ACE-KT: Cascaded Cognitive Modeling for Stage-wise Knowledge Tracing

Teng Guo **Yubin Xia** **Jinsen Ke*** **Mingliang Hou** **Jiaqi Zheng** **Zitao Liu**
Guangdong Institute of Smart Education, Jinan University
Guangzhou, Guangdong, China

Abstract

Knowledge Tracing (KT) aims to predict students' academic performance by modeling their knowledge mastery over time, based on their historical learning interactions. However, current KT models often oversimplify student interactions by treating them as standard time series rather than as cognitive processes. Consequently, modeling student learning as a process of cognitive transformation rather than as a mere sequence of time-stamped events remains a fundamental challenge in KT research. To address this issue, we propose **ACE-KT** (c**A**scaded **C**ognitive mod**E**ling for **K**nowledge **T**racing), a novel framework inspired by cognitive process theory, which shifts the focus from purely sequential modeling to cognitive representation learning. Specifically, we design a cascaded cognitive framework inspired by human cognitive processes in three sequential stages: convolution-based rhythm perception module, Transformer encoder-based contextual structuring module, and cognitive integration module implemented via a selective structured state space model. Extensive experiments on five real-world datasets demonstrate that **ACE-KT** consistently outperforms 20 SOTA KT baselines, demonstrating its effectiveness. The source code is publicly available at our GitHub repository (https://github.com/AWord992/ACEKT.git).

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300. Copyright 2026 by the author(s). *Corresponding author: kejinsen@stu.jnu.edu.cn

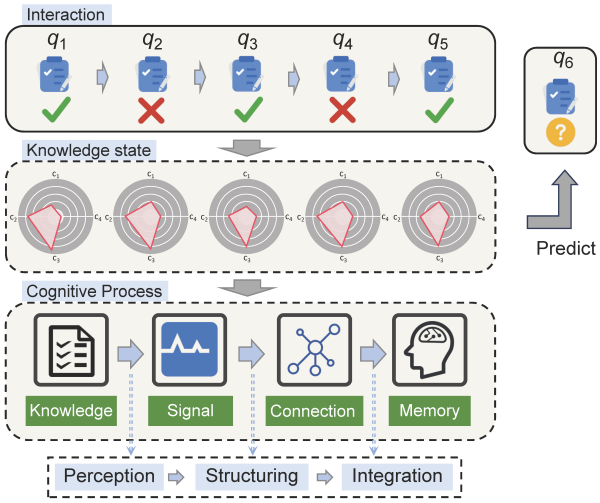

Figure 1: Illustration of the KT Task and Cognitive Process.

1 INTRODUCTION

Knowledge Tracing (KT) is the task of leveraging students' historical learning interaction data to model their evolving mastery of knowledge over time, with the ultimate goal of predicting their future performance in subsequent learning activities (as illustrated in Figure 1). Fundamentally, KT aims to estimate students' latent knowledge states, which reflect their understanding of specific concepts, based on their interactions with educational platforms such as Massive Open Online Courses (MOOCs) and Intelligent Tutoring Systems (Ossiannilsson, 2020; Bai et al., 2025; Zhan et al., 2024). Accurate estimation of these dynamic knowledge states empowers KT models to facilitate timely pedagogical interventions and provide adaptive feedback, thereby enhancing learning efficiency and fostering sustained student engagement (Piech et al., 2015; Chen et al., 2018; Zhang et al., 2017).

Since the 1990s, KT has been a prominent research

topic, with Corbett and Anderson pioneering the estimation of students' current understanding of individual knowledge components (KCs) (Corbett and Anderson, 1994). A KC refers to a cognitive process that a learner uses to perform specific steps in a task, either independently or in conjunction with other KCs. In recent years, the KT field has witnessed substantial progress with the advent of deep learning. A wide range of deep learning-based KT models (DLKT) have been proposed to capture the complex temporal and semantic dependencies embedded in student interaction data. Representative approaches include auto-regressive deep sequential models (Piech et al., 2015; Yeung and Yeung, 2018; Chen et al., 2018; Guo et al., 2021), memory-augmented KT models (Zhang et al., 2017), attention-based KT models (Pandey and Karypis, 2019; Ghosh et al., 2020), and graph-based KT models (Tong et al., 2020; Tu et al., 2023). Beyond architectural innovations, these studies have also incorporated various cognitive perspectives, such as modeling forgetfulness, carelessness, and other behavioral factors, to deepen our understanding of the cognitive process and improve the performance of KT models.

Existing educational and psychological research has evidenced that the cognitive process is not instantaneous but unfolds gradually through multiple cascaded cognitive stages (Kolb, 1984; Brewer and Pani, 1983). Specifically, internalizing knowledge typically involves three key steps (shown in Figure 1): (1) perception and initial encoding of learning signals, (2) structured representation of relationships among concepts or tasks, and (3) temporal integration of knowledge over time to form a coherent mental model. These stages reflect how learners process information from recognizing content to understanding its structure, and ultimately incorporating it into long-term knowledge, with each stage building on the previous one and requiring a distinct form of cognitive processing. However, most existing KT models treat the cognitive process as a flat stream of events, encoding each interaction in a uniform manner without regard to its cognitive role or position within the broader learning trajectory. Formally, the cognitive process is modeled as a temporal sequence rather than a structured cognitive process. This flattened view oversimplifies the complexity of human learning, so it not only struggles to capture how students transition from initial exposure to structured understanding and eventual integration, but also limits the model's ability to reflect authentic learning behaviors. As a consequence, such models often suffer from reduced performance when applied in real-world educational settings.

To address the limitations discussed above, we propose a new KT model named ACE-KT (c**A**scaded **C**ognitive mod**E**ling for **KT**), which aims to better capture the evolving nature of student learning by modeling the internal structure and progression of cognitive processing. Specifically, we design a novel multi-stage KT block called the cascaded cognitive block, which is composed of three sequential modules. First, we introduce rhythm perception module (RPM) that performs a residual convolution on the input embeddings (e.g., question, KC, difficulty), separating stable patterns from local fluctuations and generating a rhythm perception representation. Then, this representation is passed into contextual structuring module (CSM) based on a Transformer encoder, which learns contextual interactions and KC dependencies. To capture the integration phase of learning, we introduce cognitive integration module (CIM) at the final stage of our cascade block. Here, we improve the selective structured state space module (SSM) by incorporating a hierarchical representation structure inspired by the Hungry Hungry Hippos model (H3) (Dao et al., 2023) and introducing the TeLU activation function (Fernandez, 2024), allowing more expressive modeling of nonlinear cognitive transitions and adaptive learning dynamics. The key contributions of this work can be summarized as follows:

- We propose ACE-KT, a novel framework built upon a cascaded cognitive block that explicitly models the core stages of cognitive evolution: perception, structuring, and integration.
- We introduce a novel stage-wise cognitive modeling paradigm that offers a targeted and interpretable framework to understand student cognitive processes, paving the way for future innovations in cognitively informed KT models.
- We conduct extensive experiments on five real-world datasets, showing that ACE-KT consistently outperforms 20 state-of-the-art KT baselines, thereby validating the significance of capturing the cognitive dynamics of student learning.

2 RELATED WORKS

2.1 Knowledge Tracing

In recent years, the field of KT has witnessed significant progress driven by the accelerated development of deep learning. Piech et al. (2015) introduced the Deep KT (DKT) model, which was the first to use recurrent neural networks (RNN) to encode a student's knowledge state as a hidden vector, thus pioneering the application of deep learning in KT tasks. Since then,

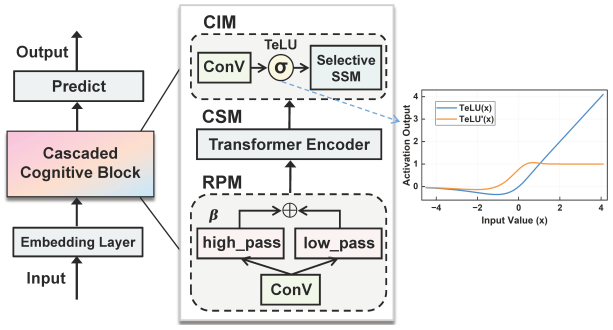

Figure 2: The ACE-KT Framework.

a wide range of methods have used deep learning to model the complex dynamics of human learning, consistently outperforming traditional methods oriented to psychology (Nagatani et al., 2019; Yeung and Yeung, 2018; Minn et al., 2018; Lee and Yeung, 2019; Guo et al., 2021; Chen et al., 2018).

