TRACE THE SOURCE: BEHAVIORAL DRIFT DISCRIMINATION VIA PROFILE REPLAY FOR KNOWLEDGE TRACING

Anonymous authors
Paper under double-blind review

ABSTRACT

Knowledge Tracing (KT) aims to dynamically model students' mastery of knowledge concepts from their historical learning interaction sequences. Most methods rely on item IDs and binary correctness. It makes it difficult to separate true ability changes from short-term drift, leading to biased mastery updates. To address this issue, we propose **BddKT**, a **B**ehavioral **D**rift **D**iscrimination via Profile Replay for **K**nowledge **T**racing. At each interaction, BddKT uses the LLM to build an item profile and integrates it with sequential representations. This design provides multi-source evidence for separating drift from true ability change. Moreover, we design a temporal model with a profile-replay forgetting mechanism to obtain drift-robust current states. In addition, we further introduce a profile replay calibration module that retrieves relevant historical profiles and uses gated fusion to correct drift-driven updates. Extensive experiments on four public datasets show that BddKT outperforms SOTA KT models in predictive accuracy and drift discrimination. It achieves up to 6.99% absolute improvement in AUC and 7.42% absolute improvement in ACC. Our code and experimental log can be found in the supplementary material.

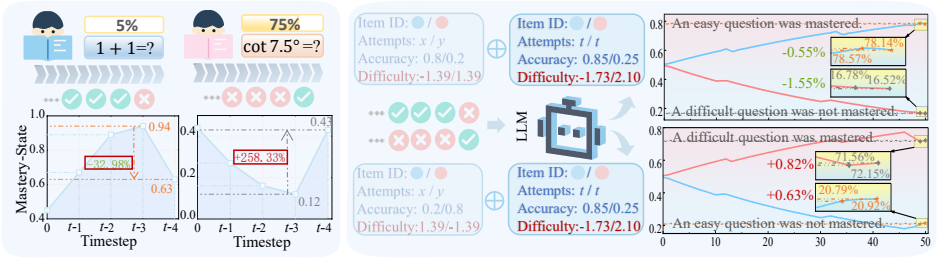
1 INTRODUCTION

Knowledge Tracing (KT) Liu et al. (2021b) is a core technique in intelligent tutoring systems and personalized education. It aims to infer a student's evolving mastery of latent knowledge concepts (KCs) from historical learning interactions and to predict future performance Corbett & Anderson (1994). As online education platforms scale up, interaction logs become larger and more diverse. Student behaviors also become more variable across time and contexts. These trends raise the requirements for KT models. Models need strong predictive accuracy, and they also need robustness to irregular responses in real learning scenarios.

KT has developed through multiple modeling lines Abdelrahman et al. (2023). Early probabilistic models Yudelson et al. (2013); Liu et al. (2021a); Pelánek (2017) represent mastery as latent states and update them with structured transitions. Logistic and feature-based methods Cen et al. (2006); Pavlik et al. (2009) add interpretable signals, such as attempts and item difficulty. With deep learning Piech et al. (2015); Yeung & Yeung (2018); Nagatani et al. (2019), recurrent models and attention-based models improve sequence modeling and capture long-range dependencies Hochreiter & Schmidhuber (1997). Memory-augmented methods Zhang et al. (2017); Abdelrahman & Wang (2019) introduce external memory to store and update knowledge states. Later, attention-based models Pandey & Karypis (2019); Ghosh et al. (2020); Choi et al. (2020) proved highly effective in capturing global dependencies within sequences. Graph-based methods Nakagawa et al. (2019); Park et al. (2024) model relations among concepts with graph structures. Despite these advances, most methods still rely mainly on item identifiers and binary correctness feedback. They often treat short-term outcomes as direct evidence of mastery change, which does not always hold in practice.

In real learning trajectories, student responses often show behavioral drift. Responses can deviate from true mastery due to slips, guesses, or short-term strategy changes. This drift creates ambiguity in mastery estimation. Similar response patterns can have different causes and different meanings. For example, an incorrect response may reflect non-mastery for one student but a careless slip for

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) Behavioral Drift Phenomenon

(b) Discrimination via Profile Replay

Figure 1: Behavioral drift in knowledge tracing and our discrimination approach. (a) Behavioral Drift Phenomenon: An atypical response outcome that occurs within an otherwise stable response sequence. (b) Discrimination via Profile Replay: The profile replay is used to discriminate against behavioral drift.

another. A correct response on a hard item may come from guessing rather than true mastery gain. Item difficulty further sharpens this ambiguity. The same correctness feedback can carry different meanings on easy and hard items. If a model only uses item IDs and binary correctness or short-window result Shen et al. (2020), these cases can look similar in the input. The model may then update mastery in the wrong direction and produce biased mastery-state prediction. As shown in Figure 1(a).

To address this problem, we propose BddKT, a profile replay framework for behavioral drift discrimination in KT. As shown in Figure 1(b), it combines Qwen2.5-7B-Instruct-generated item descriptions with item-level statistics and sequential representations to model a current state robust to transient perturbations. A profile replay calibrator then retrieves relevant historical profiles via multi-head attention and uses gated fusion to refine this state. By suppressing drift-driven fluctuations while preserving true mastery transitions, BddKT produces more faithful mastery trajectories and reliable predictions. Our main contributions are summarized as follows:

- We develop an item profiling scheme that integrates Qwen2.5-7B-Instruct-generated semantic descriptions with item-level statistics and explicitly models item difficulty as a critical cue for distinguishing easy-slip and hard-guess behaviors under similar short-horizon response patterns.
- We design a temporal modeling module with a profile-replay forgetting mechanism and an attention-based replay calibration module with gated fusion to adaptively correct mastery updates under drift.
- We conduct extensive experiments on four public benchmark datasets, demonstrating that BddKT consistently outperforms the best 15 KT baselines in performance. An average growth rate of 3.6225% AUC improvement and 3.3825% ACC improvement.

