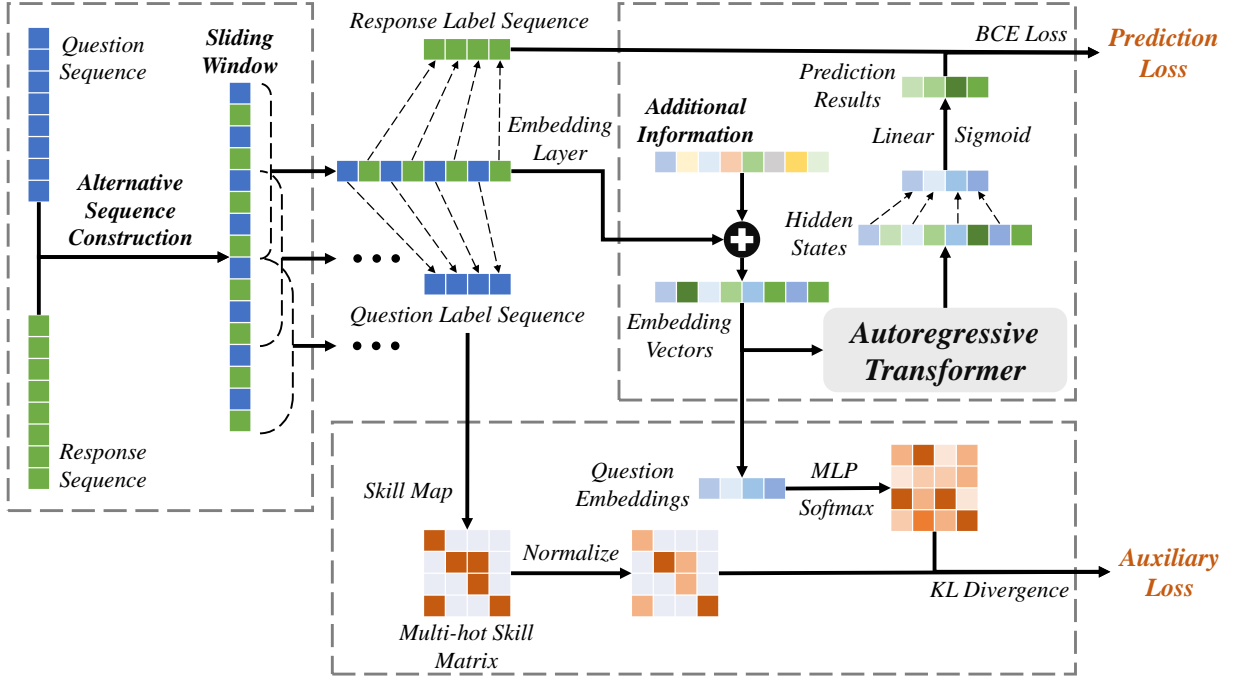


Fig. 2. The overall architecture of the proposed AAKT framework. It comprises three integral components.

correctly, namely $\Pr\{\tilde{c}_{m+1} = 1 \mid \mathcal{I}', \tilde{q}_{m+1}, \tilde{t}_{m+1}\}$ (\tilde{a}_{m+1} is posterior information so we do not know it before the question is answered).

B. Preliminaries

As presented in the first contribution in Section I, the generative perspective of knowledge tracing task aligns with the setting of autoregressive modeling. We initiate the detailed interpretation by introducing the concept of autoregressive models.

Mathematically, the autoregressive model characterizes a system in which its status (dependent variable) linearly depends on its past status. This system can be mathematically expressed through a stochastic difference equation, as illustrated below:

$$y_t = \beta_0 + \sum_{i=1}^p \beta_i y_{t-i} + \epsilon_t. \quad (1)$$

In this equation, β_i is a constant denoting how the status i steps ago influences current values. Typically, the expectation is for β_i to decrease as i increases, signifying that events from further in the past exert less impact on current occurrences. If anything beyond p steps ago holds no influence, the model is denoted as $AR(p)$. Here, ϵ_t represents a “noise” term accounting for random events affecting the system’s status. Actually, the principle of BKT [1] is consistent with $AR(1)$, where current values are merely influenced by the status one step ago. In BKT, a student’s knowledge status related to a specific skill is decided by the latest response and other fixed probabilities, such as the probability of making a mistake when applying a known skill.

In deep learning models, autoregressive models (AR models) are mainly used in natural language processing, such as GPT series [26], [27], [28]. Unlike original AR models, nonlinear transformation is introduced as a special extension (RNN, Transformer, etc.) to enable more flexible contextual interactions. AR models are unidirectional and can only see preceding information when processing specific sequences. Additionally, the coefficients β_i in Eq. 1 are not constant; instead, they dynamically depend on the model’s inputs. Through learning from previous steps and incorporating prior outputs as inputs, AR models recurrently predict the subsequent step. This adaptability positions AR models to excel in both prediction and generative tasks.

When processing sequential information, AR models predict the next step by analyzing previous words and estimating the probability distribution of the whole corpus provided. Formally, if the corpus \mathcal{C} and a sequence $\mathbf{x} = \{x_1, x_2, \dots, x_n\} (\forall i \in \{1, \dots, n\}, x_i \in \mathcal{C})$ is given, the target of AR models can be described as:

$$\begin{aligned} & \max_{\Theta} \mathcal{L}(\mathbf{x}) \\ &= \log \Pr(x_1 | \Theta) \Pr(x_2 | x_1; \Theta) \cdots \Pr(x_n | \mathbf{x}_{<n}; \Theta) \\ &= \sum_{i=1}^n \log \Pr(x_i | \mathbf{x}_{<i}; \Theta) \\ &= \sum_{i=1}^n \log \frac{\exp(\text{hidden}_{\Theta}(\mathbf{x}_{<i})^T \cdot \text{emb}(x_i))}{\sum_{x \in \mathcal{C}} \exp(\text{hidden}_{\Theta}(\mathbf{x}_{<i})^T \cdot \text{emb}(x))}, \end{aligned} \quad (2)$$

where $\text{hidden}_{\Theta}(\cdot)$ and $\text{emb}(\cdot)$ represent hidden states of model and embedding vector respectively.

Similarly, exercise sequences within knowledge tracing tasks embody causal relationships, where the response to the current question hinges on preceding questions and responses.

Consequently, we posit that autoregressive models can be effectively employed in knowledge tracing tasks.

C. Alternate Autoregressive Modeling

1) *Alternate Sequence Construction*: Some current KT models split the raw datasets into two kinds of sequences, namely history sequence and query sequence [11], [16], [29]. In accordance with the notations outlined in Table I, the exercise sequence $\mathcal{I} = \{\text{Ex}_1, \text{Ex}_2, \dots\}$ is divided and transferred into **History Sequence** and **Query Sequence**:

$$\begin{aligned} \mathcal{H}_i &= \{h_1, \dots, h_i\} \\ &= \{f_h(\tilde{q}_1, \tilde{t}_1, \tilde{c}_1), \dots, f_h(\tilde{q}_i, \tilde{t}_i, \tilde{c}_i)\}, \\ \mathcal{Q}_i &= \{\text{qry}_1, \dots, \text{qry}_i\} \\ &= \{f_{\text{qry}}(\tilde{q}_1, \tilde{t}_1), \dots, f_{\text{qry}}(\tilde{q}_i, \tilde{t}_i)\}. \end{aligned} \quad (3)$$

In Eq. 3, $f_h(\cdot)$ and $f_{\text{qry}}(\cdot)$ denote functions that embed arguments into a vector. history sequence takes every (question, skills, response) tuple in the history interactions, which is the full sequence of student's interactions in the learning process, and transform every tuple into a vector. Query sequence takes every (question, skills) tuple and does similar transformation. The differences between them lie in the choice of tuple and the transformations. These models learn from students' history exercise sequences and transfer them into hidden knowledge status of every step first. Subsequently, the next element in query sequence is combined with history sequence, generating a predictive result for the correctness of answering the next exercise:

