Not All Options Are Created Equal: Textual Option Weighting for Token-Efficient LLM-Based Knowledge Tracing

Jongwoo Kim[1,*] **Seongyeub Chu**[1,*] **Bryan Wong**[1] **Mun Young Yi**[1,†]
[1]KAIST, Daejeon, Republic of Korea
{gsds4885, chseye7, bryan.wong, munyi}@kaist.ac.kr

Abstract

Large Language Models (LLMs) have recently emerged as promising tools for knowledge tracing due to their strong reasoning and generalization abilities. While recent LLM-based KT methods have introduced new prompt formats, they struggle to reflect all histories of example learners within a single prompt during in-context learning (ICL), leading to limited scalability and high computational cost under token constraints. In this work, we present *LLM-based Option weighted Knowledge Tracing (LOKT)*, a simple yet effective LLM-based KT framework that encodes the interaction histories of example learners in context as *textual categorical option weights (TCOW)*. These are semantic labels (e.g., "inadequate") assigned to the options selected by learners when answering questions, helping understand LLM. Experiments on multiple-choice datasets show that LOKT outperforms existing non LLM- and LLM-based KT models in both cold-start and warm-shot settings. Moreover, LOKT enables scalable and cost-efficient inference, performing strongly even under strict token constraints. Our code is available at https://github.com/kimjongwoo-cell/LOKT$_{model}$.

1 Introduction

Knowledge Tracing (KT) is a core method in learning analytics that aims to estimate and track a learner's evolving understanding of specific knowledge components (KCs) over time. It models how a learner's knowledge state changes with each learning interaction, such as answering a question, and uses this to predict future performance (Piech et al., 2015; Xia et al., 2019). However, its effectiveness drops in cold-start scenarios where interaction data is limited (Fu et al., 2024; Zhao et al., 2020).

To tackle the bottleneck, there have been increasing efforts to leverage Large Language Models (LLMs), which possess broad prior knowledge and strong reasoning capabilities (Santoso et al., 2024; Doe and Smith, 2023). Most existing approaches rely on either fine-tuning (FT) LLMs using learner interaction data (Jung et al., 2024; Doe and Smith, 2023) or in-context learning (ICL) (Li et al., 2024a). Particularly, the ICL-based approach has recently gained attentions owing to its flexibility and practicality by providing prompts without updating model parameters. As shown in Figure 1-(a), this method predicts how the test student will answer the next question by looking at a few example sequences from other students, which are provided in the prompt as references.

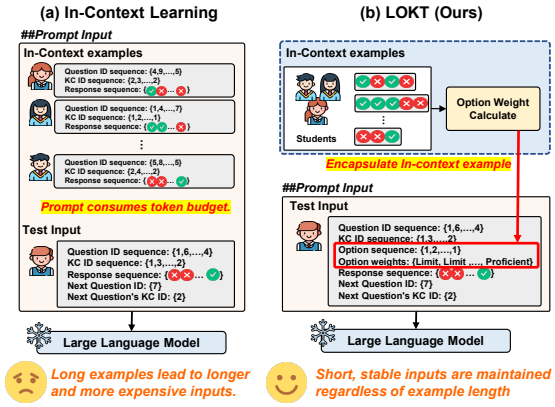


Figure 1: Comparison between the ICL approach and LOKT. (a) ICL requires including complete learner interaction histories in prompts, resulting in increased token usage, while (b) LOKT compresses information through textual option weights, maintaining constant token usage while achieving superior performance.

While ICL-based KT has shown promising performance in few-shot prompt settings, it raises a key question: **Is this approach truly effective for KT tasks, where each example consists of long interaction sequences?** To investigate this, we examined the cost, scalability, and performance of ICL-based KT across various few-shot configurations (Figure 2). Including more examples in

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 16114–16128
November 4-9, 2025

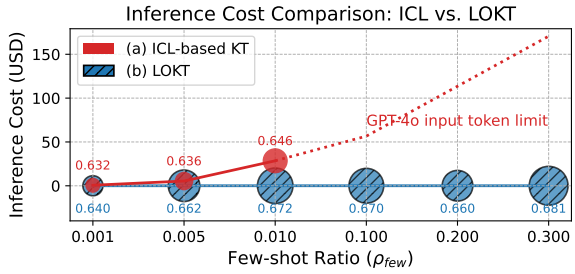


Figure 2: Inference cost and performance across few-shot ratios for (a) ICL-based KT and (b) LOKT. ICL incurs rising cost and hits GPT-4o's token limit as few-shot size increases. In contrast, LOKT maintains fixed cost while improving with more examples.

the prompt increases token usage and inference cost, while reducing them due to token budget constraints degrades performance. These patterns, consistent with findings in other domains (Jiang et al., 2023; Li et al., 2024a; Han et al., 2024; Liu et al., 2023), highlight the structural limitations of ICL for long-context KT scenarios.

To address these limitations, we propose a novel method called **LOKT**(LLM-based Option weighted Knowledge Tracing). As shown in Figure 1 (b), LOKT generates an encapsulated representation of learner behavior by using option weights, instead of inserting full interaction histories into prompts as in prior ICL-based work. This approach allows LOKT to maintain a fixed input length regardless of the number of learner interactions. Option weights quantify learners' current knowledge states represented in the selected option in multiple-choice question (MCQ) responses. These weights support systematic analysis of learner understanding or misconception based on option-level selections. However, their continuous values (ranging from -1 to 1 (Huang et al., 2024; An et al., 2022)) are difficult for LLMs to interpret (Santoso et al., 2024). To address this, we propose textual categorical option weights (TCOW) which transforms the continuous weights into textual categories such as *Inadequate*, *Limited*, and *Proficient* (see Figure 1 b). This facilitates LLMs' comprehension of the information lying in selected options.

We conduct experiments on five multiple-choice knowledge tracing datasets. To assess the effectiveness of our approach, we compare LOKT with both LLM-based baselines and conventional KT models under cold-start condition. Our findings show that LOKT improves LLMs' knowledge tracing performance with shorter prompts which save costs. Our contributions are as follows:

- We introduce LOKT, a method that leverages option weights to distill long-context information into a compact sequence, thereby enhancing KT performance in a cost-effective and efficient manner, particularly in cold-start scenarios.
- We propose TCOW, which further transforms the continuous option weights into textual categories, enhancing LLMs' interpretation of the option information during KT.
- Experiments on five multiple-choice KT datasets show that LOKT achieves higher accuracy and stability in diverse cold scenarios and warm scenarios than previous KT and LLM-based methods.