2 Related Work

Knowledge Tracing (KT) aims to dynamically estimate students' mastery states over knowledge concepts (KCs) from their historical interaction sequences and to predict future performance. Prior research has evolved from early probabilistic formulations and logistic-regression paradigms Pelánek (2017) to deep sequential modeling frameworks Song et al. (2022), substantially improving representational capacity and predictive accuracy. Meanwhile, short-term fluctuations in learning behaviors have made robustness and interpretable state-update mechanisms central challenges for KT models.

Sequential Dependency Models DKT Piech et al. (2015) use recurrent structures to capture long-range dependencies and track how latent knowledge states (KS) evolve over time. ATKT Ghosh et al. (2020) adds abnormal perturbations during training to improve stability under irregular behaviors. However, many RNN-based models Chen et al. (2023); Bai et al. (2025) update states mainly based on the current response. When short-term responses fluctuate, the model may treat them as true mastery changes. This can bias mastery-state prediction.

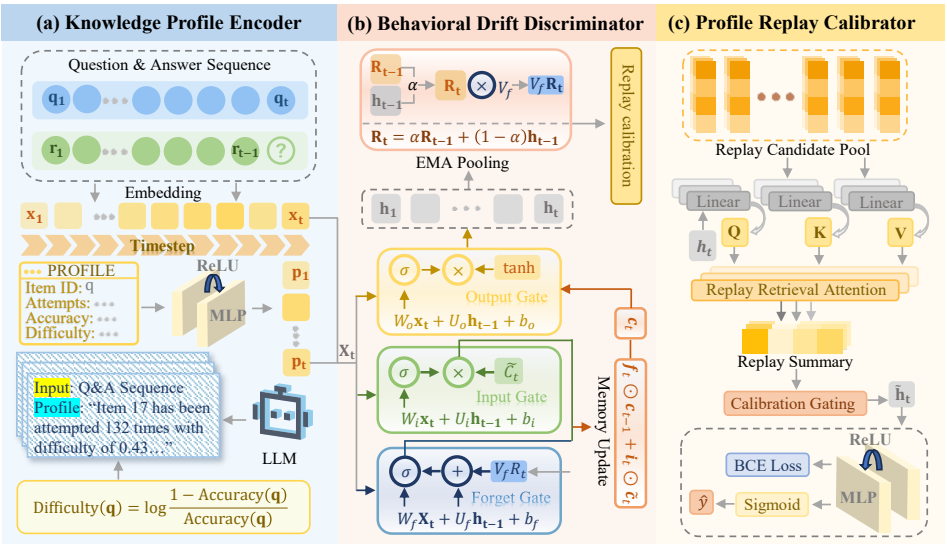


Figure 2: The framework of BddKT. (a) Knowledge Profile Encoder: Encodes students' interaction sequences and constructs item profiles with Qwen2.5-7B-Instruct-generated descriptions and item-level statistics. (b) Behavioral Drift Discriminator: Models the temporal evolution of student states and maintains drift-robust current representations. (c) Profile Replay Calibrator: Retrieves historical profile evidence via multi-head attention and applies gated calibration to fuse the current state with replay summaries for response prediction.

Memory-Enhanced Models To strengthen structured representations of KCs, memory-augmented models such as DKVMN Zhang et al. (2017) and SKVMN Abdelrahman & Wang (2019) introduce an external memory. They store KS in a readable and writable key–value structure and update it with each interaction. This design supports a finer-grained linkage. However, under short-term fluctuations, these can still be overly driven by a single observation if they lack additional discriminative evidence, like difficulty or historical performance summaries.

GNN-Based Models Graph neural networks (GNNs) organize KCs as a graph. They model prerequisite and associative relations between concepts explicitly. GKT Nakagawa et al. (2019) is a representative method in this line. Recent studies, such as DYGKT Cheng et al. (2024) and DGEKT Cui et al. (2024), further enrich graph structures and refine relation modeling. They aim to improve cross-concept transfer and structural generalization. However, stronger graph modeling does not directly solve semantic discrimination under behavioral drift.

Long-Range Dependency Models SAKT Pandey & Karypis (2019) introduces the Transformer architecture to KT. Subsequent variants Pandey & Srivastava (2020); Wang et al. (2021); Yin et al. (2023), including StableKT Li et al. (2024a) and KeenKT Li et al. (2025), improve attention stability under response fluctuations. AKT Ghosh et al. (2020) demonstrates the sequence modeling capacity of multi-head self-attention (MSA), while SimpleKT Liu et al. (2023b) retains competitive performance with a simpler attention pipeline. RobustKT Guo et al. (2025), UKT Cheng et al. (2025), and FlucKT Hou et al. (2025) address noisy or drifting behaviors, but lack explicit evidence to distinguish slips from guesses.

Existing KT methods have made progress in sequence modeling Yeung & Yeung (2018); Nagatani et al. (2019), structure modeling Im et al. (2023); Li et al. (2024b), and robust training Long et al. (2021); Shen et al. (2021); Liu et al. (2023a). However, most methods still treat the KS as a latent variable and update it from single-step observations. When short-term anomalies occur, they can misinterpret true mastery changes and produce biased updates.

3 METHODOLOGY

3.1 PROBLEM FORMULATION

Knowledge Tracing (KT) predicts future responses from a student's interaction sequence $\{(q_t, r_t)\}_{t=1}^{T}$, where q_t is the item ID, $r_t \in \{0, 1\}$ indicates an incorrect or correct response, and T is the sequence length. Given the history $\mathbf{S_{t-1}} = \{(q_i, r_i)\}_{i=1}^{t-1}$ and current item q_t, the model predicts:

$$\hat{y}_t = \Pr(r_t = 1 \mid \mathbf{S_{t-1}}, q_t), \tag{1}$$

where $\hat{y}_t$ is the probability of a correct response.