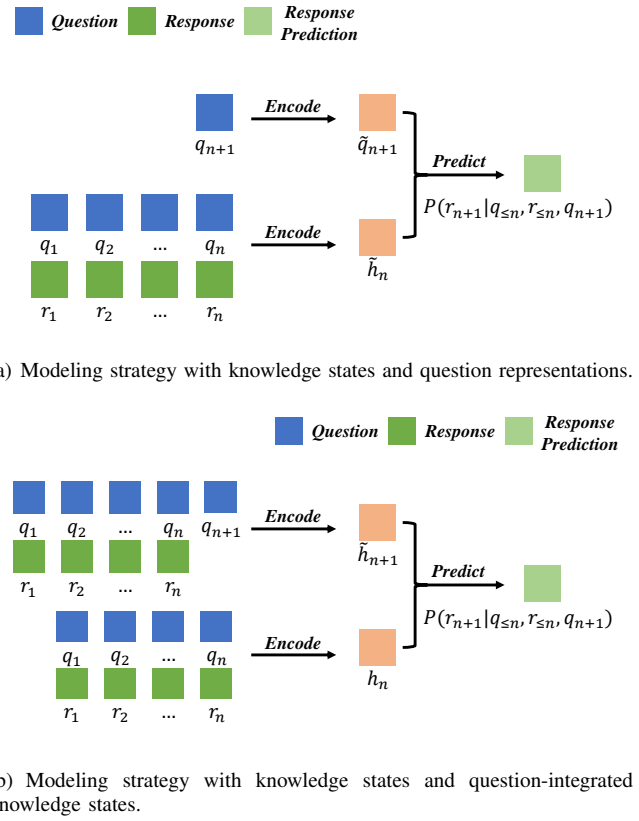
$$\hat{c}_{i+1} = \text{Predict}(\mathcal{H}_i, \text{qry}_{i+1}) \quad (4)$$

where $\text{Predict}(\cdot)$ denotes prediction layers of models. Following the guideline of Eq. 3 and Eq. 4, certain previous KT models developed two main kinds of modeling strategy (refer to Fig. 3(a) and Fig. 3(b)).

Knowledge tracing models like DKT [4], DKT+ [7], and DTransformer [13] adopt the approach illustrated in Fig. 3(a), explicitly representing students' status as \tilde{h}_n . However, this method requires two separate encoders and introduces a formal disparity between q_{n+1} and $(q_{\leq n}, r_{\leq n})$, which can detrimentally impact learning efficiency and generalization.

To address the above-mentioned issues, CL4KT [30] and LBKT [31] utilize the method demonstrated in Fig. 3(b), simultaneously considering knowledge states and question-integrated knowledge states. Here, the models capture students' knowledge states after completing n exercises and when encountering the $(n+1)$ th question. While this method mitigates generalization issues, it imposes computational overhead and introduces inconsistency by encoding the same information (i.e., $q_{\leq n}$ and $r_{\leq n}$) twice.

Considering information interaction and computational efficiency, we draw inspiration from autoregressive models and introduce a novel modification to the original exercise sequences (refer to Fig. 3(c)). The expected result of the previous two strategies is achieved by merely changing the representation of



(b) Modeling strategy with knowledge states and question-integrated knowledge states.

(c) Modeling strategy with uniform alternating states through alternating sequence.

Fig. 3. Different modeling strategy between previous KT models and AAKT.

students' exercise records. The input sequence is formulated as follows, with notations borrowed from Table I:

$$\mathcal{I}_i^{\text{in}} = \begin{cases} f_r(\tilde{q}_k, \tilde{t}_k) & , i = 2k - 1 \quad (1 \leq k \leq \frac{|\mathcal{I}|}{2}). \\ f_q(\tilde{c}_k, \tilde{a}_k) & , i = 2k \end{cases} \quad (5)$$

The embedding functions for questions ($f_q(\cdot)$) and responses ($f_r(\cdot)$) represent the transformation of information related to exercises and skills, and correctness and posterior information, respectively. As evident from Eq. 5, each element in the original sequences undergoes embedding only once. Consequently, the input sequence utilizes a unified embedding space, thereby enhancing information consistency. **A comprehensive elucidation of $f_q(\cdot)$ and $f_r(\cdot)$ is provided in Subsection III-D and Subsection III-E.**

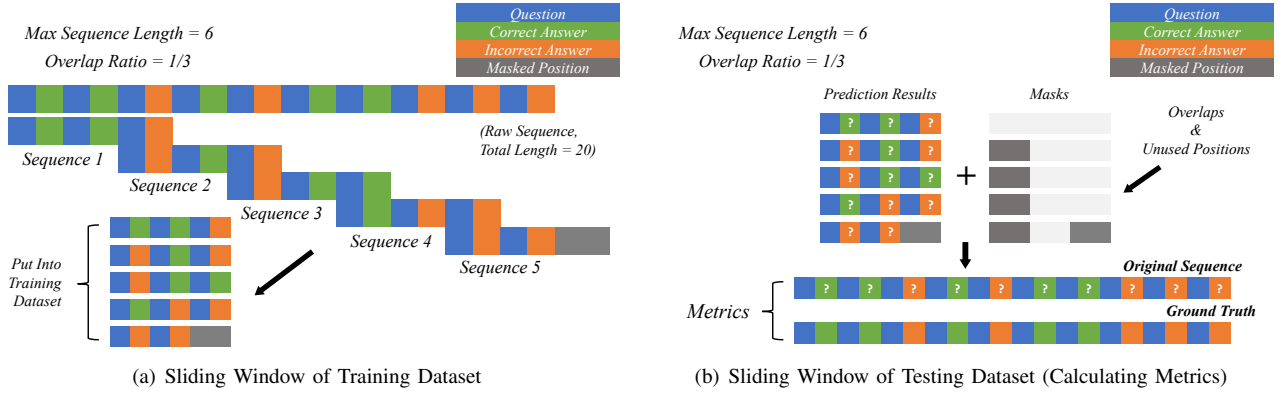


Fig. 4. Overall illustration of sliding window technique in our model. There are slight variations in the approach for the training dataset compared to the testing dataset.

According to the description of Fig. 3(c) and Eq. 5, our model only outputs the response prediction at step $2k - 1$. Consequently, the output sequence should be formulated as follows:

$$\begin{aligned} \mathcal{I}_i^{\text{out}} &= \hat{c}_i \\ &= \Pr(\tilde{c}_i = 1 | \mathcal{I}_{\leq 2i-1}^{\text{in}}). \end{aligned} \quad (6)$$

To elaborate further, a single step of hidden knowledge status (represented by a vector with multiple dimensions) is transformed into a 2D vector $\mathbf{v} = (v_{\text{correct}}, v_{\text{incorrect}})$ through a feed-forward network. Here, we interpret the first and second dimensions of \mathbf{v} as the 'raw probability' of predicting correctly and incorrectly, respectively. Then the probability is calculated as follows:

$$\begin{aligned} \Pr(\tilde{c}_k = 1 | \mathcal{I}_{\leq 2k-1}^{\text{in}}) &= \text{Sigmoid}(v_{\text{correct}} - v_{\text{incorrect}}) \\ &= \frac{1}{1 + \exp(v_{\text{incorrect}} - v_{\text{correct}})}. \end{aligned} \quad (7)$$

Finally, we use binary cross-entropy loss as the prediction loss:

$$\mathcal{L}_{\text{pred}} = -\frac{1}{|\mathcal{I}|} \sum_{i=1}^{|\mathcal{I}|} \hat{c}_i \cdot \log(\hat{c}_i) + (1 - \hat{c}_i) \cdot \log(1 - \hat{c}_i). \quad (8)$$

2) Sliding Window Technique: To optimize information utilization within the datasets, we employ the sliding window technique during preprocessing, permitting overlaps in both the training and testing datasets. However, owing to considerations of validity in the testing process and metric calculations, there are slight variations in the specific steps for training and testing. The sliding window technique is illustrated in Fig. 4. In this depiction, two key hyperparameters are set: **max sequence length** L_{max} , representing the maximum length in a data batch, and **overlap ratio** r_o , denoting the extent to which the next sliding window covers the current one (refer to Table I).