More recently, attention-based architectures have gained prominence for their ability to effectively model dependencies within student interaction data, achieving better flexibility than traditional deep neural network-based methods. Among these, Self-Attentive KT (SAKT) was the first model to introduce attention mechanisms into KT (Pandey and Karypis, 2019). Subsequently, Ghosh et al. (2020) introduced the AKT model, which incorporates three distinct attention mechanisms to extract latent relationships within the complex student–question interaction patterns. The simpleKT model, developed by Liu et al. (2023b), addresses question-specific variability through a simple dot-product attention mechanism that captures time-sensitive features of students' learning interactions, thereby revealing nuanced distinctions among items linked to common knowledge components. Most recently, the DTransformer framework tracks learner proficiency by diagnosing question-level mastery and uses contrastive learning to improve the stability of knowledge state estimation (Yin et al., 2023).

However, most existing approaches still tend to encode the cognitive process as a uniform sequence of timestamped events, without explicitly modeling the multi-stage nature of cognitive processing (e.g., perception, structuring, and integration). This oversimplification can limit their ability to capture how knowledge is progressively constructed and consolidated, motivating the need for more cognitively grounded modeling frameworks.

2.2 Cognitive Science

Cognitive science seeks to understand mental processes through structured and hierarchical frameworks of human cognition. Early cognitive science viewed mental activities primarily as symbolic manipulations governed by logical rules, leading to classical computational models that simulated cognitive tasks symbolically (Newell et al., 1958). Subsequently, the field shifted toward connectionist approaches, modeling cognition as distributed processing across interconnected neural units, emphasizing parallel computation and dynamic learning through experience (Rumelhart et al., 1986).

More recent cognitive theories adopt an integrative and dynamic perspective, highlighting the multi-stage nature of human cognitive processes. For instance, Kolb's experiential learning theory describes cognitive development as iterative cycles of perceiving stimuli, abstracting structures, and integrating experiences into existing mental models (Kolb, 1984). Similarly, Brewer emphasizes structured representations within cognitive schemas, where the relational encoding of information is fundamental to effective learning (Brewer and Pani, 1983). Modern state space theories, though developed independently, are consistent with this notion, as both view cognitive processes as dynamic transitions between latent states influenced by prior knowledge and new input, similar to the state transitions modeled in selective SSM (Van Der Maas et al., 2006; Gu et al., 2022).

Inspired by these cognitive frameworks, our ACE-KT explicitly structures its architecture into perception, structuring, and integration stages. This design reflects cognitive science's current understanding of learning as a hierarchical and dynamic process, enabling the model to more accurately represent students' evolving cognitive states.

3 PROBLEM DEFINITION

In this section, we formally define the KT task. Let $S = \{s_i\}_{i=1}^{M}$ be the set of students, $Q = \{q_i\}_{i=1}^{K}$ the set of questions, and $C = \{c_i\}_{i=1}^{N}$ the set of KCs. The answer sequence for student s_i over T time steps is represented as $R_i = \{(q_1^i, c_1^i, r_1^i), (q_2^i, c_2^i, r_2^i), \ldots, (q_T^i, c_T^i, r_T^i)\}$. Each triple (q_t^i, c_t^i, r_t^i) corresponds to an interaction record of student s_i at time step t, where q_t^i is the question, c_t^i is the KC related to the question, and r_t^i is the student's response to the question.

Given a student's historical interaction sequence over the previous T time steps, our goal is to predict the response at time step $(T+1)$ for the target question.

4 PROPOSED METHOD

In this section, we introduce ACE-KT, a cognitively grounded KT model built upon a cascaded cognitive block, which sequentially models the perception, structuring, and integration stages of student learning to better reflect the progression of cognitive processes (shown in Figure 2). Specifically, we first outline the theoretical motivation behind cascaded cognitive modeling, followed by a detailed introduction of each component in the proposed framework.

4.1 Theoretical Motivation for Cascaded Cognitive Modeling

DLKT models often rely on stacked homogeneous architectures, such as multiple layers of Transformer blocks, to capture complex sequential dependencies. However, this approach introduces several limitations from both cognitive and representational perspectives (Kumar et al., 2024; Béna and Goodman, 2025).

First, unified architectures lack functional specialization. In real-world learning, cognitive processing unfolds in distinct phases, such as perceiving stimuli, interpreting structural meaning, and consolidating acquired knowledge. Assigning all these tasks to a single repeated module places excessive burden on the network, increasing the risk of gradient conflict and over-smoothing. Formally, let a repeated module f_θ be stacked L times, producing intermediate representations $\{\mathbf{H}^{(l)}\}_{l=1}^{L}$:

$$\mathbf{H}^{(l)} = f_\theta(\mathbf{H}^{(l-1)}), \quad \mathbf{H}^{(0)} = \mathbf{X}. \tag{1}$$

In this homogeneous setting, all cognitive functions are implicitly entangled in f_θ, making it hard to assign gradient paths to distinct roles. This can be measured by computing inter-layer representation similarity:

$$\mathrm{Sim}(\mathbf{H}^{(l)}, \mathbf{H}^{(l+1)}) \to 1, \quad \text{as } l \to L. \tag{2}$$

This compounded gradient chain becomes increasingly entangled and aligned across layers, undermining the model's capacity to separate perception from reasoning and long-term consolidation. (Experiment details are shown in Appendix C)

Second, such homogeneous stacking may lead to representation collapse, where multiple layers gradually reduce the variance of internal representations, especially for input learning behaviors that are not frequently observed (Raghu et al., 2021; Wang et al., 2022). Formally, given an input sequence $\mathbf{X} \in \mathbb{R}^{T\times d}$ and its final representation $\mathbf{H}^{(L)}$ after L transformation layers, it has been empirically observed that:

$$\mathrm{rank}\left(\frac{\partial \mathbf{H}^{(L)}}{\partial \mathbf{X}}\right) \ll d, \tag{3}$$

indicating that the model primarily retains a small number of dominant representational directions. This phenomenon, which may reduce sensitivity to transient behaviors such as careless mistakes or bursty guessing, has been discussed in prior studies on deep models.

To address these issues, we propose a cascaded cognitive block that aligns the architecture with human cognitive stages:

- **RPM:** A 1D convolutional layer highlights local temporal fluctuations in learning behaviors (e.g., answering rhythm), thereby enriching the representational space and capturing short-term irregularities. We denote its output as $\tilde{\mathbf{X}} = f_{\text{percep}}(\mathbf{X})$.
- **CSM:** A Transformer encoder block captures conceptual dependencies among learning elements via self-attention over the rhythm-perception input $\tilde{\mathbf{X}}$, producing $\mathbf{H} = f_{\text{struct}}(\tilde{\mathbf{X}})$, where $\mathbf{H}$ represents the structured and context-aware representation of the sequence.
- **CIM:** An enhanced selective SSM simulates the integration stage by selectively aggregating structured knowledge over time, generating the final integrated representation $\mathbf{y} = f_{\text{integ}}(\mathbf{H})$.

The cascaded design enables a clear and modular gradient path from output to input, which can be abstractly expressed as:

$$\frac{\partial \mathbf{y}}{\partial \mathbf{X}} = \frac{\partial \mathbf{y}}{\partial \mathbf{H}} \cdot \frac{\partial \mathbf{H}}{\partial \tilde{\mathbf{X}}} \cdot \frac{\partial \tilde{\mathbf{X}}}{\partial \mathbf{X}}. \tag{4}$$

This formulation highlights how the structured chain of derivatives facilitates more stable optimization and reduces gradient entanglement across stages. Each module corresponds to a distinct cognitive function, thereby enabling targeted optimization. To further substantiate this claim, we conduct a detailed gradient similarity analysis across the three modules, examining the alignment and conflict of their gradient directions during optimization. The full experimental setup and results are provided in Appendix C.