To improve robustness to short-term response fluctuations, we use an item profile as auxiliary evidence at each step. The model is trained with binary cross-entropy:

$$\mathcal{L}_{\mathrm{BCE}} = -\sum_{t=1}^{T} \left[r_t \log \hat{y}_t + (1 - r_t) \log(1 - \hat{y}_t) \right], \tag{2}$$

where $\mathcal{L}_{\mathrm{BCE}}$ is the training objective.

3.2 KNOWLEDGE PROFILE ENCODER

As shown in Figure 2(a), this module combines an item profile with the interaction embedding to provide input for drift discrimination and calibration. At step t, the student interacts with item q_t and receives binary feedback $r_t \in \{0, 1\}$. We sum the item, response, and timestep embeddings to obtain $\mathbf{x}_t \in \mathbb{R}^d$:

$$\mathbf{x}_t = \mathbf{E}_q(q_t) + \mathbf{E}_r(r_t) + \mathbf{E}_t(t). \tag{3}$$

For each item q, we compute attempts $A(q)$ and correct attempts $C(q)$ from the training data. Its accuracy and log-odds difficulty are:

$$\mathrm{Accuracy}(q) = \frac{C(q)}{\max(A(q), 1)}, \tag{4}$$

$$\text{Difficulty}(q) = \log \frac{1 - \mathrm{Accuracy}(q)}{\mathrm{Accuracy}(q)}. \tag{5}$$

To add semantic information, we convert these statistics and the item ID into text and encode it with Qwen2.5-7B-Instruct, yielding $\mathbf{z}_q \in \mathbb{R}^{d_{\text{text}}}$:

$$\mathbf{z}_q = \mathrm{LLM}(\mathrm{Text}(q, A(q), \mathrm{Accuracy}(q), \text{Difficulty}(q))). \tag{6}$$

We project the cached item vector into the KT hidden space to obtain $\mathbf{p}_t \in \mathbb{R}^d$:

$$\mathbf{p}_t = \mathbf{W}_p \mathbf{z}_{q_t} + \mathbf{b}_p, \tag{7}$$

where $\mathbf{W}_p$ and $\mathbf{b}_p$ are the projection parameters. Thus, online profile processing requires only lookup and projection.

Finally, we fuse $\mathbf{x}_t$ and $\mathbf{p}_t$ into the input $\mathbf{X}_t \in \mathbb{R}^d$ for temporal modeling and drift discrimination:

$$\mathbf{X}_t = \mathrm{ReLU}\left(\mathbf{W}_x[\mathbf{x}_t; \mathbf{p}_t] + \mathbf{b}_x\right), \tag{8}$$

where $\mathbf{W}_x$ and $\mathbf{b}_x$ are the fusion parameters.

3.3 BEHAVIORAL DRIFT DISCRIMINATOR

This module takes the fused representation $\mathbf{X}_t$ from the knowledge profile encoder and introduces a profile-replay forgetting mechanism, so that short-term anomalies do not dominate the state update. See Figure 2(b).

Temporal State Update At step t, we update the hidden state with a gated recurrent structure. We denote the previous hidden state by $\mathbf{h}_{t-1}$, the previous cell state by $\mathbf{c}_{t-1}$, and the forget gate by $\mathbf{f}_t$, the input gate by $\mathbf{i}_t$, and the output gate by $\mathbf{o}_t$. The update is:

$$\begin{cases} \mathbf{i}_t = \sigma(\mathbf{W}_i \mathbf{X}_t + \mathbf{U}_i \mathbf{h}_{t-1} + \mathbf{b}_i) \\ \mathbf{o}_t = \sigma(\mathbf{W}_o \mathbf{X}_t + \mathbf{U}_o \mathbf{h}_{t-1} + \mathbf{b}_o) \\ \tilde{\mathbf{c}}_t = \tanh(\mathbf{W}_c \mathbf{X}_t + \mathbf{U}_c \mathbf{h}_{t-1} + \mathbf{b}_c) \\ \mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t \end{cases}, \tag{9}$$

where $\tilde{c}_t$ is the candidate cell state. Then We obtain the hidden state at the current time step:

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t). \tag{10}$$

EMA Replay Memory To provide stable historical evidence, we maintain an exponential moving average replay memory vector $\mathbf{R}_t$. We update $\mathbf{R}_t$ using the previous hidden state $\mathbf{h}_{t-1}$:

$$\mathbf{R}_t = \alpha \mathbf{R}_{t-1} + (1 - \alpha)\mathbf{h}_{t-1}, \tag{11}$$

where α is the update rate that controls how much history is retained, and we provide detailed settings for it in Section 4.4.

Profile-Replay Forgetting Mechanism We inject replay memory into the forget gate so that forgetting is conditioned on historical evidence. We define the forget gate $\mathbf{f}_t$ as

$$\mathbf{f}_t = \sigma(\mathbf{W}_f \mathbf{X}_t + \mathbf{U}_f \mathbf{h}_{t-1} + \mathbf{V}_f \mathbf{R}_t + \mathbf{b}_f), \tag{12}$$

where $\mathbf{v}_f$ is a learnable projection matrix. This design lets the model consult the replay memory $\mathbf{R}_t$ when short-term responses deviate from the typical pattern, which stabilizes forgetting and state updates.

3.4 Profile Replay Calibrator

This module takes the current state $\mathbf{h}_t$ and forms a replay summary $\mathbf{s}_t$ from the replay candidate pool. It then fuses $\mathbf{h}_t$ and $\mathbf{s}_t$ to obtain the calibrated representation $\tilde{\mathbf{h}}_t$, which is used for prediction. As shown in Figure 2(c).

Replay Candidate Pool Representation At step t, the replay candidate pool is denoted by $\mathcal{M}_t$. It stores historical profile vectors $\mathbf{p}_i$, see Eq.(7). We write the pool as a matrix:

$$\mathbf{M}_t = [\mathbf{p}_{i_1}; \mathbf{p}_{i_2}; \ldots; \mathbf{p}_{i_m}], \tag{13}$$

where m is the replay depth, and we provide detailed settings for these values in Section 4.4.