For the training dataset, an individual student's alternate exercise sequence (refer to Eq. 5) is partitioned into subsequences. The window initiates its movement from the commencement of the raw sequence and progresses towards the end. At each step, it moves forward by $L_{\text{max}} \cdot (1 - r_o)$ elements, where L_{max} and r_o represent the maximum sequence length

and overlap ratio, respectively. Additionally, the value of $L_{\text{max}} \cdot (1 - r_o)$ must be even, as a complete interaction involves two elements in the sequence (question and answer). If the range of the last window extends beyond the raw sequence, the remainder is masked, ensuring it is not considered when computing the loss. This technique effectively augments the dataset volume by introducing overlaps:

$$|\text{Dataset}'| \approx \frac{1}{1 - r_o} |\text{Dataset}|. \quad (9)$$

Handling the testing dataset and computing metrics involves some additional processes. In testing, all overlaps in the alternate sequence must be masked at corresponding positions (refer to Fig. 4(b)). This precaution is taken to prevent the same interaction from being counted multiple times during metric calculations, thereby avoiding inconsistency and inaccuracies. The efficacy of the sliding window technique in testing has been validated and shown to enhance prediction performance (refer to Section IV).

D. Auxiliary Task in Embedding

To effectively integrate information from both questions and related skills, we introduce an auxiliary task in our model. We operate under the assumption that questions and skills exhibit hierarchical differences and should be treated distinctively. Furthermore, we acknowledge that the information related to questions is more concrete than that of skills and should hold a predominant position in the model.

Specifically, given a question $q_i \in \mathcal{Q}$, we first transform it into a vector through an embedding matrix $W^q \in \mathbb{R}^{|\mathcal{Q}| \times \text{dim}}$:

$$\text{emb}(q_i) = W_i^q \in \mathbb{R}^{\text{dim}} \quad (10)$$

where dim denotes the dimension of embedding vectors, and W_i^q denotes the i^{th} row of W^q .

On that basis, we predict the skill distribution beside this question through a classifier for \mathcal{T} -class classification, which is composed of FFN followed by a softmax function:

$$\tilde{p}(t|q_i) = \text{Softmax}(\text{FNN}(\text{emb}(q_i))), t \in \mathcal{T} \quad (11)$$

Simultaneously, we transform \tilde{t}_i , the true skill set of q_i , into the skill distribution for supervision:

$$p(t|q_i) = \begin{cases} \frac{1}{|\tilde{t}_i|} & , t \in \tilde{t}_i \\ 0 & , t \notin \tilde{t}_i \end{cases} \quad (12)$$

Finally, we force the question representation $\text{emb}(q_i)$ to learn the information of corresponding skill set through minimizing the KL divergence between the predicted skill distribution $\tilde{p}(t|q_i)$ and the true skill distribution $p(t|q_i)$, which forms the auxiliary loss.

$$\mathcal{L}_{aux} = KL(p(t|q_i) \parallel \tilde{p}(t|q_i)) \quad (13)$$

The efficacy of the auxiliary task in facilitating the integration of skill-related information into the embedding vectors of questions lies in the backpropagation process, where the auxiliary loss is propagated back to the embedding weights of questions. To uphold the hierarchical nature of information, we establish a distinct pathway for skill-related information rather than adding it directly to the embedding vectors of questions. This approach represents both a conceptual and empirical improvement (**refer to Section IV**).

The auxiliary loss is regarded as one part of the total loss of the model. Therefore, to jointly learn all parameters in our model, the following loss is optimized:

$$\mathcal{L} = \mathcal{L}_{pred} + \mathcal{L}_{aux}. \quad (14)$$

E. Processing of Additional Information

In the context of online education platforms, interaction records encompass diverse attributes, including timestamps and student gender. Considering these attributes in knowledge tracing tasks can be advantageous, as relying solely on question indices and skill sets may inadequately capture the nuances of provided exercises. In addition to semantic information, where some knowledge tracing models leverage textual question descriptions [32], [33], interaction-related information can be categorized into four distinct groups, as illustrated in Fig. 5.

	Discrete	Continuous
Anterior	Related Skills Concrete Subject Student's Gender Difficulty Level (Labeled in Advance)	Timestamp of Starting
Posterior	Count of Hints/Attempts Used Student's Confidence Level	Timestamp of Finishing Total Time Spent Average Correct Rate

Fig. 5. Four categories of interaction-related information and examples.

In the AAKT framework, we introduce a viable approach to incorporate additional information, both anterior and posterior, into alternate sequences. As illustrated in Fig. 3, a comprehensive representation of a specific interaction involves two

elements in the input sequence. Essentially, anterior information should be combined with q_i , while posterior information should be combined with r_i at step i .

In our model, we have focused on utilizing one category: posterior continuous information, specifically the 'total time spent'. This choice is informed by the significance of time spent on exercises in the context of online education and the nature of the datasets. The total time spent values range from 0 ms to approximately 4×10^5 ms. To bring them into a standardized range (0 ms to 2×10^5 ms), we applied clipping. Additionally, a constant called **Time Factor** ($\tau = 60000$ ms) was introduced to partially normalize these values:

$$\text{time}_i^{norm} = \frac{\text{time}_i}{\tau}. \quad (15)$$

After normalization, these values become dimensionless and can be integrated into the model. More precisely, the normalized time values undergo multiplication by a trainable vector, denoted as \mathbf{v}_{time} , followed by addition to the embedding vectors of r_i (refer to Eq. 5):

$$\begin{aligned} \tilde{a}_i &= \text{time}_i^{norm}, \\ \mathbf{f}_q(\tilde{a}_i, \tilde{c}_i) &= \begin{cases} \tilde{a}_i \cdot \mathbf{v}_{time} + \text{emb}_{right} & , \tilde{c}_i = 1 \\ \tilde{a}_i \cdot \mathbf{v}_{time} + \text{emb}_{wrong} & , \tilde{c}_i = 0 \end{cases} \end{aligned} \quad (16)$$

Here, emb_{right} and emb_{wrong} are two trainable vectors.

IV. EXPERIMENTS

In this section, we conduct a series of experiments on real-world datasets to evaluate the performance of our model¹. Moreover, we conduct some ablation studies and visualizations to further validate the effect of the key components in AAKT architecture.

A. Datasets

Four real-world datasets with information of every total time spent on every question are used to evaluate the performance of AAKT. The statistics of four datasets are shown in Table II and Fig. 6.

1) *EdNet-KT1*: This dataset² is collected and introduced by [34]. Considering the excessively large number of records, we only choose EdNet-KT1 dataset which consists of students' practice history logs, and selected the records of 5,000 students. There are in total 188 skills and 12,276 questions. The largest number of concepts underlying one question is 6.

2) *ASSISTments2009*: This dataset is one of the most popular benchmark datasets in KT which is provided by the ASSISTments platform [35]. The version in our paper is conducted using the latest updated version "Skill-builder". There are 4,151 students, 16,891 questions, and 110 concepts. We obtain 274,042 observed responses in total. **Notably, questions with more than one skill are split into multiple questions in the original dataset, which is not proper. To address this issue, we combine them as one question in the experiments.**

¹The code is available at <https://github.com/alxzzhou/AAKT>

²This dataset is available at: <https://github.com/rriid/ednet>

TABLE II
DATASETS STATISTICS

Features/Datasets	EdNet-KT1	ASSISTments2009	ASSISTments2017	Junyi
#Students	5,000	4,151	1,709	247,605
#Questions	12,276	16,891	3,162	721
#Skills	188	110	99	40
#Records	2,208,254	274,042	942,540	25,925,983
#Avg. Questions per Skill	65.30	153.55	31.94	18.02
#Avg. Skills per Question	2.14	1.19	1.00	1.00
%Correct Answering Records	69.88%	66.15%	37.27%	82.79%
%Incorrect Answering Records	30.12%	33.85%	62.73%	17.21%
%Sequences within Length (0,10)	0.00%	55.36%	0.00%	49.56%
%Sequences within Length [10,100)	23.75%	30.76%	3.63%	33.50%
%Sequences within Length [100,1000)	53.03%	13.13%	82.67%	14.80%
%Sequences within Length [1000,10000)	22.06%	0.75%	13.70%	2.13%
%Sequences within Length [10000,+∞)	1.16%	0.00%	0.00%	0.01%