2 Related Work

2.1 Knowledge Tracing

Knowledge tracing (KT) aims to model and predict a learner's evolving mastery of KCs based on their past interactions with educational content. KT originated from statistical methods such as maximum likelihood estimation (MLE), item response theory (IRT) (Lord, 1980), and Bayesian knowledge tracing (BKT) (Corbett and Anderson, 1995). Since the evolution of neural networks, models like DKT (Piech et al., 2015) and DKT+ (Chen et al., 2020) leveraged RNNs and regularization to capture temporal patterns and improve stability. Attention-based models, including KQN (Lee and Yeung, 2019), SAKT (Xia et al., 2019), and their variants (Kim et al., 2020; Hu et al., 2021), enhanced prediction of learners' knowledge state by focusing on salient student-exercise interactions. More recent methods like ReKT (Shen et al., 2024) and ExtraKT (Li et al., 2024b) integrate contextual signals to accurately trace pupils' knowledge states. However, these models remain limited in cold-start settings with limited interaction history (Doe and Smith, 2023). To tackle the bottleneck, we newly incorporate textual option weights into LLMs' knowledge tracing.

2.2 LLM based Knowledge Tracing

LLMs have recently emerged as strong alternatives for KT, leveraging broad prior knowledge

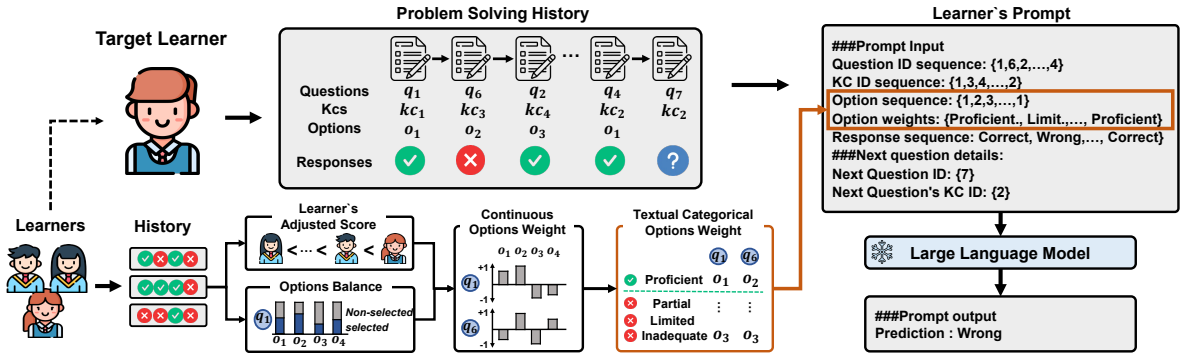


Figure 3: Option weights are first computed from peer learners' interaction data, incorporating learners' adjusted scores and option selection patterns. These option weights are converted into textual form that the LLM can better understand. By using these option weights, we encapsulate the information of in-context example learners, enabling effective representation without directly including the full interaction history in the prompt.

and reasoning to infer learners' states from limited data (Brown et al., 2020; Doe and Smith, 2023). Most approaches focus on prompt engineering rather than introducing new model architectures, typically relying on fine-tuned LLMs. Neshaei et al. use prompts summarizing KC correctness rates (Neshaei et al., 2024), while CLST suggests a [KC, Response] pair sequence as a prompt, framing KT as a language modeling task (Jung et al., 2024). Fine-tuning poses practical limitations in real-world settings, such as requiring separate training for distinct datasets. Consequently, inference-only methods such as ICL have gained attention by embedding learner histories into prompt-templates without updating model parameters (Li et al., 2024a). However, its scalability is still limited due to prompt length, token costs, and other challenges such as context management and inference latency (Zhao et al., 2023).To address these challenges, we propose LOKT, an efficient prompt-engineering method that uses textual option-weight summaries for efficient KT.

3 Methodology

3.1 Overview

LOKT introduces a structured prompting framework that compresses learner interactions into semantically meaningful representations to address the limitations of in-context learning in knowledge tracing. Figure 3 illustrates the overall architecture. We derive continuous option weights from peer learners, transform them into textual form, and incorporate them into the learner's interaction sequence to construct structured prompts.

3.2 Problem Definition

Knowledge Tracing. KT aims to model and predict a learner's evolving mastery of KCs based on their past interactions. It not only predicts whether a learner will correctly answer the next question but also captures temporal patterns in their learning behavior. Each interaction $x_{t,i}$ at time step t for learner i includes the question $q_{t,i}$, its associated knowledge components $c_{t,i}$, and the response label $y_{t,i}$. The goal is to estimate the correctness of the next response $\hat{y}_{t+1,i}$, computed as:

$$\hat{y}_{t+1,i} = \arg\max_{\omega} P(\omega \mid q_{t,i}, c_{t,i}) \quad (1)$$

Here, ω is the predicted label indicating whether the learner will answer correctly. This formulation models KT as a sequential prediction task.

Few-shot Cold-start Setting. This setting evaluates the model's generalization ability when only a limited number of learners are available. Non-LLM models follow a few-shot training setup, where the proportion of learners used for training is determined by ρ_{few}. In contrast, LLM-based models perform inference without any parameter updates and rely on ICL, where a small number of exemplar learners corresponding to ρ_{few} are included in the prompt, subject to token length constraints. Unlike general NLP tasks where few-shot size is defined by the number of exemplars (e.g., 2, 4, or 8), few-shot learning in KT is typically defined based on the proportion of training data.

3.3 Learning from Option Information

LLM-based KT models often simplify responses into correct or wrong classifications, overlooking

detailed insights from learners' option choices. Option weights address this gap by analyzing misconceptions in wrong responses, providing a more granular assessment of knowledge (Ghosh et al., 2021; An et al., 2022; Huang et al., 2024). However, continuous option weights, while capturing quantitative differences, pose challenges in integration with LLMs as the range of weights varies depending on questions where the same option weights of different exercise can indicate different insights. It can cause confusion for LLMs (Santoso et al., 2024). To overcome this, we replace these numerical weights into text-based categorical values, enhancing the LLMs' ability to interpret learners' proficiencies and misconceptions effectively.