Multi-Head Replay Retrieval Attention Let the number of heads be H. For head h, we build the query $\mathbf{q}_t^{(h)}$ from the current state and build keys and values from the pool:

$$\begin{cases} \mathbf{q}_t^{(h)} = \mathbf{W}_Q^{(h)} \mathbf{h}_t \\ \mathbf{K}_t^{(h)} = \mathbf{M}_t (\mathbf{W}_K^{(h)})^\top \\ \mathbf{V}_t^{(h)} = \mathbf{M}_t (\mathbf{W}_V^{(h)})^\top \end{cases}, \tag{14}$$

where $\tilde{c}_t$ is the candidate cell state. Then We obtain the hidden state at the current time step:

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t), \tag{15}$$

where $\mathbf{w}_Q^{(h)}$, $\mathbf{w}_K^{(h)}$ and $\mathbf{w}_V^{(h)}$ are learnable matrices. For candidate index i_j, the attention score is:

$$e_{t,j}^{(h)} = \frac{\langle \mathbf{q}_t^{(h)}, \mathbf{k}_{t,j}^{(h)} \rangle}{\sqrt{d_k}}, \tag{16}$$

where $\mathbf{k}_{t,j}^{(h)}$ is the j-th row of $\mathrm{K}_t^{(h)}$. The attention weight is computed by softmax:

$$\alpha_{t,j}^{(h)} = \frac{\exp\left(e_{t,j}^{(h)}/\tau\right)}{\sum_{\ell=1}^{m} \exp\left(e_{t,\ell}^{(h)}/\tau\right)}, \tag{17}$$

Algorithm 1: BddKT Inference Procedure.

Input: Interaction sequence $\{(q_t, r_t)\}_{t=1}^{T}$; replay depth m; EMA rate α; temperature τ
Output: Predictions $\{\hat{y}_t\}_{t=1}^{T}$

Initialize $\mathbf{h}_0, \mathbf{c}_0, \mathbf{R}_0 \leftarrow \mathbf{0}$, and $\mathcal{M}_1 \leftarrow \emptyset$
for $t \leftarrow 1$ **to** T **do**
  // Knowledge Profile Encoder
  $\mathbf{x}_t \leftarrow \mathbf{E}_q(q_t) + \mathbf{E}_r(r_t) + \mathbf{E}_t(t)$
  $\mathbf{p}_t \leftarrow \mathbf{W}_p \mathbf{z}_{q_t} + \mathbf{b}_p$
  $\mathbf{X}_t \leftarrow \mathrm{Fuse}([\mathbf{x}_t; \mathbf{p}_t])$
  // Behavioral Drift Discriminator
  $\mathbf{R}_t \leftarrow \alpha \mathbf{R}_{t-1} + (1-\alpha)\mathbf{h}_{t-1}$
  $(\mathbf{h}_t, \mathbf{c}_t) \leftarrow \mathrm{TempUpdate}(\mathbf{X}_t, \mathbf{h}_{t-1}, \mathbf{c}_{t-1}, \mathbf{R}_t)$
  // Profile Replay Calibrator
  $\mathcal{M}_t \leftarrow \mathcal{M}_{t-1} \cup \{\mathbf{p}_{t-1}\}$
  $\mathbf{M}_t \leftarrow \mathrm{BuildPool}(\mathcal{M}_t, m)$
  $\mathbf{s}_t \leftarrow \mathrm{ReplayAttn}(\mathbf{h}_t, \mathbf{M}_t; H, \tau)$
  $\tilde{\mathbf{h}}_t \leftarrow \mathrm{GateFuse}(\mathbf{h}_t, \mathbf{s}_t)$
  $\hat{y}_t \leftarrow \sigma\big(\mathrm{PredMLP}(\tilde{\mathbf{h}}_t)\big)$
return $\{\hat{y}_t\}_{t=1}^{T}$

where $\tau > 0$ is the temperature parameter. We provide detailed settings for these values in Section 4.4. The head-specific replay summary is:

$$\mathbf{s}_t^{(h)} = \sum_{j=1}^{m} \alpha_{t,j}^{(h)} \mathbf{v}_{t,j}^{(h)}, \tag{18}$$

where $\mathbf{v}_{t,j}^{(h)}$ is the j-th row of $\mathrm{V}_t^{(h)}$. We concatenate all heads and apply an output projection:

$$\mathbf{s}_t = \mathbf{W}_O[\mathbf{s}_t^{(1)}; \mathbf{s}_t^{(2)}; \ldots; \mathbf{s}_t^{(H)}], \tag{19}$$

where $\mathbf{W}_O$ is the output projection matrix.

Gated Calibration Fusion. We implement the calibration gate $\mathbf{g}_t$ with a perceptron:

$$\begin{cases} \mathbf{u}_t = \mathrm{ReLU}\left(\mathbf{W}_{g1}\left[\mathbf{h}_t; \mathbf{s}_t\right] + \mathbf{b}_{g1}\right) \\ \mathbf{g}_t = \sigma(\mathbf{W}_{g2}\mathbf{u}_t + \mathbf{b}_{g2}) \end{cases}, \tag{20}$$

where $\mathbf{W}_{g1}, \mathbf{W}_{g2}$ and $\mathbf{b}_{g1}, \mathbf{b}_{g2}$ are learnable parameters.

The gate balances the current state $\mathbf{h}_t$ and the replay summary $\mathbf{s}_t$ to form the calibrated representation:

$$\tilde{\mathbf{h}}_t = \mathbf{g}_t \odot \mathbf{h}_t + (1 - \mathbf{g}_t) \odot \mathbf{s}_t, \tag{21}$$

where $\odot$ denotes element-wise multiplication. Each dimension of $\mathbf{g}_t$ determines how much of the current state is retained, while its complement determines the contribution of historical profile evidence. This allows calibration to vary across state dimensions and interactions.