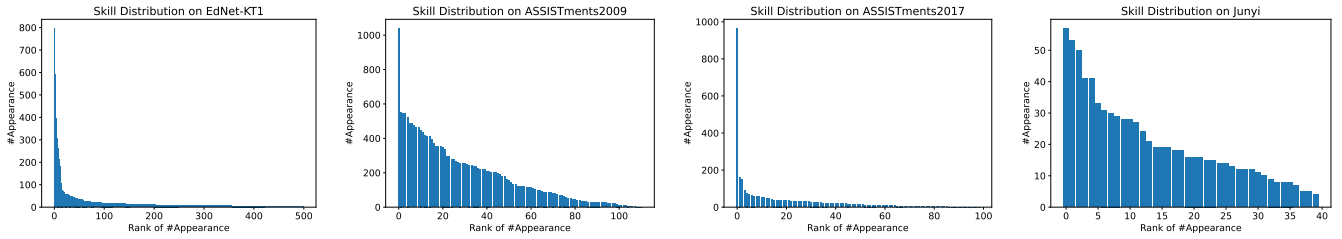


Fig. 6. Skill distribution on every dataset. For EdNet-KT1, we define that two questions are the same in terms of skill if and only if their skill sets are identical. Moreover, considering the extremely long and thin tail of skill distribution on EdNet-KT1, we truncate the list to 500 records.

3) *ASSISTments2017*: The dataset³ is obtained from the ASSISTments data mining competition 2017. We do the same preprocessing as ASSISTments2009.

4) *Junyi*: **JunyiAcademy**⁴ [36] collects data on JunyiAcademy platform from 2018/08 to 2019/09. After preprocessing, the dataset has a total of 721 exercises and 25,925,983 interactions, with 247,605 users.

B. Baselines and Evaluation Metrics

We compare AAKT with the following advanced KT models. Specifically, we classify the baseline into two categories according to the modeling strategy illustrated in Fig. 3.

Baselines using knowledge states and question representations:

- **DKT** [4] is the first knowledge tracing model that utilizes deep neural networks. It uses long-short term memory recurrent neural network (i.e., LSTM) to generate sequential prediction of students' performance. The hidden state of the LSTM can be extracted to infer the student's mastery level of different skills.
- **DKT+** [7] is an improvement of DKT. It adds a regularization term on the hidden states of students, smoothing the evolving of knowledge status.
- **DKVMN** [8] extends DKT by introducing extra memory-augmented neural networks. Two matrices (namely key matrix and value matrix) are used to better trace students'

hidden knowledge status. Moreover, DKVMN utilizes the 'read' and 'write' processes to emulate the process of learning.

- **SAKT** [11] is the first KT model using self-attention mechanism. Its overall structure resembles that of the transformer.
- **AKT** [12] adopts a novel monotonic attention mechanism to model the influence of interaction from the distant past on students' knowledge status. It also introduces the difficulty coefficient when encoding skills.
- **KQN** [29] introduces a novel concept called probabilistic skill similarity that relates the pairwise cosine and Euclidean distances between skill vectors to the odds ratios of the corresponding skills.
- **ATKT** [16] adopts adversarial training with attentive LSTM to model students' knowledge status. The training process of the model contains perturbation to improve robustness.
- **CL4KT** [30] is a contrastive learning framework for knowledge tracing that reveals semantically similar or dissimilar examples of learning history and stimulates to learn their relationships is introduced.
- **DTransformer** [13] proposes a new architecture, Diagnostic Transformer (DTransformer), along with a new training paradigm, to tackle the challenge of knowledge state tracing, and proposes a novel training algorithm based on contrastive learning that focuses on maintaining the stability of the knowledge state diagnosis.

Baselines using knowledge states and question-integrated knowledge states:

³This dataset is available at: <https://sites.google.com/view/assistmentsdatamining/data-mining-competition-2017>

⁴<https://www.junyiacademy.org/>

- **CoKT** [37] exploits the inter-student information in knowledge tracing and retrieves the sequences of peer students who have similar question-answering experiences to obtain the inter-student information, and integrates the intra-student information to trace students' knowledge states and predict their correctness in answering questions.
- **LBKT** [31] combines the forgetting factor with learners' knowledge acquisition to comprehensively update their changing knowledge states in learning and generates better performance prediction for learners against existing methods.

In this paper, one of the metrics used to evaluate the performance of models is **Area Under ROC Curve (AUC)** which is widely used in knowledge tracing tasks [4], [8], [12]. A higher value of AUC indicates better performance in predicting students' responses, and the value of 0.5 represents the performance of making predictions by random guessing. Moreover, **Accuracy (ACC)** and **Root Mean Square Error (RMSE)** are included where higher ACC and lower RMSE indicate better performance.

C. Implementation

1) *Details of Autoregressive Transformer:* In our model, we utilize GPT-J [38] instance (without pre-trained weights) as the autoregressive transformer. The design of GPT-J closely follows GPT-3 Curie [28]. However, two minor architectural improvements have been made:

- Rotary Positional Embedding (RoPE) [39] is adopted for better performance.
- The attention layer and the feedforward layer are placed in parallel for decreased communication.

2) *Other Details:* We perform 5-fold cross-validation for every combination of models and datasets. We use 80% of student sequences for training and the rest 20% of student sequences for model evaluation.

Our AAKT model is implemented with PyTorch and we use Adam optimizer to train our model, setting the learning rate to 0.001 for all datasets. All experiments are conducted on a server with one NVIDIA A5000 (24GB) graphic card. Balancing time cost and performance, we set the overlap ratio r_o to 0.5 both in training and testing. Moreover, the implementations of baselines are based on open-source code provided by corresponding authors and pyKT [40], a comprehensive python-based benchmark platform for knowledge tracing tasks.

Details of other hyperparameters are listed below. For hyperparameters that are not fixed, we utilized grid searching to find the optimal choice.

- Batch size is chosen from {32, 64, 128, 256}.
- Dimension of embedding is chosen from {64, 96, 128, 256}.
- Max sequence length is chosen from {20, 50, 100, 300, 500, 1000, 2000} according to the distribution of sequences in each dataset.

- The number of heads in multi-head attention is set to 8. This configuration is borrowed from vanilla Transformer [10].
- The number of GPT-J blocks is chosen from {2,3,4}.
- The dimension of rotary position embedding is set to $\frac{d_{emb}}{2n_{head}}$ where d_{emb} and n_{head} denotes embedding dimension and number of heads in multi-head attention respectively. This configuration is borrowed from the GitHub repository **ChatGLM-Finetuning**⁵.

D. Results and Discussions

In this section, we list the average results of cross-validation in Table III (**— means that the model does not fit the dataset**), where the highest results in every column are displayed in bold and the second highest results are underlined. From the data shown in Table III, the following observations regarding baselines and AAKT can be made:

- AAKT outperforms the state-of-the-art models on four datasets (i.e., EdNet-KT1, ASSISTments2009, ASSISTments2017, and Junyi) and three metrics, validating its effectiveness in knowledge tracing tasks.
- DKT and DKT+ have relatively high performances on EdNet-KT1 and ASSISTments2017, outperforming models with more complicated structures like DKVMN and SAKT. DKT and DKT+ are simple models that merely utilize the LSTM structure. This implies that more skillful techniques should be explored in modeling the learning sequences of students for knowledge tracing and complexity does not necessarily lead to accuracy.
- Transformer-based models in baselines (i.e., AKT and DTransformer) perform well on four datasets. Notably, DTransformer achieves the second highest metric on EdNet-KT1, ASSISTments2009, and ASSISTments2017. The utilization of the diagnostic transformer enables DTransformer to outperform SAKT and AKT which use simpler attention techniques and training paradigms.
- Contrastive learning is applicable in knowledge tracing and demonstrates great potential. Models with contrastive learning (i.e., CL4KT and DTransformer) have relatively high performance compared to all chosen baselines.