3.3.1 Continuous Option Weight

We adopt the method from (Huang et al., 2024), which computes continuous option weights in two steps: estimating learner proficiency and aggregating these estimates over option choices. Detailed calculations are provided in the Appendix E.

Learner Proficiency Measurement. For option weights calculation, we first measure each learners' proficiency based on their average correct rate, adjusted by exercise difficulty to account for varying problem-solving logs and question sets. The difficulty of a question is defined as $d_q = \frac{\text{correct}_q}{\text{count}_q}$, where count_q denotes the number of attempts on question q, and correct_q denotes the number of correct answers. Subsequently, each learner's score $\bar{x}_i$ is calculated by subtracting the average difficulty of the questions they attempted from their total number of correct answers:

$$\bar{x}_i = \frac{N_i - \sum_{q \in Q_i} d_q}{|Q_i|} \tag{2}$$

Here, i denotes a learner, N_i is the number of correctly answered questions, and Q_i is the set of questions attempted by learner i. This adjustment increases the score for correctly answering difficult questions and decreases it for wrong answers to easier ones. Then, we normalize the adjusted scores using the global mean ($\bar{x}$) and standard deviation (S_x) computed over all learners I.

Option Weight Calculation. To compute option weights that reflect the diagnostic relevance of each choice, we analyze how option selection correlates with learner proficiency. The objective is to assign higher weights to options that are indicative of either understanding or common misconception.

For a given question q and option o, we compute the selection ratio as $C_q^o = \frac{c_q^o}{c_q^o + nc_q^o}$, where c_q^o and nc_q^o denote the number of learners who selected and did not select the option, respectively. The non-selection ratio is defined as $NC_q^o = 1 - C_q^o$. We then divide learners into two groups and compute their average adjusted scores:

$$\bar{x}_{c_q^o} = \frac{\sum_{i \in I_q^o} \bar{x}_i}{|C_q^o|}, \quad \bar{x}_{nc_q^o} = \frac{\sum_{i \in NI_q^o} \bar{x}_i}{|NC_q^o|} \tag{3}$$

where $\bar{x}_i$ is the adjusted proficiency score of learner i, I_q^o denotes the set of learners who selected option o for question q, and NI_q^o denotes the set of learners who did not select option o. The difference between these two averages, $\bar{x}_{c_q^o} - \bar{x}_{nc_q^o}$, reflects how strongly the option's selection correlates with learner proficiency. Using these statistics, we define the continuous option weight as:

$$w_q^o = \frac{\bar{x}_{c_q^o} - \bar{x}_{nc_q^o}}{S_x} \times \sqrt{C_q^o \cdot NC_q^o} \tag{4}$$

3.3.2 Textual Categorical Option Weight

For representation of the relative importance of options, we sort the weights for question q as $w_q^{(1)} \geq w_q^{(2)} \geq \cdots \geq w_q^{(n_q)}$, which provides ordinal position but lacks semantic meaning, limiting its utility for language models. To enhance interpretability and better align with LLM capabilities, we convert these ranks into semantically meaningful textual categories. Each option is assigned a category based on its rank: the highest ($w_q^{(1)}$) is *Proficient*, then *Partial*, *Limited*, and the lowest ($w_q^{(n_q)}$) is *Inadequate*. based on its relative weight. This transformation supports LLMs' interpreting learners' understanding, where *Proficient* reflects strong mastery and *Inadequate* suggests misconceptions. When included in prompts, these descriptors offer semantically rich signals that help the LLM reason about learner behavior. Some options may lack sufficient observations to compute stable weights; in such cases, we set the weight to `NaN`. For completeness, mappings for questions with more than four options are provided in Appendix C.3.

3.4 Prompt Construction for KT

To predict whether a learner will answer correctly at time $t+1$, we encode their interaction history up to time t into a structured prompt. To guide LLMs in capturing problem-solving components,

namely question, KC, options, option weights, and response, we separately provided the sequence of each component to LLMs. The prompt includes the target learner's problem-solving history, along with their selected options and the corresponding TCOW representations. We provided question ID corresponding KC ID at time $t+1$ at the end of the prompt-template. Option weights are computed from the interaction of learners and exercises in few-shot examples, then transformed into TCOWs associated with the target learner's chosen options. The final prompt is fed into a frozen LLM (e.g., GPT-4o) to predict the learner's next response as either `correct` or `wrong`. An example of this prompt structure is shown in Appendi F .

4 Experiment

This section presents the experimental setup and results that validate the effectiveness of the proposed methodology. We describe the datasets used, model implementation details, and baseline configurations. To systematically evaluate our method, we design experiments to answer the following three research questions:

RQ1. *Does option weights improve LLMs' knowledge tracing performance in cold-start settings?* We evaluate whether LOKT outperforms both traditional KT and LLM-based KT methods when cold-start settings. Additionally, we conduct experiments under warm-start settings to verify the robustness of LOKT.

RQ2. *What are the scalability and efficiency benefits of using option-weight encapsulated prompts in LLM-based knowledge tracing?* We conduct experiments comparing inference costs across various few-shot settings and evaluate performance under fixed token budgets to assess the impact of encapsulating option weights.

RQ3. *To what extent does TCOW improve the knowledge tracing performance of LLMs?* We assess whether TCOW improves LLM reasoning by comparing representation formats and textual semantic variants.

4.1 Datasets and Baselines

Dataset. We used five public MCQ datasets: Eedi B, A (Wang et al., 2021), EdNet (Choi et al., 2020), Assistment09, and DBE-KT22 (Abdelrahman et al., 2022) (see Table 1). Eedi B, Eedi A, and EdNet have 4 options per question, while Assistment09 and DBE-KT22 have variable options (up to 7 and 5, respectively). We removed instances with missing or unselected answers during preprocessing.

Table 1: Datasets Description

	Eedi B	Eedi A	EdNet	Assistment09	DBE-KT22
# Interactions	1,377,653	15,840,680	95,293,926	238,307	307,058
# Learners	4,918	118,971	784,309	7,353	1,264
# Questions	948	27,613	12,284	1,993	212
# KCs	86	388	188	315	93
Sparsity	70.45%	99.52%	99.01%	98.37%	85.41%
# Options	4	4	4	7 (max)	5 (max)

Baseline Models. We compare our model against state-of-the-art KT methods, including traditional approaches such as DKT (Piech et al., 2015), DKVMN (Zhang et al., 2017), SAKT (Xia et al., 2019), AKT (Ghosh et al., 2020), ExtraKT (Li et al., 2024b), and ReKT (Shen et al., 2024). In addition, we consider recent LLM-based methods like (Neshaei et al., 2024) and CLST (Jung et al., 2024). Although these LLM-based models originally involve fine-tuning, their primary contribution lies in novel prompt design. Therefore, we adopt their prompt formats and evaluate all models in an inference-only setting to ensure a fair comparison with our approach. Detailed descriptions of the baseline models are provided in the Appendix B.