4.2 Embedding Layer

Recognizing that questions associated with the same KCs possess heterogeneous difficulty levels, it is imperative to accurately model student interactions. We represent the interaction sequences $(q_1, c_1, r_1), ..., (q_T, c_T, r_T)$ as follows:

$$\begin{aligned} \mathbf{x}_t &= \mathbf{d}_{q_t} \odot \mathbf{v}_{c_t} \oplus \mathbf{e}_{c_t}, \\ \mathbf{y}_t &= \mathbf{d}_{q_t} \odot \mathbf{v}_{(c_t, r_t)} \oplus \mathbf{e}_{(c_t, r_t)} \end{aligned} \tag{5}$$

where $\mathbf{x}_t$ represents the latent representations of question q_t and its related KC c_t at the t-th timestamp. $\mathbf{d}_{q_t}$ denotes the learnable difficulty of question q_t. The vector $\mathbf{v}_{c_t}$ captures the variation of the KC, while $\mathbf{e}_{c_t}$ represents the N-dimensional one-hot encoding of c_t. $\odot$ and $\oplus$ are the element-wise product and addition operators. Respectively, $\mathbf{y}_t$ represents the augmented representation of $\mathbf{x}_t$ by incorporating the response r_t to the question q_t. $\mathbf{e}_{(c_t,r_t)}$ incorporates the representations of both c_t and r_t. The vector $\mathbf{v}_{(c_t,r_t)}$ represents the KC-response variation of q_t, capturing the interaction between the KC c_t and the response r_t.

4.3 Cascaded Cognitive Block

In this section, the three modules are introduced in turn.

4.3.1 Rhythm Perception Module

The RPM serves as the initial stage of the cascaded cognitive block, functioning as a pre-sensory tuner that highlights short-term and long-term temporal fluctuations and enhances the diversity of input representations. The module processes the sequences of $\mathbf{x}_t$ and $\mathbf{y}_t$ to produce two distinct output sequences, $\mathbf{z}_t^{(x)}$ and $\mathbf{z}_t^{(y)}$:

$$\mathbf{z}_t^{(x)} = f_{\text{percep}}(\mathbf{x}_t), \quad \mathbf{z}_t^{(y)} = f_{\text{percep}}(\mathbf{y}_t) \tag{6}$$

To simplify the explanation of the module's inner workings, we will describe the operations on a generic input sequence $\mathbf{H}_{in} \in \mathbb{R}^{B\times T\times H}$, where B denotes the batch size, T represents the sequence length and H is the hidden dimension of the input embeddings.

We adopt a causal 1D convolution to extract stable local patterns while preserving temporal causality. This operation decomposes the input into a stable component $\mathbf{H}_{stable}$ and a fluctuation component $\mathbf{H}_{fluc}$ as:

$$\begin{aligned} \mathbf{H}_{stable} &= \text{CausalConv1D}(\mathbf{H}_{in}), \\ \mathbf{H}_{fluc} &= \mathbf{H}_{in} - \mathbf{H}_{stable}. \end{aligned} \tag{7}$$

To modulate the contribution of transient fluctuations, a learnable scaling coefficient β is applied:

$$\mathbf{H}_{adjusted} = \mathbf{H}_{stable} + \beta^2 \mathbf{H}_{fluc}. \tag{8}$$

Here, the term β^2 adaptively controls the degree to which short-term irregularities are emphasized relative to stable patterns. The resulting signal is further processed through dropout and layer normalization with a residual connection to stabilize training:

$$\mathbf{z}_t = \text{LayerNorm}\big(\text{Dropout}(\mathbf{H}_{adjusted}) + \mathbf{H}_{in}\big). \tag{9}$$

The output $\mathbf{z}_t \in \mathbb{R}^{B\times T\times H}$ (representing either $\mathbf{z}_t^{(x)}$ and $\mathbf{z}_t^{(y)}$) thus preserves essential temporal structures while improving representational variance. This rhythm perception representation provides a cleaner and more separable feature space for the subsequent CSM.

4.3.2 Contextual Structuring Module

Given the rhythm perception representations $\mathbf{z}_t^{(x)}$ and $\mathbf{z}_t^{(y)}$ from the RPM, CSM leverages a Transformer encoder block to capture high-order dependencies among learning elements. To prevent information leakage from future interactions, we follow prior work on attention-based KT (Liu et al., 2023b; Pandey and Karypis, 2019) and apply a causal masking mechanism that restricts each position to attend only to its historical interactions and itself.

Formally, we define a causal mask matrix $\mathbf{M} \in \mathbb{R}^{T\times T}$ as:

$$M_{i,j} = \begin{cases} 0, & j \le i, \\ -\infty, & j > i, \end{cases} \tag{10}$$

where i and j index temporal positions. This mask is injected into the attention computation to nullify contributions from future positions. The masked scaled dot-product attention is then defined as:

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}} + \mathbf{M}\right)\mathbf{V}, \tag{11}$$

where the query, key, and value matrices are computed as:

$$\mathbf{Q} = \mathbf{z}_t^{(x)}\mathbf{W}^Q, \quad \mathbf{K} = \mathbf{z}_t^{(x)}\mathbf{W}^K, \quad \mathbf{V} = \mathbf{z}_t^{(y)}\mathbf{W}^V, \tag{12}$$

with $\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V \in \mathbb{R}^{H\times d_k}$ denoting learnable projection matrices.

To further enhance representational power, we adopt the multi-head attention mechanism. Specifically, the outputs of h parallel attention heads are concatenated and linearly transformed:

$$\text{MultiHead}(\mathbf{z}_t^{(x)}, \mathbf{z}_t^{(y)}) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)\mathbf{W}^O, \tag{13}$$

where each attention head is computed independently as:

$$\text{head}_i = \text{Attention}(\mathbf{z}_t^{(x)}\mathbf{W}_i^Q, \mathbf{z}_t^{(x)}\mathbf{W}_i^K, \mathbf{z}_t^{(y)}\mathbf{W}_i^V), \tag{14}$$

and $\mathbf{W}^O \in \mathbb{R}^{hd_k\times H}$ is a learnable output projection.

The resulting structured representation $\mathbf{H}_{struct} \in \mathbb{R}^{B\times T\times H}$ encodes both temporal and conceptual dependencies, providing a context-aware foundation for the subsequent module.

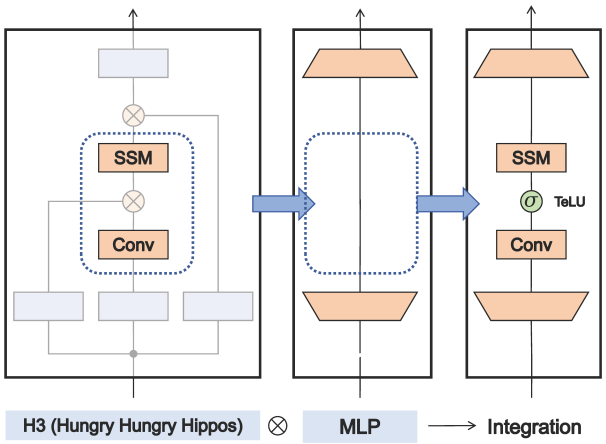

Figure 3: We present an integrated architecture combining the H3 model (Dao et al., 2023) and MLP blocks into a single unified module. Compared to the H3 block, which interleaves convolution and SSM with gating, the proposed structure simplifies the gating mechanism by introducing a TeLU activation (Fernandez, 2024) between the convolution and SSM layers. This design allows for efficient local feature enhancement and global dependency modeling in a streamlined form.

4.3.3 Cognitive Integration Module

To capture the integration phase of the learning process and enhance the model's capacity for long-range, temporally diffuse dependencies, we incorporate a selective SSM as the final stage of the cascade block. This design is grounded in cognitive learning theories, which conceptualize human learning as a sequence of latent cognitive state transitions (Van Der Maas et al., 2006; Gu et al., 2022). At each learning step, students internalize new information and gradually consolidate it with existing knowledge, a process analogous to updating an internal hidden state based on prior memory and present input. This closely resembles the internal mechanism of the selective SSM. To further enhance the model's capability in capturing cognitive integration processes, we introduce several improvements to the standard selective SSM, detailed as follows.

First, inspired by the H3 model (Dao et al., 2023), we introduce a causal 1D convolution layer to enhance the awareness of local information (shown in Figure 3). Given the structured input $\mathbf{H}_{struct} \in \mathbb{R}^{B\times T\times H}$ from the CSM, we first project it into an expanded latent space:

$$\mathbf{X}_{\text{conv}} = \text{CausalConv1D}(\mathbf{H}_{struct}\mathbf{W}_{\text{in}}), \tag{15}$$

where $\mathbf{W}_{\text{in}} \in \mathbb{R}^{H\times D_{\text{inner}}}$ is a learnable projection and CausalConv1D$(\cdot)$ captures short-term temporal dynamics.