We pass $\tilde{\mathbf{h}}_t$ through an MLP to obtain a prediction logit:

$$\begin{cases} \mathbf{z}_t = \mathrm{ReLU}(\mathbf{W}_1 \tilde{\mathbf{h}}_t + \mathbf{b}_1) \\ o_t = \mathbf{w}_2^\top \mathbf{z}_t + b_2 \end{cases}, \tag{22}$$

where $\mathbf{W}_1$, $\mathbf{b}_1$, $\mathbf{w}_2$, and b_2 are learnable parameters. The logit is then mapped to the predicted correctness probability:

$$\hat{y}_t = \sigma(o_t), \tag{23}$$

where $\sigma(\cdot)$ denotes the sigmoid function.

Algorithm 1 summarizes the BddKT inference procedure. By incorporating retrieved historical profiles into the current state before prediction, the calibration module can reduce the effect of an isolated response fluctuation while retaining information from the ongoing interaction sequence. Its effectiveness in distinguishing transient drift from mastery changes is evaluated in the experiments.

4 EXPERIMENT

4.1 EXPERIMENTAL SETTINGS

Datasets In this study, we evaluate the performance of the proposed BddKT model on four widely used public datasets for KT, including ASSISTments2009 (AS2009)[1], ASSISTments2015 (AS2015)[2], Algebra2008 (AL2008)[3], and Junyi[4]. The key statistics of these datasets are summarized in Table 1.

Table 1: Statistics of all datasets.

Datasets	Statistics			
	#Students	#Exercises	#KCs	#Records
AS2009	4,163	17,751	123	346,860
AS2015	19,917	948	100	708,631
AL2008	3,310	2,979	628	8,918,054
Junyi	247,606	722	41	25,925,922

Table 2: AUC and ACC performance of BddKT and baselines. **Boldface** denotes the best performance, underlining denotes the second-best.

Model	Venue	Backbone	Difficulty Modeling	AS2009		AS2015		AL2008		Junyi	
				AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC
DKT	NIPS'15	LSTM	✗	0.8226	0.7677	0.7276	0.7014	0.7781	0.8423	0.8492	0.8133
ATKT	MM'21		✗	0.7496	0.7214	0.7231	0.7492	0.7812	0.8327	0.7832	0.7869
DKVMN	WWW'17	KVMN	✗	0.7479	0.7329	0.7264	0.7528	0.7415	0.8455	0.8126	0.7994
SKVMN	SIGIR'19		✗	0.7517	0.7366	0.7491	0.7611	0.7650	0.5491	0.8274	0.7942
GKT	WI'19	GNN	✗	0.8063	0.7528	0.7014	0.7628	0.7528	0.8536	0.8311	0.8009
DYGKT	KDD'24		✓	0.8129	0.7843	0.7139	0.7677	0.7631	0.8657	0.8427	0.8136
DGEKT	TOIS'24		✗	0.8011	0.7871	0.7192	0.7459	0.7654	0.8319	0.8442	0.7866
SAKT	EDM'19	Transformer	✗	0.7753	0.7042	0.7128	0.7479	0.7488	0.8461	0.8170	0.7719
StableKT	WWW'24		✓	0.8511	0.7821	0.7527	0.7798	0.8382	0.8553	0.8521	0.8399
KeenKT	AAAI'26		✗	0.8613	0.7881	0.8181	0.7801	0.8918	0.8613	0.9017	0.8339
AKT	KDD'20	MSA	✓	0.8461	0.7793	0.7282	0.7581	0.7810	0.8477	0.8529	0.7816
SimpleKT	ICLR'23		✓	0.8462	0.7729	0.7247	0.7502	0.7788	0.8418	0.8459	0.7731
RobustKT	WWW'25		✗	0.7133	0.7631	0.7806	0.7660	0.8761	0.8593	0.8897	0.8377
UKT	AAAI'25		✗	0.8514	0.7809	0.7269	0.7482	0.8008	0.8133	0.8249	0.8012
FlucKT	AAAI'25		✗	0.8262	0.7537	0.7528	0.7622	0.8429	0.8392	0.8621	0.8293
BddKT	Ours	MSA	✓	**0.8843**	**0.8045**	**0.8263**	**0.7953**	**0.9541**	**0.9299**	**0.9362**	**0.8574**

Baseline Models To comprehensively assess BddKT, we compare it with representative KT models grouped by backbone families into five groups: (1) LSTM-based models: DKT, ATKT; (2) Memory-augmented models: DKVMN, SKVMN; (3) GNN-based models: GKT, DYGKT, DGEKT; (4) Transformer-based models: SAKT, StableKT, KeenKT; (5) MSA-based models: AKT, SimpleKT, RobustKT, UKT, FlucKT.

Parameter Settings For BddKT, we set the question embedding dimension to 50, the interaction embedding dimension to 200, and the hidden dimension of the final prediction module to 100. We use a batch size of 128 and truncate each student sequence to a maximum length of 200. We train the model with Adagrad. For all experiments, we set the learning rate to 1e-2, which was found to perform best among le-1, le-2, and le-3. We adopt a student-level five-fold cross-validation for four datasets and train for at most 100 epochs.