Moreover, to further investigate the inter-metric relationship in Table III, we calculate the correlation coefficient of every metric pair on every dataset (refer to Table V). These relationships can guide researchers and practitioners in selecting the most appropriate metrics for assessing model performance, considering the trade-offs between ACC, AUC, and RMSE based on the specific goals of educational knowledge tracing tasks. The following observations regarding different metrics can be made:

- The correlation coefficients between Accuracy (ACC) and Area Under the Curve (AUC) consistently demonstrate a strong positive linear relationship across all datasets, ranging from 0.80560 to 0.98965. This suggests that models achieving higher accuracy also tend to exhibit

⁵https://github.com/liucong/ChatGLM-Finetuning/blob/a137ad63ee5c7d4e2c020e6f7f12d1bf71320bd1/glm3/modeling_chatglm.py

TABLE III
THE PERFORMANCE OF AAKT AND ALL BASELINES ON 4 DATASETS

	EdNet-KT1			ASSISTments2009			ASSISTments2017			Junyi		
	AUC \uparrow	ACC \uparrow	RMSE \downarrow	AUC \uparrow	ACC \uparrow	RMSE \downarrow	AUC \uparrow	ACC \uparrow	RMSE \downarrow	AUC \uparrow	ACC \uparrow	RMSE \downarrow
DKT	0.7609	0.7410	0.4161	0.6785	0.6839	0.4541	0.7615	0.7058	0.4349	0.8029	0.8543	0.3380
DKT+	0.7640	0.7435	0.4140	0.6829	0.6823	0.4548	0.7758	0.7092	0.4331	0.8025	0.8590	0.3305
DKVMN	0.7589	0.7380	0.4212	0.6471	0.6670	0.4722	0.7317	0.6895	0.4441	0.8003	0.8522	0.3338
SAKT	0.7640	0.7426	0.4141	0.7103	0.6633	0.4746	0.7325	0.6935	0.4468	0.7996	0.8529	0.3327
AKT	0.7733	0.7498	0.4124	0.7233	0.7170	0.4369	0.7711	0.7181	0.4282	0.8057	0.8578	0.3313
KQN	-	-	-	-	-	-	0.7067	0.6801	0.4516	0.7560	0.8465	0.3446
ATKT	-	-	-	-	-	-	0.7214	0.6932	0.4431	0.7597	0.8499	0.3418
CL4KT	0.7741	0.7489	0.4103	0.7152	0.7142	0.4371	0.7786	0.7239	0.4247	0.8070	0.8561	0.3308
DTransformer	0.7797	0.7539	0.4076	0.7303	0.7193	0.4345	0.7922	0.7320	0.4214	0.8104	0.8582	0.3285
CoKT	0.7771	0.7507	0.4091	0.7246	0.7161	0.4365	0.7840	0.7271	0.4241	0.8098	0.8570	0.3298
LBKT	-	-	-	0.7285	0.7191	0.4348	0.7883	0.7302	0.4239	0.8126	0.8583	0.3311
AAKT (Ours)	0.7827	0.7554	0.4064	0.7357	0.7223	0.4340	0.8018	0.7361	0.4198	0.8146	0.8603	0.3281

TABLE IV
THE PERFORMANCE OF AAKT AND ALL BASELINES ON FILTERED ASSISTMENTS2009 AND JUNYI

	ASSISTments2009 (Filtered)			Junyi (Filtered)		
	AUC \uparrow	ACC \uparrow	RMSE \downarrow	AUC \uparrow	ACC \uparrow	RMSE \downarrow
DKT	0.6830	0.6755	0.4579	0.8030	0.8476	0.3311
DKT+	0.6861	0.6787	0.4570	0.8021	0.8524	0.3309
DKVMN	0.6499	0.6578	0.4788	0.7937	0.8499	0.3340
SAKT	0.6627	0.6601	0.4750	0.8020	0.8531	0.3326
AKT	0.7189	0.7087	0.4391	0.8087	0.8548	0.3347
KQN	-	-	-	0.7606	0.8422	0.3421
ATKT	-	-	-	0.7650	0.8423	0.3438
CL4KT	0.7181	0.7164	0.4380	0.8024	0.8532	0.3319
DTransformer	0.7338	0.7185	0.4340	0.8138	0.8617	0.3258
CoKT	0.7269	0.7190	0.4345	0.8104	0.8590	0.3291
LBKT	0.7322	0.7210	0.4342	0.8144	0.8620	0.3270
AAKT (Ours)	0.7370	0.7266	0.4338	0.8179	0.8644	0.3237

TABLE V
THE CORRELATION COEFFICIENTS BETWEEN DIFFERENT METRICS

	EdNet-KT1	ASSISTments2009	ASSISTments2017	Junyi
ACC and AUC	0.98965	0.80560	0.97162	0.88353
AUC and RMSE	-0.95895	-0.98810	-0.99085	-0.91221
ACC and RMSE	-0.93551	-0.76554	-0.97392	-0.92088

higher AUC values. Additionally, the other two pairs of metrics show a negative correlation.

- The correlation coefficients demonstrate variations across diverse datasets. Specifically, the correlation coefficient between ACC and AUC on ASSISTments2009 is relatively lower compared to other datasets. This characteristic is also observed in the relationship between ACC and RMSE.

As depicted in Table II, approximately half of the sequences in ASSISTments2009 and Junyi are shorter than 10 steps. Additionally, around 20% of students in ASSISTments2009 responded to only one question. The prevalence of extremely short sequences in these datasets introduces a potential bias in model predictions, particularly for models sensitive to shorter sequences. To mitigate this influence, sequences shorter than 10 steps in ASSISTments2009 and Junyi are excluded, and the performances of models are reassessed (refer to Table IV). This operation is also seen in EKT [41]. These processed datasets are marked as “**Filtered**”.

Generally, the performances on filtered ASSISTments2009 and Junyi show some fluctuation. Our model ranks first on these two datasets.

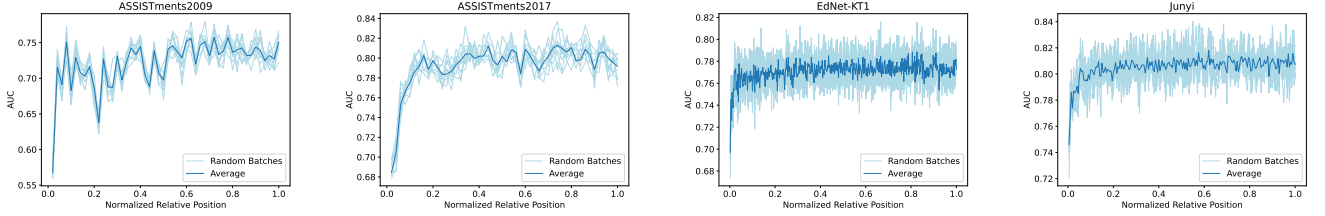
E. Ablation Study

To further investigate the contribution of key components in AAKT, we conducted a series of ablation studies. Variants of AAKT with specific components removed are defined. Results are listed in Table VI.

- **AAKT w/o Sliding** removes sliding window technique. That is to say, the r_o is set to zero both in training and testing.
- **AAKT w/o Skill** removes the information related to skills.
- **AAKT w/o Time** removes the attribute ‘total time spent’ for every question in the dataset.
- **AAKT w/o Time&Skill** removes both ‘total time spent’ attributes and the information related to skills.