Second, the result is then passed through a TeLU activation (Fernandez, 2024) to enhance gradient flow:

$$\mathbf{X} = \text{TeLU}(\mathbf{X}_{\text{conv}}) = \mathbf{X}_{\text{conv}} \odot \tanh\big(\exp(\mathbf{X}_{\text{conv}})\big). \tag{16}$$

TeLU is selected over SiLU (used in Mamba (Gu and Dao, 2023)) because it better approximates the identity function in its active region and preserves non-zero gradients for negative inputs, reducing neuron inactivation. These properties facilitate faster convergence and more stable modeling of students' evolving cognitive states over time.

After that, to model long-term dependencies and selective memory, we adopt a selective SSM model:

$$\mathbf{h}_{t+1} = \mathbf{A}_t\odot\mathbf{h}_t+\mathbf{B}_t\odot\mathbf{X}_t, \quad \mathbf{Y}_t = \mathbf{C}_t^\top\mathbf{h}_t+\mathbf{D}\odot\mathbf{X}_t, \tag{17}$$

where $\mathbf{h}_t \in \mathbb{R}^{B\times D_{\text{inner}}}$ denotes the latent integration state, $\mathbf{A}_t, \mathbf{D} \in \mathbb{R}^{D_{\text{inner}}}$ are global transition parameters, and $\mathbf{B}_t, \mathbf{C}_t \in \mathbb{R}^{D_{\text{inner}}}$ are input-dependent gates learned adaptively at each time step.

Finally, the output $\mathbf{Z} \in \mathbb{R}^{B\times T\times H}$ is obtained through a residual projection:

$$\mathbf{Z} = \mathbf{Y}\mathbf{W}_{\text{out}} + \mathbf{H}_{struct}, \tag{18}$$

where $\mathbf{W}_{\text{out}} \in \mathbb{R}^{D_{\text{inner}}\times H}$.

This design enables selective information retention and rhythm-aware abstraction, further aligning with cognitive theories that view learning as a series of latent state transitions. Unlike standard SSMs, we explicitly integrate rhythm-perception convolution, TeLU activation, and adaptive gating to enhance the modeling of fine-grained temporal irregularities observed in educational data.

4.4 Prediction Layer

After the cascaded cognitive block processes sequential learning interactions through three stages, we obtain the final enriched representation $\mathbf{Z}_{t+1} \in \mathbb{R}^{B\times H}$. To transform these representations into performance predictions, we employ a two-layer fully connected neural network:

$$\hat{\mathbf{r}}_{t+1} = \sigma(\mathbf{W}_2\,\delta(\mathbf{W}_1[\mathbf{Z}_{t+1} : \mathbf{x}_{t+1}] + \mathbf{b}_1) + \mathbf{b}_2)\,, \tag{19}$$

where $[\mathbf{Z}_{t+1} : \mathbf{x}_{t+1}]$ denotes the concatenation of the cascade-processed representation $\mathbf{Z}_{t+1}$ and the current question embedding $\mathbf{x}_{t+1}$. $\sigma(\cdot)$ and $\delta(\cdot)$ denote the Sigmoid and ReLU activation functions, respectively. $\mathbf{W}_1$, $\mathbf{W}_2$, $\mathbf{b}_1$, and $\mathbf{b}_2$ are trainable parameters.

We train the model by minimizing the binary cross-entropy loss between the actual student responses $\mathbf{r}_{t+1}$

Table 1: Statistics of the Datasets.

Dataset	#Students	#KCs	#Questions	#Interactions
AS2009	3,852	123	17,737	337,415
AL2005	574	112	173,113	607,021
BD2006	1,145	493	129,263	1,817,458
NIPS34	4,918	57	948	1,382,678
EdNet	4,687	188	11,901	596,600

and the predicted probabilities $\hat{\mathbf{r}}_{t+1}$:

$$\mathcal{L} = -\sum_{t} \left[\mathbf{r}_{t+1} \log \hat{\mathbf{r}}_{t+1} + (1 - \mathbf{r}_{t+1}) \log(1 - \hat{\mathbf{r}}_{t+1})\right]. \tag{20}$$

5 EXPERIMENTS

In this section, we first introduce the experimental setup, including the five real-world datasets used, the baseline methods, and the implementation details. We then analyze the experimental results to address the following three questions:

- **RQ1**: Does ACE-KT outperform current KT frameworks?
- **RQ2**: What is the contribution of each functional component of ACE-KT to its overall performance?
- **RQ3**: Does ACE-KT maintain computational efficiency despite its cascading design?

5.1 Experimental Setting

5.1.1 Datasets

We evaluate the effectiveness of ACE-KT across five widely used datasets, including AS2009, AL2005, BD2006, NIPS34 and EdNet (Liu et al., 2022), to comprehensively evaluate the performance of our models. Table 1 presents the statistics for all datasets. More detailed information on these five datasets and experiments can be found in the Appendix A.

5.1.2 Baselines

To comprehensively and systematically evaluate the performance of ACE-KT, we selected 20 state-of-the-art KT methods as baselines for comparison, including DKT (Piech et al., 2015), DKT+ (Yeung and Yeung, 2018), DKT-F (Nagatani et al., 2019), KQN (Lee and Yeung, 2019), DKVMN (Zhang et al., 2017), ATKT (Guo et al., 2021), GKT (Nakagawa et al., 2019), SAKT (Pandey and Karypis, 2019), SAINT (Choi et al., 2020), AKT (Ghosh et al., 2020), SKVMN (Abdelrahman and Wang, 2019), HAWKES (Wang et al., 2021), Deep-IRT (Yeung, 2019), DIMKT (Shen et al., 2022), AT-DKT (Liu et al., 2023a), simpleKT (Liu et al., 2023b), FoLiBiKT (Im et al., 2023), DTransformer (Yin et al., 2023), HD-KT (Ma et al., 2024), and extraKT (Li et al., 2024). Details of these comparison models are provided in Appendix A.

Table 2: Performance Comparisons in Terms of AUC.

Model	AUC				
	AS2009	**AL2005**	**BD2006**	**NIPS34**	**EdNet**
DKT	0.7541	0.8149	0.8015	0.7689	0.6133
DKT+	0.7547	0.8156	0.8020	0.7696	0.6189
DKT-F	–	0.8147	0.7985	0.7733	0.6168
KQN	0.7477	0.8027	0.7936	0.7684	0.6111
DKVMN	0.7473	0.8054	0.7983	0.7673	0.6158
ATKT	0.7470	0.7995	0.7889	0.7665	0.6065
GKT	0.7424	0.8110	0.8046	0.7689	0.6223
SAKT	0.7246	0.7880	0.7740	0.7517	0.6072
SAINT	0.6958	0.7775	0.7781	0.7873	0.6614
AKT	<u>0.7853</u>	0.8306	0.8208	0.8033	0.6721
SKVMN	0.7332	0.7463	0.7287	0.7513	0.6182
HAWKES	0.7224	0.8210	0.8068	0.7767	0.6837
Deep-IRT	0.7465	0.8040	0.7976	0.7672	0.6173
DIMKT	0.7717	0.8277	0.8167	0.8030	0.6748
AT_DKT	0.7555	0.8246	0.8104	0.7816	0.6249
simpleKT	0.7744	0.8254	0.8160	0.8035	0.6599
FoLiBiKT	0.7838	0.8316	<u>0.8208</u>	0.8033	0.6747
DTransformer	0.7725	0.8188	0.8093	0.7944	0.6736
extraKT	0.7824	<u>0.8317</u>	0.8110	<u>0.8045</u>	0.6730
HD-KT	0.7736	0.8229	0.8039	0.7932	**0.6992**
ACE-KT	**0.7877**	**0.8373**	**0.8219**	**0.8045**	<u>0.6859</u>

5.2 Main Results (RQ1)

Based on the AUC results presented in Table 2, ACE-KT achieves the best AUC on four of the five benchmark datasets and remains highly competitive on EdNet compared to 20 state-of-the-art KT methods proposed between 2015 and 2024. Specifically, ACE-KT achieves the highest AUC scores on AS2009, AL2005, BD2006, and NIPS34, and its performance is only 0.014 below the top method on EdNet. Furthermore, ACE-KT demonstrates highly competitive performance on ACC. More details of experimental results are shown in Appendix B. These extensive experimental results not only demonstrate the effectiveness of the proposed method but also validate the stage-wise cognitive modeling paradigm in accurately capturing cognitive processes.