4.2 PERFORMANCE COMPARISON

As shown in Table 2, all results reported in the table are averaged over five-fold cross-validation. BddKT achieves the best overall performance on all four datasets in terms of AUC and ACC. In AUC, on AL2008, BddKT improves AUC from 0.8918 to 0.9541, which corresponds to a 6.99% relative gain over the strongest baseline. On Junyi, BddKT increases AUC from 0.9017 to 0.9362, yielding a 3.83% relative gain. BddKT also improves AUC on AS2009 and AS2015, reaching 0.8843 and

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

[2]https://sites.google.com/site/assistmentsdata/datasets/2015-assistments-skill-builder-data

[3]https://pslcdatashop.web.cmu.edu/KDDCup/downloads.jsp

[4]https://pslcdatashop.web.cmu.edu/Project

Table 3: Ablation study on key components of BddKT.

metrics	Models	AS2009	AS2015	AL2008	Junyi
AUC	w/o Profile	0.8216	0.7274	0.6764	0.8480
	w/o EMA	0.8517	0.7963	0.7477	0.8822
	w/o RC	0.8303	0.7454	0.6901	0.8543
	BddKT	**0.8843**	**0.8263**	**0.9541**	**0.9362**
ACC	w/o Profile	0.7671	0.7022	0.8437	0.8141
	w/o EMA	0.7908	0.7782	0.9167	0.8482
	w/o RC	0.7776	0.7209	0.8643	0.8279
	BddKT	**0.8045**	**0.7953**	**0.9299**	**0.8574**

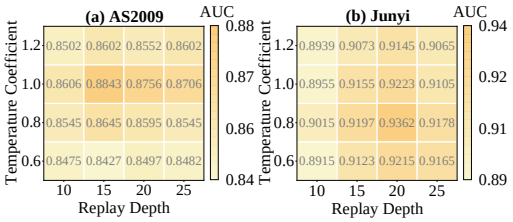


Figure 4: The hyper-parameter analysis on the replay depth and Temperature Coefficient in (a) AS2009 and (b) Junyi.

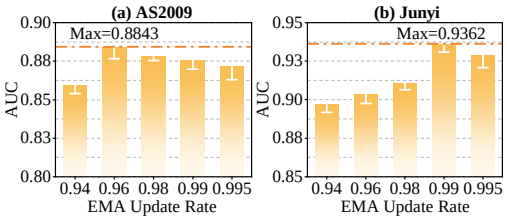


Figure 3: A hyper-parameter analysis on the EMA update rate of BddKT in (a) AS2009 and (b) Junyi.

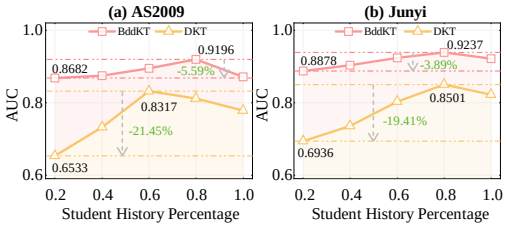


Figure 5: Temporal robustness under different student history percentages on (a) AS2009 and (b) Junyi.

0.8263, respectively. In ACC, on AL2008, improving ACC from 0.8657 to 0.9299, which gives a 7.42% relative improvement. These results support the effectiveness of profile replay and calibration for drift discrimination and robust temporal prediction.

4.3 Ablation Experiment

To evaluate the contribution of each key component in BddKT, we design three ablated variants: (1) w/o Profile: removing the Qwen2.5-7B-Instruct-based profile evidence; (2) w/o EMA: removing the EMA-based replay memory update; (3) w/o RC: removing the replay calibration (RC) module.

Table 3 shows that all components are necessary. Removing the profile evidence causes the largest degradation overall. The AUC drops on all four datasets from 6.98% to 28.9% relative declines. Removing EMA also consistently hurts performance, with AUC drops from 0.03 to 0.2064. This confirms that replay memory smoothing stabilizes state updates. Removing RC leads to substantial drops from 0.054 to 0.264, indicating that retrieval-based replay evidence and calibration are central to robust prediction under behavioral drift.

4.4 Hyperparameter Analysis

EMA update rate Figure 3 shows the effect of the EMA update rate α on AS2009 and Junyi; error bars indicate variability across random seeds. BddKT achieves its highest AUC at $\alpha = 0.96$ on AS2009 (0.8843) and $\alpha = 0.99$ on Junyi (0.9362). A smaller α makes replay memory more responsive, which may benefit AS2009's sparser histories, whereas a larger α smooths updates across Junyi's richer, more repetitive traces. These explanations are consistent with the observed optima but require further analysis to confirm.

Replay Depth and Temperature Figure 4 shows that AS2009 achieves its highest AUC of 0.8843 at $m = 15, \tau = 1.0$, whereas Junyi reaches 0.9362 at $m = 20, \tau = 0.8$. AS2009's sparser histories may favor a moderate candidate pool and a higher temperature, limiting weak matches and overreliance on a single profile. Junyi's richer, more repetitive traces may support a larger pool and a lower temperature that emphasizes stronger replay cues.

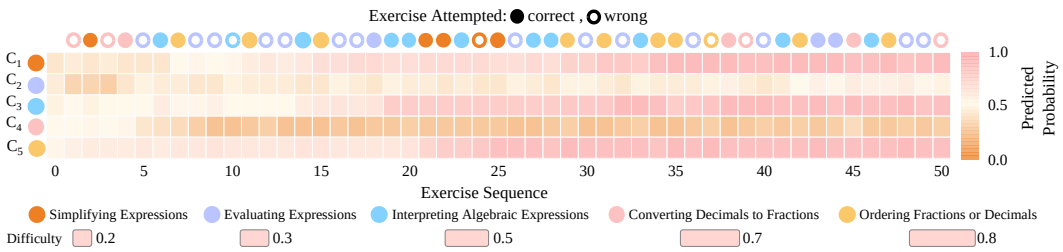


Figure 6: The predicted results of Student #1 on five concepts with different difficulty levels over 50 time steps from AS2009.

4.5 Temporal Robustness Analysis

To assess sensitivity to limited interaction history, we evaluate temporal robustness at different history ratios on AS2009 and Junyi, shown in Figure 5. DKT's AUC falls from 0.8317 to 0.6533 on AS2009 (21.45%) and from 0.8501 to 0.6936 on Junyi (19.41%). BddKT shows smaller relative drops: from 0.9196 to 0.8682 on AS2009 (5.59%) and from 0.9237 to 0.8878 on Junyi (3.89%). These results suggest that BddKT is less sensitive to limited interaction history.