TABLE VI
THE PERFORMANCE OF ORIGINAL AAKT MODEL AND ITS VARIANTS

	EdNet-KT1			ASSISTments2009			ASSISTments2017			Junyi		
	AUC \uparrow	ACC \uparrow	RMSE \downarrow	AUC \uparrow	ACC \uparrow	RMSE \downarrow	AUC \uparrow	ACC \uparrow	RMSE \downarrow	AUC \uparrow	ACC \uparrow	RMSE \downarrow
AAKT	0.7827	0.7554	0.4064	0.7357	0.7223	0.4340	0.8018	0.7361	0.4198	0.8146	0.8603	0.3281
w/o Sliding	0.7693	0.7500	0.4081	0.7302	0.7141	0.4385	0.7827	0.7241	0.4288	0.7998	0.8531	0.3305
w/o Skill	0.7713	0.7515	0.4076	0.7309	0.7160	0.4378	0.7974	0.7310	0.4260	0.8086	0.8563	0.3293
w/o Time	0.7726	0.7531	0.4072	0.7315	0.7181	0.4372	0.7846	0.7287	0.4248	0.8094	0.8568	0.3291
w/o Time&Skill	0.7690	0.7485	0.4083	0.7299	0.7139	0.4381	0.7819	0.7254	0.4281	0.8046	0.8550	0.3301
w/o Auxiliary Task	0.7679	0.7477	0.4090	0.7282	0.7134	0.4410	0.7831	0.7267	0.4305	0.8029	0.8502	0.3315



(a) AUC Distribution of Random Batches on ASSISTment2009 (b) AUC Distribution of Random Batches on ASSISTment2017 (c) AUC Distribution of Random Batches on EdNet-KT1 (d) AUC Distribution of Random Batches on Junyi

Fig. 7. AUC distribution on random batches in different datasets. The x-axis denotes the relative position of each question in the sequence. Light blue lines are results of 10 randomly chosen batches in each figure and the dark blue line is their average.

- **AAKT w/o Auxiliary Task** removes the auxiliary task. Specifically, it directly adds the embedding vectors of questions and skills together. For questions with more than one skill, we take the average of skill embedding vectors.

From Table VI, the following observations can be made:

- Removal of any key component (i.e., Sliding Window Technique, Additional Information, and Auxiliary Task) results in a decline in metrics across all four datasets. This underscores the validated effectiveness of these key components in our model.
- AAKT without the Sliding Window exhibits inferior performance across all datasets, affirming the positive impact of the Sliding Window technique.
- AAKT without Time & Skill performs less optimally than both AAKT without Time and AAKT without Skill. This observation suggests that incorporating more additional information contributes to improved model performance.
- Across all datasets, AAKT without Skill outperforms AAKT without the Auxiliary Task, highlighting the negative influence of an inadequate embedding strategy. The assumption of equal importance between questions and skills in embedding proves less effective within the AAKT framework.

F. Visualization

In this subsection, we conduct some visualizations to get deeper insights into some key components of AAKT.

1) *Case Study*: In this part, a simple case study is conducted to reinforce the validation of alternate sequence construction and autoregressive modeling. We select a specific alternate sequence from EdNet-KT1 where our model predicted the first 8 answers correctly. Moreover, the attention weights

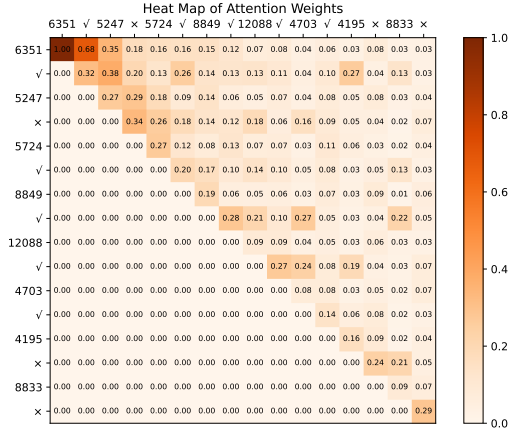


Fig. 8. Heat map of attention weights of first causal self-attention layer in our model. The x-axis and y-axis denote the same interaction subsequence from a certain sequence in EdNet-KT1 and the elements denote the specific attention weights (with softmax). Our model predicts all the answers to this subsequence correctly. Therefore, attention weights can demonstrate the preference of our model on the history of interactions when predicting.

of the first attention layer in our model are extracted when it is calculating the prediction results. A heat map is plotted according to the attention weights (refer to Fig. 8).

Our model has different preferences on history interactions when predicting the answer to the next question. Specifically, it emphasizes the result of answering question 5247 when predicting the answer of 5724. Similar emphasis can be observed from the prediction of 12088, 4703, 4195, and 8833. Our model utilizes the information of history responses to predict the answer to the current question, successfully and efficiently combining the information of questions and responses. This result is credited to alternative autoregressive modeling of original sequences.

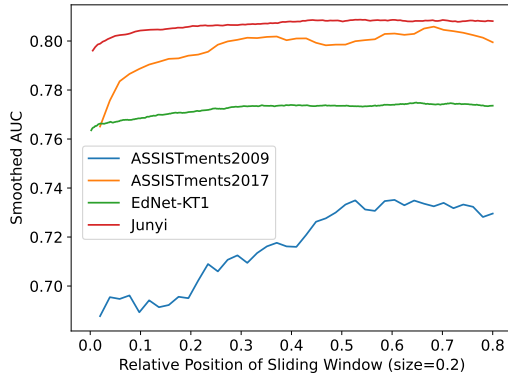


Fig. 9. Smoothed AUC results in Fig. 7 using a sliding window with the length of $0.2 \times \max_sequence_length$. The x-axis denotes the normalized relative position of the starting point of the sliding window.

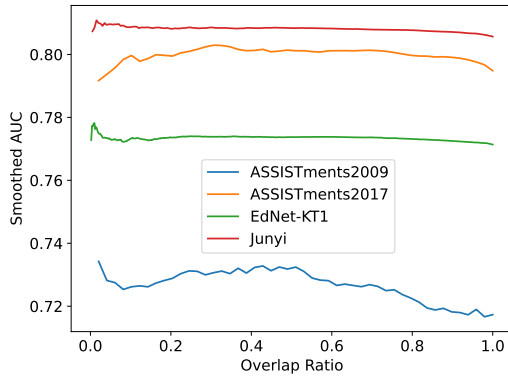


Fig. 10. The relationship between overlap ratio and the mean AUC score in the corresponding sliding window.

2) *The Effectiveness of Sliding Window*: To validate the effectiveness of the sliding window technique, it is imperative to examine alternate sequences horizontally. A meticulous analysis of the model's performance in predicting responses at specific positions is warranted.

Firstly, we demonstrate the overall AUC distribution on different positions and different datasets. We set the value of r_o to 0.0 and choose 10 random batches from the testing process on every dataset and calculate AUC in terms of the position in the sequence. These batches with shape (L_{batch}, L_{max}) are transposed to (L_{max}, L_{batch}) . Then for every position $1 \leq i \leq L_{max}$, there is a 1D array recording every prediction in this batch. After removing the masked and useless positions, AUC is calculated on these 1D arrays. Fig. 7 shows the visualizations of this experiment (the settings of L_{max} may be different on each dataset). The AUC of starting 10% prediction results are relatively lower, then increasing quickly. Afterward, the AUC fluctuates randomly and dramatically around a fixed value.

To make an in-depth analysis of Fig. 7, we smoothed all these results using a sliding window with the length of $0.2 \times L_{max}$. In Fig. 9, the outcomes of the smoothing process are presented, revealing discernible patterns amidst the randomness. Generally, a noticeable upward trend in mean

AUC scores is evident as the sliding window progresses through randomly selected batches from EdNet-KT1, Junyi, and ASSISTments2017. Although ASSISTments2009 exhibits more fluctuations, the overall trend persists.

Following the explanation provided in Subsection III-C2, as r_o decreases from 1.0, the prediction window expands from the last element of each sequence. Fig. 10 illustrates the relationship between r_o and the mean AUC in the corresponding prediction window, using the same randomly selected batches of data described above. The line representing ASSISTments2009 exhibits a small peak around 0.5, while others remain relatively stable. Given the information presented in Fig. 10, the fixed value of 0.5 for r_o in our model is demonstrated to be a reasonable and close-to-optimal choice.

3) *The Effectiveness of Auxiliary Task*: To validate whether the auxiliary task effectively incorporates skill-related information into the embedding vectors of questions, we perform an experiment to visualize the overall distribution on a 2D plane using UMAP [42], a proficient dimension reduction algorithm. The outcomes, depicted in Fig. 11, clearly indicate the successful accomplishment of the auxiliary task, as it distinctly segregates embedding vectors of questions based on their respective skill(s) in UMAP maps.