4.2 Model Training Setup

All experiments use five different random seeds. Each dataset are split into 70% training, 15% validation, and 15% test sets. Non-LLM models use 100-dimensional embeddings and are trained with the Adam optimizer, with both the learning rate and weight decay set to 0.001. A batch size of 256 is used, and early stopping is applied with a patience of 5 epochs. The training data is sampled per label according to the given few-shot ratio (ρ_{few}), and each learner's sequence length is limited to 50. All training runs on a system equipped with an NVIDIA RTX 3090 Ti GPU.

4.3 Inference and Evaluation Protocol

Both LLM-based and non-LLM models are evaluated using a balanced test set consisting of 50 correct and 50 wrong responses. This setting controls API usage and ensures fair comparisons across models. LLM baselines operate in an inference-only mode via the OpenAI API, primarily using few-shot ICL prompts, with ID-based prompts applied. Our proposed method, LOKT, is evaluated in a without in-context examples input directly rather than use uncapsulted . All reported results are averaged over multiple random seeds.

Table 2: Performance comparison of ACC and F1 scores (mean±std) at a training ratio of $\rho_{\text{train}} = 0.001$. **Bold** indicates the best performance, and *underline* denotes the second-best.

	Eedi B		Eedi A		Ednet		Assistment09		DBE-KT22	
Model	**ACC**	**F1**	**ACC**	**F1**	**ACC**	**F1**	**ACC**	**F1**	**ACC**	**F1**
Non-LLM Based										
DKT	0.495±0.044	0.496±0.041	0.489±0.061	0.497±0.062	0.527±0.050	0.513±0.077	0.490±0.067	0.495±0.076	0.471±0.048	0.481±0.048
DKVMN	0.515±0.048	0.526±0.034	0.503±0.023	0.503±0.034	0.539±0.021	0.521±0.060	0.459±0.054	0.459±0.071	0.490±0.034	0.520±0.035
SAKT	0.493±0.085	0.494±0.092	0.513±0.091	0.497±0.093	0.483±0.034	0.540±0.092	0.483±0.092	0.540±0.099	0.548±0.081	0.623±0.083
AKT	0.535±0.065	0.476±0.024	0.523±0.045	0.538±0.048	0.538±0.023	0.645±0.048	0.507±0.084	0.574±0.071	0.493±0.028	0.659±0.059
ExtraKT	0.573±0.018	0.518±0.022	0.561±0.018	0.534±0.020	0.552±0.016	0.522±0.021	0.568±0.018	0.529±0.023	0.571±0.022	0.533±0.023
ReKT	0.577±0.019	0.522±0.023	0.563±0.018	0.538±0.021	0.554±0.017	0.527±0.019	0.571±0.019	0.532±0.024	0.573±0.021	0.537±0.024
LLM-Based (GPT-4o)										
(Neshaei et al., 2024)	0.660±0.015	0.679±0.014	0.624±0.019	0.615±0.018	0.581±0.020	**0.656±0.016**	0.632±0.015	**0.681±0.009**	0.608±0.017	0.688±0.014
CLST	0.674±0.007	0.678±0.011	0.638±0.015	0.660±0.016	0.577±0.020	0.616±0.018	0.620±0.012	0.651±0.017	0.657±0.019	0.693±0.018
LOKT	**0.694±0.017**	**0.683±0.019**	**0.686±0.015**	**0.698±0.017**	**0.668±0.012**	0.646±0.011	**0.640±0.016**	0.644±0.016	**0.698±0.011**	**0.737±0.019**

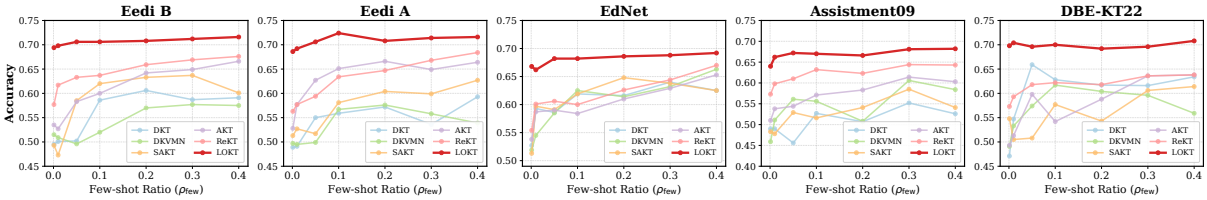


Figure 4: Performance comparison across different training ratios (ρ_{few} from 0.001 to 0.4). LLM-based baselines are excluded due to prompt length limitations in ICL. LOKT results are reported using GPT-4o.

5 Results

5.1 Performance in Cold-Start (RQ1)

Extreme Cold-Start Results. Table 2 presents accuracy and F1 scores under an extreme cold-start setting ($\rho_{\text{few}} = 0.001$) across five public datasets, where LLM-based models consistently outperform traditional KT approaches, demonstrating superior generalization with minimal data. Notably, **LOKT** achieves the best overall performance by leveraging structured, context-aware prompts and incorporating TCOW, enabling more accurate predictions of learner responses. Compared to prior LLM-based baselines such as CLST and (Neshaei et al., 2024), LOKT provides richer semantic representations, maintaining robust performance even under severe data scarcity.

Performance across Few-shot Ratios. Figure 4 illustrates the performance of various KT models, under different Few-shot ratios. Overall, LOKT consistently achieves the highest accuracy across all datasets. This strong performance is attributed to LOKT's use of TCOW, which quantify not only correct answers but also the degree of understanding and misconceptions reflected in wrong choices. LLM-based ICL models such as CLST and (Neshaei et al., 2024) are excluded due to prompt length limitations. This indicates that while ICL methods can partially address the cold-start problem, they face challenges in scalability and efficiency. In this regard, LOKT presents a more practical and robust solution for cold-start KT.

Evaluating Generalization Capacity.