4.6 Temporal Robustness Analysis

To assess BddKT's drift discrimination and mastery calibration, we visualize predicted correctness probabilities over 50 interactions. In Figure 6, the student answers concept C_5 correctly at steps 7, 11, 29, 31, 34, and 35, then incorrectly at step 37. With historical profiles and item difficulty, the predicted probability changes little, suggesting that BddKT treats the error as a short-term slip-type drift. Correct responses at steps 42 and 47 are consistent with this interpretation, illustrating stable drift discrimination and calibration in this case.

4.7 Case Study

As shown in Figure 7, we study two representative scenarios. In the left case, the boy makes one mistake after consecutive correct responses. BddKT performs profile replay, prioritizes replay evidence, and identifies a short-term slip-type drift. In the right case, the girl answers correctly after consecutive incorrect responses. BddKT similarly detects a short-term guess-type drift and calibrates the mastery update. These cases indicate that BddKT can discriminate drift and produce more stable predictions.

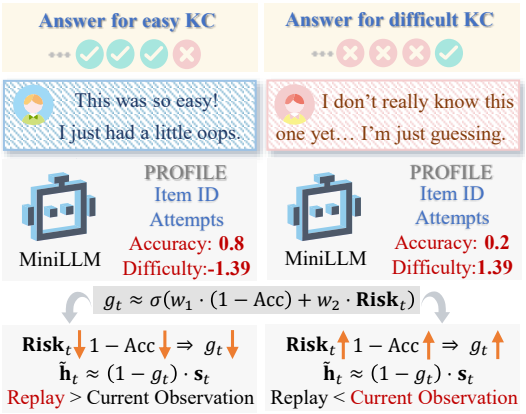


Figure 7: Case examples of BddKT.

5 Conclusion

This paper proposed BddKT, a knowledge tracing framework for behavioral drift discrimination via profile replay. BddKT encodes Qwen2.5-7B-Instruct-based item profiles with difficulty-related statistics, learns a current state via a replay-forgetting temporal module, and then retrieves historical profiles to form a replay summary for gated calibration. However, BddKT still relies on the quality of generated item profiles and available difficulty-related statistics, and it does not explicitly incorporate richer behavioral signals such as response time, hints, or affective states. Future work will extend profile construction with richer learning signals and improve the efficiency of profile replay for real-world deployment.

AI USE STATEMENT

We used Qwen2.5-7B-Instruct for item-profile construction in BddKT, as described in Section 3.2. This process uses textual representations of item identifiers and training-data statistics, including attempt counts, accuracy, and difficulty, to obtain semantic profile representations for knowledge tracing. These representations are model inputs rather than independent evidence of students' cognitive states. We also used ChatGPT to assist with English editing, and language polishing. The authors take responsibility for the final manuscript, including its methods, experimental results, interpretations, and AI-assisted content.

REPRODUCIBILITY STATEMENT

Section 3 presents the model formulation and inference procedure, including item-profile construction, temporal modeling, and replay calibration, with pseudocode provided in Algorithm 1. Section 4.1 identifies the benchmark datasets and reports the main hyperparameters, optimizer, training budget, and student-level five-fold cross-validation protocol. Item-profile statistics are computed from training data, as described in Section 3.2. The supplementary material provides the implementation and experimental logs to support reproduction of the reported results.

REFERENCES

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

Ghodai Abdelrahman, Qing Wang, and Bernardo Nunes. Knowledge tracing: A survey. *ACM Comput. Surv.*, 55(11):1–37, 2023.

Youheng Bai, Xueyi Li, Zitao Liu, Yaying Huang, Mi Tian, and Weiqi Luo. Rethinking and improving student learning and forgetting processes for attention based knowledge tracing models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 27822–27830, 2025.

Hao Cen, Kenneth Koedinger, and Brian Junker. Learning factors analysis–a general method for cognitive model evaluation and improvement. In *Proceedings of the International conference on intelligent tutoring systems*, pp. 164–175, 2006.

Jiahao Chen, Zitao Liu, Shuyan Huang, Qiongqiong Liu, and Weiqi Luo. Improving interpretability of deep sequential knowledge tracing models with question-centric cognitive representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 14196–14204, 2023.

Ke Cheng, Linzhi Peng, Pengyang Wang, Junchen Ye, Leilei Sun, and Bowen Du. Dygkt: Dynamic graph learning for knowledge tracing. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 409–420, 2024.

Weihua Cheng, Hanwen Du, Chunxiao Li, Ersheng Ni, Liangdi Tan, Tianqi Xu, and Yongxin Ni. Uncertainty-aware knowledge tracing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 27905–27913, 2025.

Youngduck Choi, Youngnam Lee, Junghyun Cho, Jineon Baek, Byungsoo Kim, Yeongmin Cha, Dongmin Shin, Chan Bae, and Jaewe Heo. Towards an appropriate query, key, and value computation for knowledge tracing. In *Proceedings of the seventh ACM conference on learning @ scale*, pp. 341–344, 2020.

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

Chaoran Cui, Yumo Yao, Chunyun Zhang, Hebo Ma, Yuling Ma, Zhaochun Ren, Chen Zhang, and James Ko. Dgekt: a dual graph ensemble learning method for knowledge tracing. *ACM Transactions on Information Systems*, 42(3):1–24, 2024.

Aritra Ghosh, Neil Heffernan, and Andrew S Lan. Context-aware attentive knowledge tracing. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2330–2339, 2020.

Teng Guo, Yu Qin, Yubin Xia, Mingliang Hou, Zitao Liu, Feng Xia, and Weiqi Luo. Enhancing knowledge tracing through decoupling cognitive pattern from error-prone data. In *Proceedings of the ACM on Web Conference*, pp. 5108–5116, 2025.

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

Mingliang Hou, Xueyi Li, Teng Guo, Zitao Liu, Mi Tian, Renqiang Luo, and Weiqi Luo. Cognitive fluctuations enhanced attention network for knowledge tracing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 14265–14273, 2025.

Yoonjin Im, Eunseong Choi, Heejin Kook, and Jongwuk Lee. Forgetting-aware linear bias for attentive knowledge tracing. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, pp. 3958–3962, 2023.