While the observed improvements in AUC and ACC metrics may seem modest (less than 1%) on certain datasets, they remain meaningful considering the con-

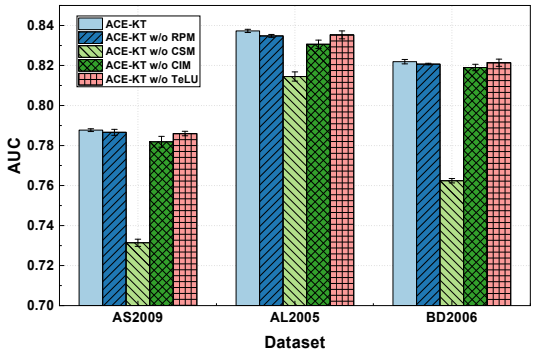

Figure 4: Ablation Experiment Results of AUC

text window size of 200 interactions. Recent benchmarking analyses (Liu et al., 2022) indicate that many performance gains reported in KT literature lack statistical reliability, and the cumulative improvement in AUC metrics since 2015 is only about 3.5%. By rigorously adhering to the evaluation standards and protocols recommended by pyKT (Liu et al., 2022) and performing comprehensive hyperparameter optimization for all baseline models, our evaluation provides a robust and unbiased comparison, ensuring the validity and significance of our results.

5.3 Ablation Study (RQ2)

To investigate the contribution of each component in the proposed cascaded cognitive block, we conducted a series of ablation experiments. Specifically, we examined the impact of (i) the RPM, (ii) the CSM, (iii) the CIM, and (iv) the TeLU activation within the CIM. All experiments were performed under the same experimental setup, and performance was measured by AUC and ACC (shown in Figure 4 and Figure 6 in Appendix B).

Several key observations can be classified into four aspects: (1) Removing the RPM (w/o RPM) consistently decreased AUC by 0.1% $\sim$ 0.3% on all datasets, confirming its role in enhancing local temporal sensitivity. (2) Eliminating the CSM (w/o CSM) caused the most severe performance degradation (up to 7.2% AUC drop on BD2006), indicating that the self attention-based relational structuring is indispensable for capturing dependencies among learning elements. (3) Removing the CIM (w/o CIM) also resulted in noticeable declines (up to 0.8% AUC drop), validating the effectiveness of the selective SSM mechanism for long range knowledge consolidation. (4) Replacing the TeLU activation with SiLU (w/o TeLU) led to smaller but consistent performance drops (0.1% $\sim$ 0.2% AUC), demonstrating that TeLU's improved gradient flow contributes to more stable training and better predictive accuracy.

Overall, the full ACE-KT model outperforms all variants across datasets, highlighting the complementary contributions of the three modules and the TeLU activation design.

5.4 Computational Complexity (RQ3)

ACE-KT maintains a comparable computational complexity to standard Transformer-based KT models. Although it introduces additional modules, such as causal convolution and selective SSM, the overall cost remains dominated by the attention mechanism in the relational encoding layer, whose complexity is $\mathcal{O}(T^2 \cdot H + T \cdot H^2)$, where T is the sequence length and H is the hidden dimension. The additional modules do not change the overall quadratic complexity order with respect to T. Specifically, the causal convolution in RPM introduces a complexity of $\mathcal{O}(kTH^2)$ under a standard dense Conv1d implementation, while the core selective scan in CIM is linear in T and the overall stage complexity is $\mathcal{O}(TH^2)$. Consequently, ACE-KT achieves enhanced cognitive modeling capacity with only modest additional computational overhead. Details are shown in Appendix D.

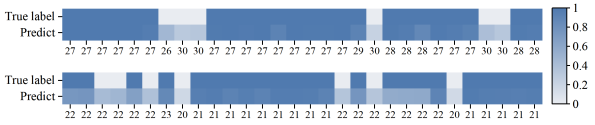

Figure 5: Visualization of the Prediction of ACE-KT

5.5 Qualitative Analysis

To qualitatively analyze the performance of ACE-KT, we randomly select a group of students from the five datasets used in our study and visualize their answering performance, as shown in Figure 5. The horizontal axis represents the sequence of different KCs involved in the questions, and the vertical axis shows the predicted probabilities (Predict) and the ground truth labels (True label). From the temporal perspective, consistent exposure to relevant learning signals facilitates stable knowledge acquisition. In contrast, from the structural perspective, irregular or incoherent relationships between KCs may disrupt the organization of knowledge and slow cognitive progression. ACE-KT is explicitly designed to disentangle and capture such temporal–structural inconsistencies through its cascaded cognitive modeling framework.

6 CONCLUSION

In this paper, we propose ACE-KT, a novel framework grounded in cognitive process theory that moves beyond purely sequential modeling toward cognitively informed representations of learning. We propose a three-module cascaded cognitive block that simulates the human cognitive process through three sequential stages. The perception stage employs a convolutional layer to extract pre-attentive patterns from the input sequence. The structuring stage leverages a Transformer encoder to model contextual relationships and KC dependencies. Finally, the integration stage utilizes an enhanced selective SSM, which incorporates stacked linear transformations and TeLU activations to support temporal abstraction and consolidate cognitive representations. We conduct comprehensive experiments on five real-world datasets, where ACE-KT consistently achieves superior performance compared to 20 state-of-the-art KT models. These results confirm the model's capability to effectively capture the cognitive dynamics underlying student learning processes.

Acknowledgments

This work was supported in part by GuangDong Basic and Applied Basic Research Foundation (Grant No. 2025A1515011171) and in part by the Key Laboratory of Smart Education of Guangdong Higher Education Institutes, Jinan University (2022LSYS003).

References

Abdelrahman, G. and Wang, Q. (2019). Knowledge tracing with sequential key-value memory networks. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 175–184.

Bai, Y., Li, X., Liu, Z., Huang, Y., Guo, T., Hou, M., Xia, F., and Luo, W. (2025). cskt: Addressing cold-start problem in knowledge tracing via kernel bias and cone attention. *Expert Systems with Applications*, 266:125988.

Béna, G. and Goodman, D. F. (2025). Dynamics of specialization in neural modules under resource constraints. *Nature Communications*, 16(1):187.

Brewer, W. F. and Pani, J. R. (1983). The structure of human memory. In *Psychology of Learning and Motivation*, volume 17, pages 1–38. Elsevier.

Chen, P., Lu, Y., Zheng, V. W., and Pian, Y. (2018). Prerequisite-driven deep knowledge tracing. In *2018 IEEE International Conference on Data Mining*, pages 39–48. IEEE.

Choi, Y., Lee, Y., Cho, J., Baek, J., Kim, B., Cha, Y., Shin, D., Bae, C., and Heo, J. (2020). Towards an appropriate query, key, and value computation for knowledge tracing. In *Proceedings of the seventh ACM conference on learning@ scale*, pages 341–344.

Corbett, A. T. and Anderson, J. R. (1994). Knowledge tracing: modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, 4:253–278.

Dao, T., Fu, D. Y., Saab, K. K., Thomas, A. W., Rudra, A., and Ré, C. (2023). Hungry hungry hippos: Towards language modeling with state space models. In *Proceedings of the 11th International Conference on Learning Representations (ICLR)*.

Fernandez, A. (2024). Telu activation function for fast and stable deep learning. Master's thesis, University of South Florida.

Ghosh, A., Heffernan, N., and Lan, A. S. (2020). Context-aware attentive knowledge tracing. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2330–2339.

Gu, A. and Dao, T. (2023). Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*.

Gu, A., Goel, K., and Ré, C. (2022). Efficiently modeling long sequences with structured state spaces. In *Proceedings of the 10th International Conference on Learning Representations (ICLR)*.

Guo, X., Huang, Z., Gao, J., Shang, M., Shu, M., and Sun, J. (2021). Enhancing knowledge tracing via adversarial training. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 367–375.

Im, Y., Choi, E., Kook, H., and Lee, J. (2023). Forgetting-aware linear bias for attentive knowledge tracing. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pages 3958–3962.

Kolb, D. A. (1984). Experiential learning: Experience as the source of learning and development. *EC.: Prentice Hall*.

Kumar, S., Sumers, T. R., Yamakoshi, T., Goldstein, A., Hasson, U., Norman, K. A., Griffiths, T. L., Hawkins, R. D., and Nastase, S. A. (2024). Shared functional specialization in transformer-based language models and the human brain. *Nature communications*, 15(1):5523.

Lee, J. and Yeung, D.-Y. (2019). Knowledge query network for knowledge tracing: how knowledge interacts with skills. In *Proceedings of the 9th International conference on Learning Analytics & Knowledge*, pages 491–500.