We evaluate the generalization performance of incorporating option weights under warm-start settings ($\rho_{\text{few}} = 1$) to complement our main focus on cold-start scenarios. Experiments were conducted using two interaction lengths, 50 and 100. As shown in Figure 5, LOKT consistently outperformed baseline models, with a significant performance gain observed when using option weights compared to without option weights (LOKT w/o wgt). These results demonstrate that option-weighted representations enable LOKT to effectively capture nuanced learner behaviors across various data availability conditions, highlighting its robustness and practical applicability beyond cold-start settings.

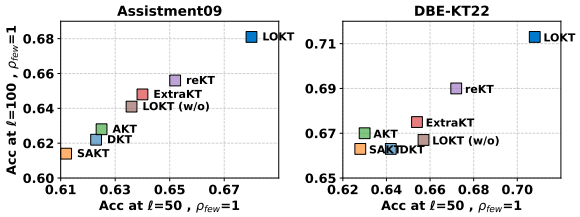


Figure 5: Accuracy at interaction sequence lengths $\ell = 50$ and $\ell = 100$ with full training data ($\rho_{\text{few}} = 1$) on Assistment09 and DBE-KT22 datasets. LOKT performs best, particularly with longer sequences.

5.2 Scalability and Efficiency (RQ2)

To evaluate the scalability and efficiency of the proposed encapsulated format using option weights, we compare LOKT with an ICL-based baseline. The baseline suffers from increased prompt lengths as the number of examples grows, leading to higher inference costs and potential context limit overflows. This study shows whether LOKT can overcome these challenges and maintain effective performance under resource-constrained conditions. To this end, we design two independent experiments: one focusing on prompt scalability and the other on performance under a fixed token budget.

Prompt Length and Token Cost. As shown in Figure 6, token usage in ICL increases linearly with the number of few-shot examples, resulting in higher inference costs and potential prompt overflow. In contrast, LOKT compresses information through TCOW, maintaining a nearly constant prompt length. These results highlight LOKT's superior scalability and token-level efficiency. The inference cost was computed based on GPT-4o pricing at $3.75[1] per million input tokens, measured per single test input. Total cost includes input and output tokens per query, emphasizing the economic advantage of LOKT's compact prompts.

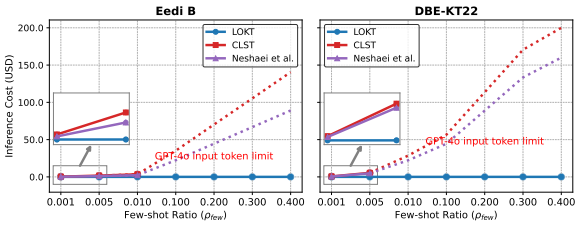


Figure 6: Comparison of inference cost (USD) across training ratios. LOKT keeps token usage low and stable, while ICL-based methods incur higher costs and may exceed GPT-4o's context limit.

Performance under Resource Constraints. We fix the token budget to a maximum of 2000 tokens, reflecting LOKT's typical prompt length, and compare model performance under identical conditions. As shown in Figure7, LOKT consistently achieves the highest accuracy across the three datasets. This demonstrates that CTOW-based prompts effectively capture learners' partial understanding and misconceptions. In contrast, CLST and Neshaei et al. (Neshaei et al., 2024) exhibit reduced or unstable performance with limited resources in cold-start scenarios. These findings underscore LOKT's strength in balancing accuracy and resource efficiency.

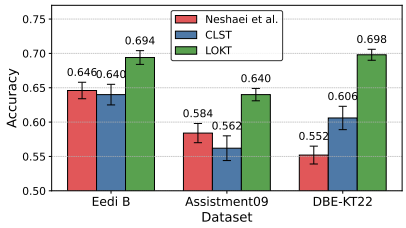


Figure 7: Accuracy under a fixed token budget (2000 tokens) on Eedi B, Assistment09, and DBE-KT22.

5.3 Effect of TCOW on KT Performance of LLMs (RQ3)

To verify whether our proposed option weight method, TCOW, effectively compresses learners' problem-solving histories while preserving rich information, we compare it against various alternative option weighting methods. The first experiment aims to demonstrate the advantages of semantically rich textual representations over continuous and ordinal formats. The second experiment examines the impact of preserving meaningful semantic structure on model performance by comparing TCOW with coarse-grained and randomly shuffled variants.

Effect of Option Weight Representation. Table 3 presents a comparison of three option weight representations: continuous, ordinal, and TCOW. This experiment shows the effectiveness of semantically strutured prompts in LLMs' knowledge tracing. The results demonstrate that TCOW consistently outperforms both continuous and ordinal across all datasets. TCOW integrates pedagogically aligned semantic categories that support an LLM reason more effectively with regard to learners' understanding, leading to clear performance gains. While continuous provides a highly compact encoding and achieves comparable performance to standard ICL-based models, it lacks sufficient structure for LLMs to interpret nuanced learner behavior. Ordinal improves slightly by incorporating ranking information, but the absence of semantic content limits its effectiveness. This confirms that semantic structure, not just compression, is key to enabling LLMs in knowledge tracing.

Effect of Semantic Textual Alignment. To evaluate the impact of semantic alignment in textual categorical option weights, we compare three design strategies. The first variant, *Random*, retains

[1]This price is based on the OpenAI API pricing as of April 28, 2025.

Table 3: Performance with different option weight formats under GPT-4o at $\rho_{\text{few}} = 0.001$.

Method	Eedi B		Eedi A		EdNet		Assistment09		DBE-KT22	
	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
Continuous	0.656	0.625	0.672	0.694	0.622	0.505	0.620	0.614	0.658	0.707
Ordinal	0.668	0.617	0.676	0.696	0.630	0.525	0.630	0.627	0.670	0.714
TCOW	**0.694**	**0.683**	**0.686**	**0.698**	**0.668**	**0.640**	**0.640**	**0.644**	**0.698**	**0.737**

the categorical labels (e.g., *Proficient*, *Partial*, *Limited*, *Inadequate*) but randomly shuffles their assignment to options, disregarding correctness and semantic structure. The second, *Binary*, simplifies the labels into two coarse-grained categories: *Proficient* and *Limited*, eliminating intermediate distinctions. The final variant, **TCOW** (our proposed method), preserves the hierarchy and ensures that label assignment reflects the semantic order implied by the option rankings. As shown in Table 4, TCOW significantly outperforms the other variants, demonstrating that maintaining semantic alignment in textual prompts leads to more accurate modeling of learner understanding. For another few-shot settings, please refer to the Appendix F.3.