Detailed analysis of UMAP maps on every dataset are listed as follows:

- **EdNet-KT1**: The majority of embedding vectors are correctly categorized. However, there is a cluster that contains various types of points. This phenomenon can be ascribed to the notably skewed distribution of skills across the question set, as illustrated in Fig. 6. The distribution in EdNet-KT1 presents an elongated and slender tail, with certain skill sets appearing only once in the comprehensive question set.
- **ASSISTments2009**: The majority of embedding vectors are correctly categorized.
- **ASSISTments2017**: The majority of embedding vectors are correctly categorized.
- **Junyi**: The majority of embedding vectors are correctly categorized.

V. CONCLUSION

In this paper, we propose AAKT, a novel knowledge tracing model based on the autoregressive transformer. Different from previous knowledge tracing models with transformers which separate question and response, we combine question and response sequences, constructing alternate sequences as input. Particularly, the GPT-J instance (without pre-trained weights) is utilized as an autoregressive transformer backbone to extract the hidden knowledge status of students. We also introduce the sliding window technique to allow valid overlaps between batches both in training and evaluation, maximizing the utility of limited datasets and enhancing overall performance. To emphasize the hierarchical relation between questions and skills, we introduce an auxiliary task to better combine them. Moreover, we analyze a paradigm of handling additional information in the setting of our model and select 'total time spent' as additional information. Comparative experiments involving

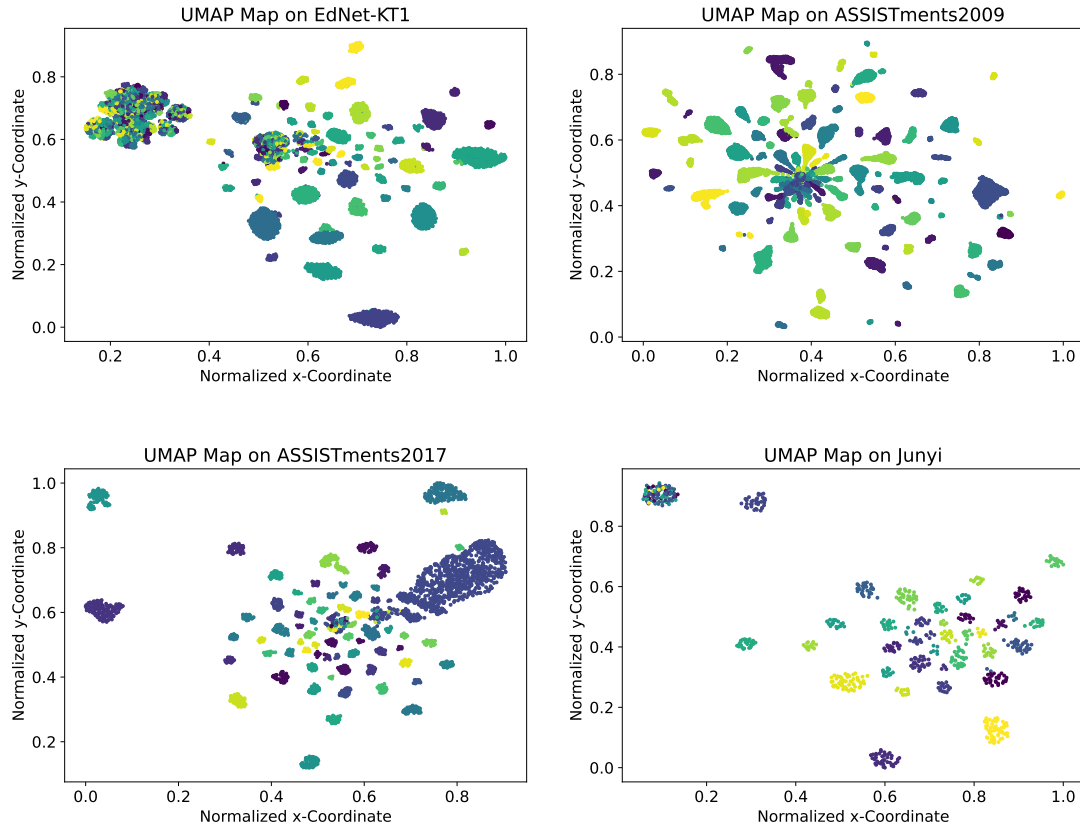


Fig. 11. UMAP mappings of embedding vectors processed by the auxiliary task. Distinct colors within each subfigure signify different skills or skill sets. It is important to note that distances between clusters hold no explicit meaning.

11 previous knowledge tracing models on four real-world datasets affirm the efficacy of AAKT in knowledge tracing tasks. Additionally, we demonstrate the contributory impact of key components of AAKT on its overall performance. Lastly, visualization experiments are conducted to elucidate the effectiveness of specific components, including alternate sequence construction, sliding window implementation, and auxiliary task integration.

Possible directions of future work include 1) designing more advanced methodologies for combining information from questions and skills, 2) exploring more efficient autoregressive transformer structures tailored for knowledge tracing tasks, and 3) devising more efficient approaches for utilizing limited datasets in training and testing phases.

REFERENCES

- [1] A. T. Corbett and J. R. Anderson, "Knowledge tracing: Modeling the acquisition of procedural knowledge," *User Modeling and User-Adapted Interaction*, vol. 4, no. 4, pp. 253–278, 1994.
- [2] M. Yudelson, K. R. Koedinger, and G. J. Gordon, "Individualized Bayesian knowledge tracing models," in *Proceedings of the 16th International Conference on Artificial Intelligence in Education*, 2013, pp. 171–180.
- [3] R. Pelánek, "Bayesian knowledge tracing, logistic models, and beyond: An overview of learner modeling techniques," *User Modeling and User-Adapted Interaction*, vol. 27, no. 3–5, pp. 313–350, 2017.
- [4] C. Piech, J. Bassen, J. Huang, S. Ganguli, M. Sahami, L. J. Guibas, and J. Sohl-Dickstein, "Deep knowledge tracing," in *Proceedings of the 29th Annual Conference on Neural Information Processing Systems*, 2015, pp. 505–513.
- [5] G. Abdelrahman, Q. Wang, and B. P. Nunes, "Knowledge tracing: A survey," *ACM Computing Surveys*, vol. 55, no. 11, pp. 224:1–224:37, 2023.
- [6] X. Song, J. Li, T. Cai, S. Yang, T. Yang, and C. Liu, "A survey on deep learning based knowledge tracing," *Knowledge-Based Systems*, vol. 258, p. 110036, 2022.
- [7] C. Yeung and D. Yeung, "Addressing two problems in deep knowledge tracing via prediction-consistent regularization," in *Proceedings of the 5th Annual ACM Conference on Learning at Scale*, 2018, pp. 5:1–5:10.
- [8] J. Zhang, X. Shi, I. King, and D. Yeung, "Dynamic key-value memory networks for knowledge tracing," in *Proceedings of the 26th International Conference on World Wide Web*, 2017, pp. 765–774.
- [9] G. Abdelrahman and Q. Wang, "Knowledge tracing with sequential key-value memory networks," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2019, pp. 175–184.
- [10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st Annual Conference on Neural Information Processing Systems*, 2017, pp. 5998–6008.
- [11] S. Pandey and G. Karypis, "A self attentive model for knowledge tracing," in *Proceedings of the 12th International Conference on Educational Data Mining*, 2019.
- [12] A. Ghosh, N. T. Heffernan, and A. S. Lan, "Context-aware attentive knowledge tracing," in *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2020, pp. 2330–2339.
- [13] Y. Yin, L. Dai, Z. Huang, S. Shen, F. Wang, Q. Liu, E. Chen, and X. Li, "Tracing knowledge instead of patterns: Stable knowledge tracing with diagnostic transformer," in *Proceedings of the ACM Web Conference 2023*, 2023, pp. 855–864.
- [14] D. Shin, Y. Shim, H. Yu, S. Lee, B. Kim, and Y. Choi, "SAINT+: integrating temporal features for ednet correctness prediction," in *Proceedings of the 11th International Learning Analytics and Knowledge Conference*, 2021, pp. 490–496.