Li, X., Bai, Y., Guo, T., Zheng, Y., Hou, M., Zhan, B., Huang, Y., Liu, Z., Gao, B., and Luo, W. (2024). Extending context window of attention based knowledge tracing models via length extrapolation. In *ECAI 2024*, pages 1479–1486. IOS Press.

Liu, Z., Liu, Q., Chen, J., Huang, S., Gao, B., Luo, W., and Weng, J. (2023a). Enhancing deep knowledge tracing with auxiliary tasks. In *Proceedings of the 2023 World Wide Web Conference*.

Liu, Z., Liu, Q., Chen, J., Huang, S., and Luo, W. (2023b). simpleKT: a simple but tough-to-beat baseline for knowledge tracing. In *Proceedings of the 11th International Conference on Learning Representations (ICLR)*.

Liu, Z., Liu, Q., Chen, J., Huang, S., Tang, J., and Luo, W. (2022). pykt: a python library to benchmark deep learning based knowledge tracing models. *Advances in Neural Information Processing Systems*, 35:18542–18555.

Ma, H., Yang, Y., Qin, C., Yu, X., Yang, S., Zhang, X., and Zhu, H. (2024). Hd-kt: advancing robust knowledge tracing via anomalous learning interaction detection. In *Proceedings of the ACM on Web Conference 2024*, pages 4479–4488.

Minn, S., Yu, Y., Desmarais, M. C., Zhu, F., and Vie, J.-J. (2018). Deep knowledge tracing and dynamic student classification for knowledge tracing. In *2018 IEEE International Conference on Data Mining*, pages 1182–1187. IEEE.

Nagatani, K., Zhang, Q., Sato, M., Chen, Y.-Y., Chen, F., and Ohkuma, T. (2019). Augmenting knowledge tracing by considering forgetting behavior. In *The World Wide Web Conference*, pages 3101–3107.

Nakagawa, H., Iwasawa, Y., and Matsuo, Y. (2019). Graph-based knowledge tracing: modeling student proficiency using graph neural network. In *IEEE/WIC/ACM International Conference on Web Intelligence*, pages 156–163.

Newell, A., Shaw, J. C., and Simon, H. A. (1958). Elements of a theory of human problem solving. *Psychological review*, 65(3):151.

Ossiannilsson, E. (2020). Sustainability: the futures of education in the global context: sustainable distance education. *Sustainability*.

Pandey, S. and Karypis, G. (2019). A self attentive model for knowledge tracing. In *Proceedings of the 12th International Conference on Educational Data Mining (EDM)*.

Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L. J., and Sohl-Dickstein, J. (2015). Deep knowledge tracing. *Advances in Neural Information Processing Systems*, 28.

Raghu, M., Unterthiner, T., Kornblith, S., Zhang, C., and Dosovitskiy, A. (2021). Do vision transformers see like convolutional neural networks? *Advances in Neural Information Processing Systems*, 34:12116–12128.

Rumelhart, D. E., Hinton, G. E., McClelland, J. L., et al. (1986). A general framework for parallel distributed processing. *Parallel distributed processing: Explorations in the microstructure of cognition*, 1(45-76):26.

Shen, S., Huang, Z., Liu, Q., Su, Y., Wang, S., and Chen, E. (2022). 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*, pages 427–437.

Tong, S., Liu, Q., Huang, W., Hunag, Z., Chen, E., Liu, C., Ma, H., and Wang, S. (2020). Structure-based knowledge tracing: an influence propagation view. In *2020 IEEE International Conference on Data Mining*, pages 541–550. IEEE.

Tu, H., Yu, S., Saikrishna, V., Xia, F., and Verspoor, K. (2023). Deep outdated fact detection in knowledge graphs. In *IEEE International Conference on Data Mining Workshops*, pages 1443–1452, Shanghai, China. IEEE.

Van Der Maas, H. L., Dolan, C. V., Grasman, R. P., Wicherts, J. M., Huizenga, H. M., and Raijmakers, M. E. (2006). A dynamical model of general intelligence: the positive manifold of intelligence by mutualism. *Psychological review*, 113(4):842.

Wang, C., Ma, W., Zhang, M., Lv, C., Wan, F., Lin, H., Tang, T., Liu, Y., and Ma, S. (2021). Temporal cross-effects in knowledge tracing. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 517–525.

Wang, P., Zheng, W., Chen, T., and Wang, Z. (2022). Anti-oversmoothing in deep vision transformers via the fourier domain analysis: From theory to practice. In *International Conference on Learning Representations*.

Yeung, C.-K. (2019). Deep-irt: Make deep learning based knowledge tracing explainable using item response theory. *arXiv preprint arXiv:1904.11738*.

Yeung, C.-K. and Yeung, D.-Y. (2018). Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*, pages 1–10.

Yin, Y., Dai, L., Huang, Z., Shen, S., Wang, F., Liu, Q., Chen, E., and Li, X. (2023). Tracing knowledge instead of patterns: stable knowledge tracing with

diagnostic transformer. In *Proceedings of the ACM Web Conference 2023*, pages 855–864.

Zhan, B., Guo, T., Li, X., Hou, M., Liang, Q., Gao, B., Luo, W., and Liu, Z. (2024). Knowledge tracing as language processing: A large-scale autoregressive paradigm. In *International Conference on Artificial Intelligence in Education*, pages 177–191. Springer.

Zhang, J., Shi, X., King, I., and Yeung, D.-Y. (2017). Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th International Conference on World Wide Web*, pages 765–774.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Not Applicable]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes]
   (d) Information about consent from data providers/curators. [Yes]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

A Supplemental Experiments

Supplemental Experimental Settings: We introduce the datasets used in the research as well as the corresponding comparison algorithms.

First, we introduce each dataset in detail:

- **Assist2009 (AS2009)**: This dataset is made up of math questions, collected from the free online tutoring ASSISTments platform during the school year 2009-2010. Over the past decade, it has become a widely accepted benchmark for evaluating KT methods.
- **Algebra 2005-2006 (AL2005)**: This dataset is from the KDD Cup 2010 EDM Challenge that contains 13- to 14-year-old students' responses to Algebra questions. It contains detailed step-level student responses. In our experiments, we uniquely identify each question by concatenating the question name with the step name.
- **Bridge to Algebra 2006-2007 (BD2006)**: The BD2006 dataset, like the Algebra2005 dataset, comprises mathematical questions extracted from logs of students' interactions with intelligent tutoring systems. Also part of the KDD Cup 2010 EDM Challenge, BD2006 constructs unique questions in a manner similar to Algebra2005.
- **NeurIPS2020 Education Challenge (NIPS34)**: This dataset is from the Tasks 3 & 4 at the NeurIPS 2020 Education Challenge. It contains students' answers to multiple-choice diagnostic math questions and is collected from the Eedi platform.
- **EdNet**: The EdNet dataset is a large-scale hierarchical student activity dataset collected by Santa, an artificial intelligence tutoring system. It contains 131,317,236 interaction records from 784,309 students, making it one of the largest publicly available educational interaction datasets. For our experiments, we sampled interaction records from 5,000 students in the full EdNet dataset. In addition, records without KC tags were removed to ensure data quality.

Besides, we summarized the detailed information of baselines as follows:

- **DKT** (Piech et al., 2015): uses an LSTM layer to encode the student's knowledge state and predict their responses.
- **DKT+** (Yeung and Yeung, 2018): introduces prediction-consistent regularization into the DKT model to improve accuracy and stability across sequential student interactions, addressing input reconstruction and prediction waviness.
- **DKT-F** (Nagatani et al., 2019): incorporates forgetting mechanisms into KT to improve prediction accuracy by modeling memory decay over time and the frequency of skill interactions.
- **KQN** (Lee and Yeung, 2019): introduces the Knowledge Query Network, which uses neural networks to dynamically model student–skill interactions and improve both interpretability and prediction accuracy in knowledge tracing.
- **DKVMN** (Zhang et al., 2017): introduces Dynamic Key-Value Memory Networks to track and update students' concept mastery using a dual-memory architecture.
- **ATKT** (Guo et al., 2021): applies adversarial training to improve the generalization of knowledge tracing models, using an attentive LSTM structure for student performance prediction.
- **GKT** (Nakagawa et al., 2019): introduces Graph-based Knowledge Tracing, which uses graph neural networks to represent educational concepts as a dynamic graph and improve student performance prediction through structured knowledge interactions.
- **SAKT** (Pandey and Karypis, 2019): employs a self-attention mechanism to improve prediction by focusing on relevant past student interactions.