Table 4: Performance at $\rho_{\text{few}} = 0.001$ using TCOW with varying semantic textual alignment under GPT-4o.

	Eedi B		Eedi A		Ednet		Assistment09		DBE-KT22	
Model	ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
Random	0.662	0.612	0.656	0.648	0.593	0.560	0.630	0.580	0.654	0.658
Binary	0.666	0.659	0.656	0.651	0.632	0.648	0.620	0.619	0.659	0.684
TCOW	**0.694**	**0.683**	**0.686**	**0.698**	**0.668**	**0.640**	**0.640**	**0.644**	**0.698**	**0.737**

5.4 Further Analysis

To further examine the impact of incorporating option weights into a prompt-template for LLM-based KT, we conducted additional experiments, including an ablation study, model size analysis.

Ablation of Option Weight. In Table 5, LOKT (w/o opt, wgt) removes both option and option weight, LOKT (w/o wgt) omits only option weight. Including only the option sequence yields some improvement, but the best results are achieved when both the option sequence and weights are incorporated. In particular, using TCOW allows the LLM to more accurately interpret the diagnostic meaning of each choice and the learner's understanding. This enables LOKT to capture partial knowledge and misconceptions more effectively than previous methods, highlighting the importance of semantically enriching the prompt for superior performance.

Performance by LLM Parameter Size. LOKT consistently outperforms baseline methods across all model sizes, as shown in Figure 8. As the LLM size

Table 5: Performance for $\rho_{\text{few}} = 0.001$ using GPT-4o. O = Option Prompt, OW = Option Weight

O	OW	Eedi B		Eedi A		EdNet		Assistment09		DBE-KT22	
		ACC	F1	ACC	F1	ACC	F1	ACC	F1	ACC	F1
✗	✗	0.660	0.628	0.660	0.691	0.580	0.609	0.636	**0.658**	0.666	0.706
✓	✗	0.667	0.641	0.664	0.694	0.609	0.630	0.634	0.632	0.668	0.705
✓	✓	**0.694**	**0.683**	**0.686**	**0.698**	**0.668**	**0.640**	**0.640**	0.644	**0.698**	**0.737**

increases, the performance gap in favor of LOKT becomes more pronounced, indicating that larger models benefit even more from the option weight-based prompting scheme. These results highlight the robustness and scalability of LOKT across different LLM environments.

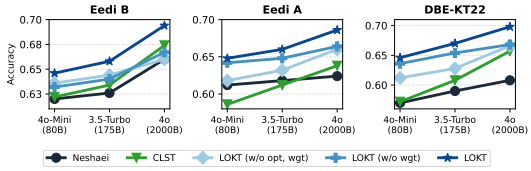


Figure 8: Accuracy of LOKT and baselines across GPT sizes (4o-Mini: 80B, 3.5-Turbo: 175B, 4o: 2000B).

6 Conclusion

In this paper, we analyze the fundamental limitations of existing methods in LLM-based knowledge tracing models developed to tackle the cold-start problem, focusing on inefficiency and scalability issues caused by long interaction histories. To address these challenges, we propose LOKT, a simple yet effective framework that compresses learner interactions into textual categorical option weights within prompts, enabling large language models to efficiently understand learners' problem-solving histories. Extensive experiments on multiple public datasets demonstrate that LOKT surpasses conventional KT models and LLM-based in-context learning methods in accuracy, scalability, and cost efficiency. Our study highlights that leveraging effective compression techniques in in-context learning is key to scalable and practical knowledge tracing with LLMs.

Limitations

This paper has two main limitations. First, it lacks human evaluation of LLM predictions, limiting qualitative validation of interpretability and real-world applicability. Second, the option weight calculation depends on statistical assumptions that may not fully match the data, potentially affecting performance. Future work will include expert

assessments and broader experiments to address these issues.

References

Ghodai Abdelrahman, Sherif Abdelfattah, Qing Wang, and Yu Lin. 2022. Dbe-kt22: A knowledge tracing dataset based on online student evaluation. *arXiv preprint arXiv:2208.12651*.

Suyeong An, Junghoon Kim, Minsam Kim, and Juneyoung Park. 2022. No task left behind: Multi-task learning of knowledge tracing and option tracing for better student assessment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 4, pages 4424–4431.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems (NeurIPS 2020)*, pages 1877–1901.

Zhiwei Chen, Yichao Lu, and Mingjun Zheng. 2020. Dkt+: Enhanced deep knowledge tracing with regularization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1234–1244.

Youngduck Choi, Youngnam Lee, Dongmin Shin, Junghyun Cho, Seoyon Park, Seewoo Lee, Jineon Baek, Chan Bae, Byungsoo Kim, and Jaewe Heo. 2020. Ednet: A large-scale hierarchical dataset in education. In *Artificial Intelligence in Education: 21st International Conference, AIED 2020, Ifrane, Morocco, July 6–10, 2020, Proceedings, Part II 21*, pages 69–73. Springer.

A. W. Corbett and K. L. Anderson. 1995. Knowledge tracing: A model for stochastic modeling of students' kinematics. In *User Modeling, Adaptation, and Personalization*, pages 25–30. Springer.

Jane Doe and John Smith. 2023. Leveraging large language models for enhanced knowledge tracing. In *Proceedings of the 2023 Conference on Educational Data Mining (EDM)*, pages 150–160. Educational Data Mining Society.

Lingyue Fu, Hao Guan, Kounianhua Du, Jianghao Lin, Wei Xia, Weinan Zhang, Ruiming Tang, Yasheng Wang, and Yong Yu. 2024. Sinkt: A structure-aware inductive knowledge tracing model with large language model. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 632–642.

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

Aritra Ghosh, Jay Raspat, and Andrew Lan. 2021. Option tracing: Beyond correctness analysis in knowledge tracing. In *International Conference on Artificial Intelligence in Education*, pages 137–149. Springer.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. 2024. Token-budget-aware llm reasoning. *arXiv preprint arXiv:2412.18547*.

Danyang Hu, Yucheng Wu, and Shun Zheng. 2021. Attention-based knowledge tracing (atkt). In *Proceedings of the 15th International Conference on Educational Data Mining (EDM)*, pages 1–10.