Xueyi Li, Youheng Bai, Teng Guo, Zitao Liu, Yaying Huang, Xiangyu Zhao, Feng Xia, Weiqi Luo, and Jian Weng. Enhancing length generalization for attention based knowledge tracing models with linear biases. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pp. 5918–5926, 2024a.

Xueyi Li, Youheng Bai, Teng Guo, Ying Zheng, Mingliang Hou, Bojun Zhan, Yaying Huang, Zitao Liu, Boyu Gao, and Weiqi Luo. Extending context window of attention based knowledge tracing models via length extrapolation. In *Proceedings of the 27th European Conference on Artificial Intelligence*, pp. 1479–1486. 2024b.

Zhifei Li, Lifan Chen, Jiali Yi, Xiaoju Hou, Yue Zhao, Wenxin Huang, Miao Zhang, Kui Xiao, and Bing Yang. Keenkt: Knowledge mastery-state disambiguation for knowledge tracing. *arXiv preprint arXiv:2512.18709*, 2025.

Fei Liu, Xuegang Hu, Chenyang Bu, and Kui Yu. Fuzzy bayesian knowledge tracing. *IEEE Transactions on Fuzzy Systems*, 30(7):2412–2425, 2021a.

Qi Liu, Shuanghong Shen, Zhenya Huang, Enhong Chen, and Yonghe Zheng. A survey of knowledge tracing. *CoRR*, abs/2105.15106, 2021b.

Zitao Liu, Qiongqiong Liu, Jiahao Chen, Shuyan Huang, Boyu Gao, Weiqi Luo, and Jian Weng. Enhancing deep knowledge tracing with auxiliary tasks. In *Proceedings of the ACM web conference*, pp. 4178–4187, 2023a.

Zitao Liu, Qiongqiong Liu, Jiahao Chen, Shuyan Huang, and Weiqi Luo. simplekt: A simple but tough-to-beat baseline for knowledge tracing. In *Proceedings of the Eleventh International Conference on Learning Representations*, 2023b.

Ting Long, Yunfei Liu, Jian Shen, Weinan Zhang, and Yong Yu. Tracing knowledge state with individual cognition and acquisition estimation. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pp. 173–182, 2021.

Koki Nagatani, Qian Zhang, Masahiro Sato, Yan-Ying Chen, Francine Chen, and Tomoko Ohkuma. Augmenting knowledge tracing by considering forgetting behavior. In *Proceedings of the world wide web conference*, pp. 3101–3107, 2019.

Hiromi Nakagawa, Yusuke Iwasawa, and Yutaka Matsuo. Graph-based knowledge tracing: modeling student proficiency using graph neural network. In *Proceedings of the IEEE/WIC/ACM international conference on web intelligence*, pp. 156–163, 2019.

Shalini Pandey and George Karypis. A self attentive model for knowledge tracing. In Michel C. Desmarais, Collin F. Lynch, Agathe Merceron, and Roger Nkambou (eds.), *Proceedings of the 12th International Conference on Educational Data Mining*, 2019.

Shalini Pandey and Jaideep Srivastava. Rkt: relation-aware self-attention for knowledge tracing. In *Proceedings of the 29th ACM international conference on information & knowledge management*, pp. 1205–1214, 2020.

Soonwook Park, Donghoon Lee, and Hogun Park. Enhancing knowledge tracing with concept map and response disentanglement. *Knowledge-Based Systems*, 302:112346, 2024.

Philip I. Pavlik, Hao Cen, and Kenneth R. Koedinger. Performance factors analysis - A new alternative to knowledge tracing. In *Proceedings of the 14th International Conference on Artificial Intelligence in Education*, volume 200, pp. 531–538, 2009.

Radek Pelánek. Bayesian knowledge tracing, logistic models, and beyond: an overview of learner modeling techniques. *User Model. User Adapt. Interact.*, 27(3-5):313–350, 2017.

Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein. Deep knowledge tracing. *Advances in neural information processing systems*, 28, 2015.

Shuanghong Shen, Qi Liu, Enhong Chen, Han Wu, Zhenya Huang, Weihao Zhao, Yu Su, Haiping Ma, and Shijin Wang. Convolutional knowledge tracing: Modeling individualization in student learning process. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pp. 1857–1860, 2020.

Shuanghong Shen, Qi Liu, Enhong Chen, Zhenya Huang, Wei Huang, Yu Yin, Yu Su, and Shijin Wang. Learning process-consistent knowledge tracing. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pp. 1452–1460, 2021.

Xiangyu Song, Jianxin Li, Taotao Cai, Shuiqiao Yang, Tingting Yang, and Chengfei Liu. A survey on deep learning based knowledge tracing. *Knowl. Based Syst.*, 258:110036, 2022.

Chenyang Wang, Weizhi Ma, Min Zhang, Chuancheng Lv, Fengyuan Wan, Huijie Lin, Taoran Tang, Yiqun Liu, and Shaoping Ma. Temporal cross-effects in knowledge tracing. In *Proceedings of the 14th ACM international conference on web search and data mining*, pp. 517–525, 2021.

Chun-Kit Yeung and Dit-Yan Yeung. Addressing two problems in deep knowledge tracing via prediction-consistent regularization. In *Proceedings of the fifth ACM conference on learning @ scale*, pp. 1–10, 2018.

Yu Yin, Le Dai, Zhenya Huang, Shuanghong Shen, Fei Wang, Qi Liu, Enhong Chen, and Xin Li. Tracing knowledge instead of patterns: Stable knowledge tracing with diagnostic transformer. In *Proceedings of the ACM web conference*, pp. 855–864, 2023.

Michael Yudelson, Kenneth R. Koedinger, and Geoffrey J. Gordon. Individualized bayesian knowledge tracing models. In *Proceedings of the 16th International Conference on Artificial Intelligence in Education*, volume 7926, pp. 171–180, 2013.

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