- **SAINT** (Choi et al., 2020): uses a Transformer-based architecture with separate encoder-decoder structures for questions and responses, applying deep self-attention to enhance knowledge tracing.

- **AKT** (Ghosh et al., 2020): employs a monotonic attention mechanism and Rasch model embeddings to improve both the accuracy and interpretability of learner predictions.

- **SKVMN** (Abdelrahman and Wang, 2019): utilizes Sequential Key-Value Memory Networks to improve prediction accuracy by integrating recurrent modeling with memory networks.

- **HAWKES** (Wang et al., 2021): uses the Hawkes process to model the dynamic and adaptive impact of prior interactions on learning outcomes.

- **Deep-IRT** (Yeung, 2019): combines Item Response Theory with deep learning to provide explainable predictions of student performance and item difficulty.

- **DIMKT** (Shen et al., 2022): enhances knowledge tracing by incorporating question difficulty, improving the accuracy and interpretability of student knowledge assessments.

- **AT-DKT** (Liu et al., 2023a): improves deep knowledge tracing by incorporating auxiliary tasks to refine question representations and capture individual student performance history.

- **simpleKT** (Liu et al., 2023b): uses dot-product attention to model question-specific variations and student interaction dynamics for knowledge tracing.

- **FoLiBiKT** (Im et al., 2023): enhances existing attention-based knowledge tracing models by incorporating forgetting behavior as a linear bias.

- **DTransformer** (Yin et al., 2023): adopts a Transformer-based architecture with temporal and cumulative attention to model and diagnose learner knowledge states.

- **HD-KT** (Ma et al., 2024): applies hybrid interaction denoising via sequential autoencoders and attention mechanisms to improve robustness and predictive performance.

- **extraKT** (Li et al., 2024): introduces linear bias attention to extend the effective context window, enabling robust modeling of short-term forgetting and varying interaction lengths.

Finally, we introduce implementation details. We adopt standardized experimental settings from **pyKT** Liu et al. (2022): 1) A 5-fold cross-validation is employed, with 60% for training, 20% for validation, and 20% for testing. For computational efficiency, we truncate each student's interaction sequence to 200 interactions, splitting sequences longer than 200 into multiple sequences. 2) We apply early stopping with a patience of 10 during training. Models are optimized using Adam for up to 200 epochs per hyper-parameter combination, with Bayesian search for hyper-parameter tuning. The key hyperparameters include embedding dimension, hidden state dimension, and prediction layer dimension [64, 128, 256]; learning rate [1e-3, 1e-4, 1e-5]; dropout rate [0.05, 0.1, 0.3, 0.5]; and random seed [42, 3407]. 3) To ensure robustness, each experiment is conducted over five independent runs using different random seeds. We report the mean and standard deviation of the results. Model performance is evaluated using AUC and ACC metrics. All models were implemented using PyTorch (version 1.11.0) and trained on a Linux server equipped with NVIDIA RTX 4090 (24GB) GPUs.

B Detailed Experiment Results

Table 4 and Table 3 report the experimental results in terms of ACC and AUC. Additionally, Figure 6 reports the ablation experiment results in terms of ACC. We conducted five-fold cross-validation, and the tables present the average performance across the five folds together with the corresponding standard deviations, thereby reflecting the variability of the results.

Table 3: Performance Comparisons in Terms of AUC.

Model	AUC				
	AS2009	**AL2005**	**BD2006**	**NIPS34**	**EdNet**
DKT	0.7541±0.0011	0.8149±0.0011	0.8015±0.0008	0.7689±0.0002	0.6133±0.0006
DKT+	0.7547±0.0017	0.8156±0.0011	0.8020±0.0004	0.7696±0.0002	0.6189±0.0012
DKT-F	–	0.8147±0.0013	0.7985±0.0013	0.7733±0.0003	0.6168±0.0019
KQN	0.7477±0.0011	0.8027±0.0015	0.7936±0.0014	0.7684±0.0003	0.6111±0.0022
DKVMN	0.7473±0.0006	0.8054±0.0011	0.7983±0.0009	0.7673±0.0004	0.6158±0.0022
ATKT	0.7470±0.0008	0.7995±0.0023	0.7889±0.0008	0.7665±0.0001	0.6065±0.0003
GKT	0.7424±0.0021	0.8110±0.0009	0.8046±0.0008	0.7689±0.0024	0.6223±0.0017
SAKT	0.7246±0.0017	0.7880±0.0063	0.7740±0.0008	0.7517±0.0005	0.6072±0.0018
SAINT	0.6958±0.0023	0.7775±0.0017	0.7781±0.0013	0.7873±0.0007	0.6614±0.0019
AKT	0.7853±0.0017	0.8306±0.0019	0.8208±0.0007	0.8033±0.0003	0.6721±0.0022
SKVMN	0.7332±0.0009	0.7463±0.0022	0.7287±0.0052	0.7513±0.0005	0.6182±0.0114
HAWKES	0.7224±0.0006	0.8210±0.0012	0.8068±0.0010	0.7767±0.0010	0.6837±0.0016
Deep-IRT	0.7465±0.0006	0.8040±0.0013	0.7976±0.0006	0.7672±0.0006	0.6173±0.0008
DIMKT	0.7717±0.0011	0.8277±0.0009	0.8167±0.0008	0.8030±0.0002	0.6748±0.0030
AT_DKT	0.7555±0.0005	0.8246±0.0019	0.8104±0.0009	0.7816±0.0002	0.6249±0.0020
simpleKT	0.7744±0.0018	0.8254±0.0003	0.8160±0.0006	0.8035±0.0000	0.6599±0.0027
FoLiBiKT	0.7838±0.0013	0.8316±0.0012	0.8208±0.0016	0.8033±0.0002	0.6747±0.0036
DTransformer	0.7725±0.0025	0.8188±0.0025	0.8093±0.0009	0.7944±0.0003	0.6736±0.0028
extraKT	0.7824±0.0013	0.8317±0.0021	0.8110±0.0009	**0.8045±0.0003**	0.6730±0.0025
HD-KT	0.7736±0.0016	0.8229±0.0006	0.8039±0.0014	0.7932±0.0004	**0.6992**±0.0027
ACE-KT	**0.7877±0.0007**	**0.8373±0.0008**	**0.8219±0.0011**	**0.8045±0.0004**	0.6859±0.0021

C Gradient Similarity Analysis

To quantitatively examine whether the proposed cascaded design alleviates gradient entanglement across modules, we conduct a detailed analysis of gradient similarity for the three functional modules: the *RPM* (P), the *CSM* (S), and the *CIM* (I).

During backpropagation, for each module $m \in \{P, S, I\}$, we capture the gradient of its output activation $\mathbf{z}_m$ with respect to the loss $\mathcal{L}$:

$$\mathbf{g}_m = \frac{\partial \mathcal{L}}{\partial \mathbf{z}_m} \tag{21}$$

where $\mathbf{z}_m$ denotes the module output before being passed to the subsequent stage. We then compute the pairwise cosine similarity between gradient vectors from different modules:

$$\mathrm{Sim}(m_i, m_j) = \frac{\langle \mathbf{g}_{m_i}, \mathbf{g}_{m_j} \rangle}{\|\mathbf{g}_{m_i}\| \|\mathbf{g}_{m_j}\|} \tag{22}$$

where $(m_i, m_j) \in \{(P,S), (S,I), (I,P)\}$ and $\langle \cdot, \cdot \rangle$ denotes the inner product. This metric measures the alignment between the optimization directions of two modules: values close to 1 indicate highly aligned gradients, whereas values close to 0 or negative suggest weaker alignment, orthogonality, or even conflict.

For each training epoch t, let $\mathcal{B}_t$ denote the set of mini-batches in that epoch. We randomly sample one mini-batch $b_t \in \mathcal{B}_t$ and use its gradient similarity as an estimate of the inter-module gradient relationship for that epoch:

$$\widehat{\mathrm{Sim}}^{(t)}_{m_i,m_j} = \mathrm{Sim}(m_i, m_j; b_t), \quad b_t \sim \mathrm{Uniform}(\mathcal{B}_t). \tag{23}$$

This strategy provides a representative view of the typical gradient interaction among modules during training.

Figure 7 illustrates the variation in cosine similarity of activation gradients between the three modules (RPM, CSM, and CIM) within the two stacked cascaded cognitive blocks of ACE-KT across training epochs. Several observations can be made from the figure: 1)For the vast majority of module pairs, the cosine similarity exhibits

Table 4: Performance Comparisons in Terms of ACC.