Tao Huang, Xinjia Ou, Huali Yang, Shengze Hu, Jing Geng, Zhuoran Xu, and Zongkai Yang. 2024. Pull together: Option-weighting-enhanced mixture-of-experts knowledge tracing. *Expert Systems with Applications*, 248:123419.

Huiqiang Jiang, Qianhui Wu, Xufang Luo, Dongsheng Li, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. 2023. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. *arXiv preprint arXiv:2310.06839*.

Heeseok Jung, Jaesang Yoo, Yohaan Yoon, and Yeonju Jang. 2024. Clst: Cold-start mitigation in knowledge tracing by aligning a generative language model as a students' knowledge tracer. *arXiv preprint arXiv:2406.10296*.

Seungwon Kim, Jonghyun Lee, and Kyung Hyun Lee. 2020. Separated self-attentive neural knowledge tracing (saint). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1234–1243.

Jinseok Lee and Dit-Yan Yeung. 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.

Haoxuan Li, Jifan Yu, Yuanxin Ouyang, Zhuang Liu, Wenge Rong, Juanzi Li, and Zhang Xiong. 2024a. Explainable few-shot knowledge tracing. *arXiv preprint arXiv:2405.14391*.

Xueyi Li, Youheng Bai, Teng Guo, Ying Zheng, Mingliang Hou, Bojun Zhan, Yaying Huang, Zitao Liu, Boyu Gao, and Weiqi Luo. 2024b. Extending context window of attention based knowledge tracing models via length extrapolation. In *ECAI 2024*, pages 1479–1486. IOS Press.

Junyi Liu, Liangzhi Li, Tong Xiang, Bowen Wang, and Yiming Qian. 2023. Tcra-llm: Token compression retrieval augmented large language model for inference cost reduction. *arXiv preprint arXiv:2310.15556*.

Frederic M. Lord. 1980. *Applications of Item Response Theory to Practical Testing Problems*. Routledge.

Seyed Parsa Neshaei, Richard Lee Davis, Adam Hazimeh, Bojan Lazarevski, Pierre Dillenbourg, and Tanja Käser. 2024. Towards modeling learner performance with large language models. *arXiv preprint arXiv:2403.14661*.

Charles Piech, George Huang, Tom Qin, Yanjie Zhao, Margaret A. Othman, Guangyu Huang, Janette Shim, Jessica Yosinski, and Jan Ondruska. 2015. Deep knowledge tracing. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 505–513.

Jennifer Santoso, Kenkichi Ishizuka, and Taiichi Hashimoto. 2024. Large language model-based emotional speech annotation using context and acoustic feature for speech emotion recognition. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11026–11030. IEEE.

Xiaoxuan Shen, Fenghua Yu, Yaqi Liu, Ruxia Liang, Qian Wan, Kai Yang, and Jianwen Sun. 2024. Revisiting knowledge tracing: A simple and powerful model. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 263–272.

Zichao Wang, Angus Lamb, Evgeny Saveliev, Pashmina Cameron, Jordan Zaykov, Jose Miguel Hernandez-Lobato, Richard E Turner, Richard G Baraniuk, Craig Barton, Simon Peyton Jones, and 1 others. 2021. Results and insights from diagnostic questions: The neurips 2020 education challenge. In *NeurIPS 2020 Competition and Demonstration Track*, pages 191–205. PMLR.

Yingxia Xia, Xiaodong He, and Jun Zhou. 2019. Self-attentive knowledge tracing. In *International Conference on Learning Representations (ICLR)*.

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

Jinjin Zhao, Shreyansh Bhatt, Candace Thille, Neelesh Gattani, and Dawn Zimmaro. 2020. Cold start knowledge tracing with attentive neural turing machine. In *Proceedings of the Seventh ACM Conference on Learning@ Scale*, pages 333–336.

Wayne Xin Zhao, Zhicheng Wei, Siyu Ding, Yusheng Su, Zhiyuan Liu, Maosong Sun, and Ji-Rong Wen. 2023. A Survey of Large Language Models. *arXiv preprint arXiv:2303.18223*.

A Data Source

Table 6: Dataset sources used in this study

Dataset	Source URL
Eedi B	https://Eedi.com/research
Eedi A	https://Eedi.com/research
EdNet	https://github.com/riiid/ednet
Assistment09	https://sites.google.com/site/assistmentsdata
DBE-KT22	https://doi.org/10.26193/6DZWOH

B Baselines

- **DKT**: A recurrent neural network model that captures learners' knowledge states over time based on their interaction sequences.
- **DKVMN**: A dynamic key-value memory network that models individual knowledge components with an external memory structure.
- **SAKT**: A self-attention based model that focuses on important past interactions to predict learner performance.
- **AKT**: An attentive knowledge tracing model incorporating question difficulty and knowledge component relations for improved predictions.
- **ExtraKT**: A model applying linearly decayed attention bias to capture short-term forgetting, enabling robust performance across varying context lengths.
- **ReKT**: ReKT models student knowledge using questions, concepts, and domains with a lightweight cognitive-inspired FRU mechanism, achieving accurate KT with low computational cost.
- **(Neshaei et al., 2024)**:This study explores using pre-trained large language models for knowledge tracing, finding that while zero-shot approaches underperform, fine-tuned LLMs outperform naive baselines and match or exceed traditional Bayesian methods in predicting student performance.
- **CLST**: This study proposes CLST, a generative LLM-based framework for mitigating cold-start issues in knowledge tracing by representing problem-solving data in natural language and fine-tuning the model for effective cross-domain student performance prediction.

C Option Weight Assignment Method

C.1 Option Weight Representation Transformations

Table 7: Comparison of Transformation Methods: Continuous, Ordinal, and TCOW Assignments

Method	Correct Option	Wrong Option 1	Wrong Option 2	Wrong Option 3
Continuous	0.3	0.1	-0.2	-0.5
Ordinal	1	2	3	4
TCOW	Proficient	Partial	Limited	Inadequate

This section explains how continuous option weights, such as those shown in Table 7 (e.g., 0.3, 0.1, -0.2, -0.5), which can be challenging for large language models (LLMs) to interpret directly, are transformed into more interpretable formats. The ordinal transformation assigns a rank to each option based on its weight (e.g., 1, 2, 3, 4), while the categorical transformation maps these weights into meaningful textual labels such as *Proficient*, *Partial*, *Limited*, and *Inadequate*. By converting numerical weights into semantically rich categories, as shown in the categorical row of Table 7, this approach better leverages the strengths of LLMs and enhances the accuracy of knowledge tracing.