- [15] S. Pu, M. Yudelson, L. Ou, and Y. Huang, "Deep knowledge tracing with transformers," in *Proceedings of the 21st International Conference on Artificial Intelligence in Education*, 2020, pp. 252–256.
- [16] X. Guo, Z. Huang, J. Gao, M. Shang, M. Shu, and J. Sun, "Enhancing knowledge tracing via adversarial training," in *Proceedings of the 29th ACM Multimedia Conference*, 2021, pp. 367–375.
- [17] R. S. J. de Baker, A. T. Corbett, and V. Aleven, "More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing," in *Proceedings of the 9th International Conference on Intelligent Tutoring Systems*, 2008, pp. 406–415.
- [18] Z. A. Pardos and N. T. Heffernan, "KT-IDEM: Introducing item difficulty to the knowledge tracing model," in *Proceedings of the 19th International Conference on User Modeling, Adaption and Personalization*, 2011, pp. 243–254.
- [19] M. Khajaj, R. V. Lindsey, and M. Mozer, "How deep is knowledge tracing?" in *Proceedings of the 9th International Conference on Educational Data Mining*, 2016.
- [20] S. Shen, Z. Huang, Q. Liu, Y. Su, S. Wang, and E. Chen, "Assessing student's dynamic knowledge state by exploring the question difficulty effect," in *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2022, pp. 427–437.
- [21] J. Vie and H. Kashima, "Knowledge tracing machines: Factorization machines for knowledge tracing," in *Proceedings of the 33rd AAAI Conference on Artificial Intelligence, the 31st Innovative Applications of Artificial Intelligence Conference, the 9th AAAI Symposium on Educational Advances in Artificial Intelligence*, 2019, pp. 750–757.
- [22] S. Rendle, "Factorization machines," in *Proceedings of the 10th IEEE International Conference on Data Mining*, 2010, pp. 995–1000.
- [23] X. Shi, Z. Chen, H. Wang, D. Yeung, W. Wong, and W. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proceedings of the 29th Annual Conference on Neural Information Processing Systems*, 2015, pp. 802–810.
- [24] F. Wang, Z. Huang, Q. Liu, E. Chen, Y. Yin, J. Ma, and S. Wang, "Dynamic cognitive diagnosis: An educational priors-enhanced deep knowledge tracing perspective," *IEEE Transactions on Learning Technologies*, vol. 16, no. 3, pp. 306–323, 2023.
- [25] J. Cui, Z. Chen, A. Zhou, J. Wang, and W. Zhang, "Fine-grained interaction modeling with multi-relational transformer for knowledge tracing," *ACM Transactions on Information Systems*, vol. 41, no. 4, pp. 104:1–104:26, 2023.
- [26] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," <https://www.mikecaptain.com/resources/pdf/GPT-1.pdf>, 2018.
- [27] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.
- [28] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," in *Proceedings of the 34th Annual Conference on Neural Information Processing Systems*, 2020.
- [29] J. Lee and D. Yeung, "Knowledge query network for knowledge tracing: How knowledge interacts with skills," in *Proceedings of the 9th International Conference on Learning Analytics and Knowledge*, 2019, pp. 491–500.
- [30] W. Lee, J. Chun, Y. Lee, K. Park, and S. Park, "Contrastive learning for knowledge tracing," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 2330–2338.
- [31] B. Xu, Z. Huang, J. Liu, S. Shen, Q. Liu, E. Chen, J. Wu, and S. Wang, "Learning behavior-oriented knowledge tracing," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023, pp. 2789–2800.
- [32] S. Sonkar, A. S. Lan, A. E. Waters, P. Grimaldi, and R. G. Baraniuk, "qDKT: Question-centric deep knowledge tracing," in *Proceedings of the 13th International Conference on Educational Data Mining*, 2020.
- [33] W. Wang, H. Ma, Y. Zhao, F. Yang, and L. Chang, "SEEP: Semantic-enhanced question embeddings pre-training for improving knowledge tracing," *Information Sciences*, vol. 614, pp. 153–169, 2022.
- [34] Y. Choi, Y. Lee, D. Shin, J. Cho, S. Park, S. Lee, J. Baek, C. Bae, B. Kim, and J. Heo, "EdNet: A large-scale hierarchical dataset in education," in *Proceedings of the 21st International Conference on Artificial Intelligence in Education*, 2020, pp. 69–73.
- [35] L. Razzaq, J. Patvarczki, S. F. Almeida, M. Vartak, M. Feng, N. T. Heffernan, and K. R. Koedinger, "The assistant builder: Supporting the life cycle of tutoring system content creation," *IEEE Transactions on Learning Technologies*, vol. 2, no. 2, pp. 157–166, 2009.
- [36] H. Chang, H. Hsu, and K. Chen, "Modeling exercise relationships in e-learning: A unified approach," in *Proceedings of the 8th International Conference on Educational Data Mining*, pp. 532–535.
- [37] T. Long, J. Qin, J. Shen, W. Zhang, W. Xia, R. Tang, X. He, and Y. Yu, "Improving knowledge tracing with collaborative information," in *Proceedings of the 15th ACM International Conference on Web Search and Data Mining*, 2022, pp. 599–607.
- [38] B. Wang, "Mesh-Transformer-JAX: Model-Parallel Implementation of Transformer Language Model with JAX," <https://github.com/kingoflolz/mesh-transformer-jax>, May 2021.
- [39] J. Su, M. H. M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu, "Roformer: Enhanced transformer with rotary position embedding," *Neurocomputing*, vol. 568, p. 127063, 2024.
- [40] Z. Liu, Q. Liu, J. Chen, S. Huang, J. Tang, and W. Luo, "pyKT: A Python library to benchmark deep learning based knowledge tracing models," in *Proceedings of the 36th Annual Conference on Neural Information Processing Systems*, 2022.
- [41] Q. Liu, Z. Huang, Y. Yin, E. Chen, H. Xiong, Y. Su, and G. Hu, "EKT: Exercise-aware knowledge tracing for student performance prediction," *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 1, pp. 100–115, 2021.
- [42] L. McInnes, J. Healy, N. Saul, and L. Großberger, "UMAP: Uniform manifold approximation and projection," *Journal of Open Source Software*, vol. 3, no. 29, p. 861, 2018.



Hao Zhou is a BSc student in the School of Computer Science and Engineering, Beihang University. His research interests include natural language processing and machine learning.



Wenge Rong is a professor in the School of Computer Science and Engineering, Beihang University, China. He received his PhD from the University of Reading, UK, in 2010; MSc from Queen Mary College, University of London, UK, in 2003; and BSc from Nanjing University of Science and Technology, China, in 1996. He has many years of experience as a senior software engineer in numerous research projects and commercial software products. His area of research covers machine learning, natural language processing, and information management.



Jianfei Zhang received the BSc degree from the School of Computer Science of Engineering, Beihang University, in 2019. He is currently working toward a PhD degree with the School of Computer Science and Engineering, Beihang University. His research interests include natural language processing and machine learning.



Qing Sun is an associate professor in the School of Computer Science and Engineering, Beihang University. She received her Ph.D. from the Department of Information System of Beihang University in 2017; MSc and BSc from the School of Information Science of Beijing Normal University in 2008 and 2005, respectively. Her research interests include smart education, data mining, and knowledge engineering.



Yuanxin Ouyang received her BSc and PhD degrees from Beihang University, in 1997 and 2005, respectively. She is a professor at School of Computer Science and Engineering, Beihang University, China. Her area of research covers recommender systems, data mining, and social networks.



Zhang Xiong is a professor in the School of Computer Science and Engineering at Beihang University, and director of the Advanced Computer Application Research Engineering Center of the National Educational Ministry of China. He has published over 250 referred papers in international journals and conference proceedings and won a National Science and Technology Progress Award. His research interests span from smart cities, knowledge management, and information systems.