Model	ACC				
	AS2009	**AL2005**	**BD2006**	**NIPS34**	**EdNet**
DKT	0.7244±0.0014	0.8097±0.0005	0.8553±0.0002	0.7032±0.0004	0.6462±0.0028
DKT+	0.7248±0.0009	0.8097±0.0007	0.8553±0.0003	0.7039±0.0004	0.6571±0.0019
DKT-F	–	0.8090±0.0005	0.8536±0.0004	0.7076±0.0002	0.6402±0.0021
KQN	0.7228±0.0009	0.8025±0.0006	0.8532±0.0006	0.7028±0.0001	0.6422±0.0043
DKVMN	0.7199±0.0010	0.8027±0.0007	0.8545±0.0002	0.7016±0.0005	0.6444±0.0030
ATKT	0.7208±0.0009	0.7998±0.0019	0.8511±0.0004	0.7013±0.0002	0.6369±0.0009
GKT	0.7153±0.0032	0.8088±0.0008	0.8555±0.0002	0.7014±0.0028	0.6625±0.0064
SAKT	0.7063±0.0018	0.7954±0.0020	0.8461±0.0005	0.6879±0.0004	0.6391±0.0041
SAINT	0.6936±0.0034	0.7791±0.0016	0.8411±0.0065	0.7180±0.0006	0.6522±0.0024
AKT	0.7392±0.0021	0.8124±0.0011	0.8587±0.0005	0.7323±0.0005	0.6655±0.0042
SKVMN	0.7156±0.0012	0.7837±0.0023	0.8406±0.0005	0.6885±0.0005	0.6555±0.0152
HAWKES	0.7046±0.0008	0.8115±0.0009	0.8559±0.0005	0.7110±0.0007	0.6917±0.0013
Deep-IRT	0.7195±0.0004	0.8037±0.0009	0.8543±0.0003	0.7014±0.0008	0.6457±0.0033
DIMKT	0.7354±0.0019	0.8109±0.0005	0.8579±0.0001	0.7312±0.0005	0.6699±0.0038
AT_DKT	0.7250±0.0007	0.8144±0.0008	0.8560±0.0005	0.7146±0.0002	0.6512±0.0039
simpleKT	0.7320±0.0012	0.8083±0.0005	0.8579±0.0003	0.7328±0.0001	0.6557±0.0029
FoLiBiKT	0.7393±0.0010	0.8135±0.0016	0.8585±0.0008	0.7323±0.0001	0.6676±0.0028
DTransformer	0.7295±0.0017	0.8043±0.0021	0.8555±0.0007	0.7295±0.0007	0.6665±0.0017
extraKT	0.7355±0.0017	0.8110±0.0009	**0.8605±0.0012**	**0.7340±0.0004**	0.6668±0.0016
HD-KT	0.7321±0.0006	0.8120±0.0007	0.8542±0.0005	0.7237±0.0007	**0.6933±0.0006**
ACE-KT (Ours)	**0.7406±0.0009**	**0.8151±0.0003**	0.8594±0.0005	0.7332±0.0007	0.6863±0.0030

a decreasing trend as the number of epochs increases. This suggests that as training progresses, the learning directions of different modules become more independent or orthogonal, which is generally regarded as a positive signal of component specialization and decoupling. Such behavior is consistent with reduced gradient redundancy. 2) There are significant differences between blocks. In Block 0 (the shallow block), the cosine similarity among all pairs of modules generally remains at a relatively high level throughout the entire training process, suggesting that the modules in this block tend to work more collaboratively in terms of functionality. In contrast, in Block 1 (the deeper block), the cosine similarity of the CIM-RPM and RPM-CSM pairs drops sharply at an early stage (around Epoch 3 to 4) and remains extremely low thereafter. This suggests that in the deeper block and at later stages of training, these modules receive more distinct optimization signals, with gradient update directions that differ substantially. Such a high degree of decoupling likely encourages the submodules to take on more differentiated roles, thereby enhancing the model's ability to handle complex information.

In summary, as the model training progresses, the submodules of ACE-KT tend to develop stronger gradient orthogonality and functional specialization, particularly when dealing with mini-batches that may challenge inter-module coupling. The shallow block exhibits stronger global collaboration among its modules, while the deeper block shows a more pronounced trend of independent learning between modules, which may contribute to improved robustness and efficiency when tackling complex tasks. While this analysis does not constitute a formal proof, it provides supportive evidence that the modules gradually specialize and decouple as training progresses.

D Computational Complexity

The proposed ACE-KT model consists of a three-stage cascaded architecture, each designed to reflect different phases of human cognition: perception, structuring, and integration. The overall computational complexity is the sum of the costs from these modules.

RPM This stage employs a 1D convolutional layer with kernel size k for rhythm-aware pattern extraction. Given an input sequence of length T and hidden dimension H, the time complexity is:

$$\mathcal{O}(k \cdot T \cdot H^2)$$

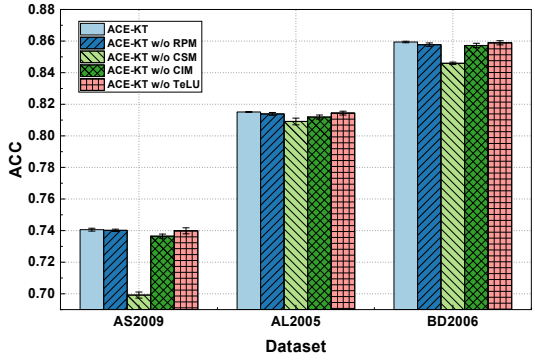

Figure 6: Ablation experiment results of ACC

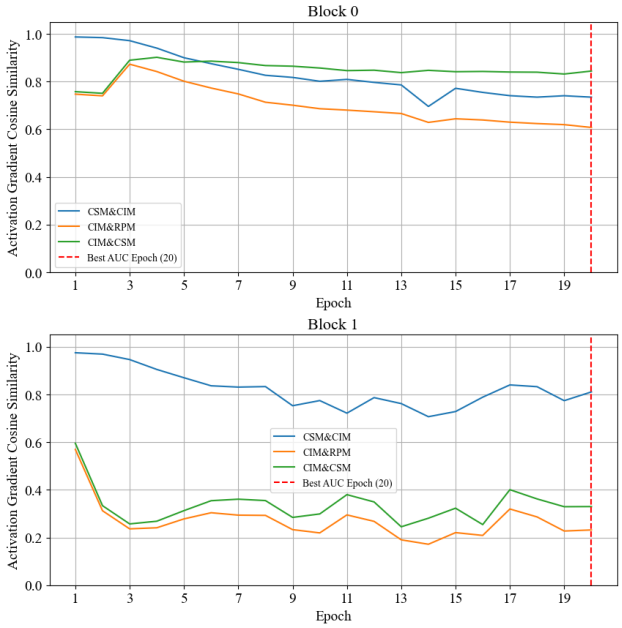

Figure 7: Evolution of randomly sampled cosine similarity of activation gradients between module pairs across epochs. Lower similarity indicates reduced gradient entanglement and improved functional specialization.

Under a standard dense Conv1d implementation, this term remains linear with respect to the sequence length T. Since k is typically small, this term introduces only moderate overhead and does not change the overall complexity order.

CSM This stage is implemented using a standard Transformer encoder. The self-attention mechanism requires pairwise interaction across the sequence, resulting in:

$$\mathcal{O}(T^2 \cdot H + T \cdot H^2). \tag{24}$$

This term dominates the overall time complexity with respect to the sequence length T. The memory complexity is also $\mathcal{O}(T^2)$ due to attention weight storage.

CIM This stage uses a selective SSM to capture long-range and temporally diffuse dependencies. The core selective scan can be computed in linear time with respect to the sequence length. However, considering the input/output projections and parameter generation layers, the overall time complexity of this stage is:

$$\mathcal{O}(T \cdot H^2), \tag{25}$$

which remains linear in T and is computationally efficient.

Overall. Combining all three modules, the total time complexity per forward pass is:

$$\mathcal{O}(T^2 \cdot H + T \cdot H^2 + k \cdot T \cdot H^2), \tag{26}$$

where the quadratic self-attention term remains dominant with respect to T. The convolutional and SSM components introduce only modest overhead. In practice, ACE-KT exhibits training and inference costs comparable to existing Transformer-based KT models, with no substantial additional asymptotic memory footprint beyond the attention module.