C.2 Categorical Option Weight Semantic Transformations

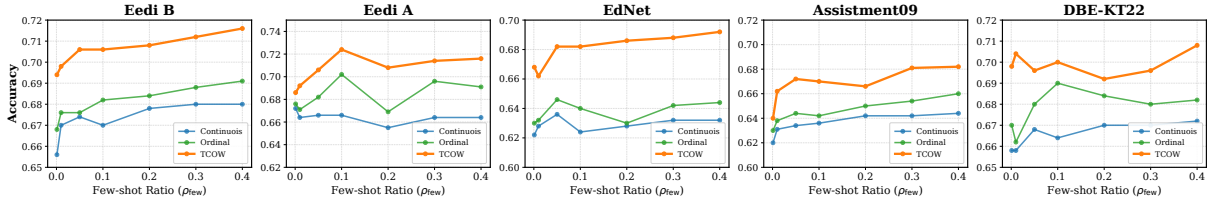
Table 8: Comparison of Transformation Methods: Random, Binary, and TCOW Assignments

Method	Correct Option	Wrong Option 1	Wrong Option 2	Wrong Option 3
Random	Inadequate	Partial	Limited	Proficient
Binary	Proficient	Inadequate	Inadequate	Inadequate
TCOW	Proficient	Partial	Limited	Inadequate

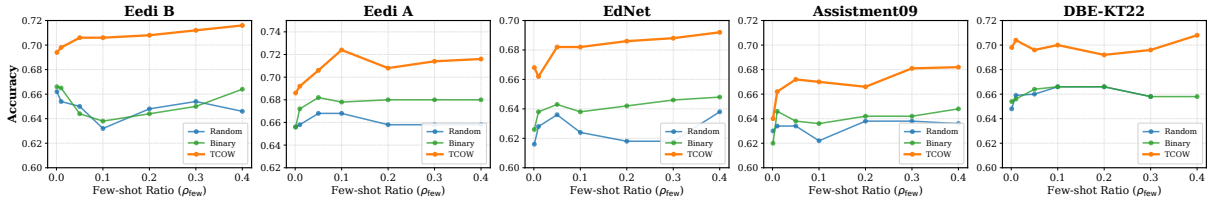
This section compares three methods for transforming option weights into categorical representations: *Random*, *Binary*, and *Textual Categorical Option Weight (TCOW)*. As shown in Table 8, random transformations assign labels arbitrarily, which leads to inconsistent performance. Binary transformations classify the correct answer as *Proficient* and all wrong answers uniformly as *Inadequate*, but this fails to distinguish finer differences among wrong options. In contrast, the fine-grained TCOW method assigns more detailed categories such as *Partial* and *Limited*, effectively capturing varying levels of learner understanding and misconceptions. This approach provides the best alignment for knowledge tracing tasks.

Table 9: Categorical Assignments Based on Number of Options

# Options	Correct	Wrong					
	1st	2nd	3rd	4th	5th	6th	7th
2	Proficient	Inadequate					
3	Proficient	Limited	Inadequate				
4	Proficient	Partial	Limited	Inadequate			
5	Proficient	Partial	Limited	Inadequate	Inadequate		
6	Proficient	Partial	Limited	Limited	Inadequate	Inadequate	
7	Proficient	Partial	Partial	Limited	Limited	Inadequate	Inadequate



(a) Performance metrics for different representation configurations, comparing Continuous, Ordinal, and TCOW (our proposed).



(b) Performance metrics for different semantic configurations, comparing Random, Binary, and TCOW (our proposed).

Figure 9: Ablation analysis for option weight strategies. (a) Semantic configuration variants. (b) Representation format variants.

C.3 Option Weight Expansion for MCQ

This section presents the categorizing scheme for option weights in multiple-choice questions (MCQs) based on the number of options. For MCQs with four options, TCOW, namely *Proficient*, *Partial*, *Limited*, and *Inadequate*, are assigned for each option in descending order. For MCQs with more than four options, additional TCOWs are allocated to the extra options in ascending order of weight values, with a maximum of two. This allocation aims to balance the number of TCOWs while placing greater emphasis on options with higher weights, based on the assumption that these weights carry richer information.

D Performance in Diverse Few-shot

D.1 Effect of Option Weight Representation

As shown in Figure 9a, text-based categorical weights consistently exhibit superior performance across various datasets and scenarios. They perform particularly well in shorter interaction sequences and maintain robust accuracy as sequence lengths increase. The figure highlights that text-based categorical weights deliver consistent results even in data-limited environments, aligning effectively with LLM input structures to enhance both predictive accuracy and interpretability. In contrast, continuous weights tend to struggle in sparse data conditions, failing to capture consistent patterns, while ordinal weights show some improvement but lack the contextual clarity of text-based weights.

D.2 Effect of Textual Option Weight Semantic

The figure 9b shows that LOKT with fine-grained categorical weights consistently outperforms other weight methods (Random, Binary) across diverse few-shot settings. Even at low data ratios and short sequences, LOKT maintains high accuracy, with performance improving further as data ratios and sequence lengths increase. Unlike Binary weights, which fail to capture subtle differences in knowledge states, fine-grained categorical weights effectively leverage the hierarchical structure of options, aligning with LLMs' strength in processing semantically rich inputs. This demonstrates that the performance gains stem from meaningful semantic representations rather than merely converting weights into text.

E Algorithm of option weight calculation

Algorithm 1 Option Weights Calculation

Require: All learner records $X = \{X_1, X_2, \ldots, X_{|I|}\}$ with $|I|$ learners, $|Q|$ questions, and each question has $|O|$ options; A set of learners' proficiency $\bar{x} = \{\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_{|I|}\}$; Standard deviation of average proficiency of all learners S_x;

Ensure: The weight matrix $W = [w_q^o]_{|Q|\times|O|}$

// Calculating the standard deviation of learner score

$\bar{x} \leftarrow \frac{i}{|I|}\sum_{i\in I}\bar{x}_l$ // Average score calculating

$S_x \leftarrow \sqrt{\frac{1}{|I|}\sum_{l\in I}(\bar{x}_i - \bar{x})^2}$ // Standard deviation calculating

// Calculating the proportion of selected and unselected for each option

for $q = 1, 2, \ldots, |Q|$ **do**

  for $o = 1, 2, \ldots, |O|$ **do**

    $c_q^o \leftarrow$ number of learners who selected option o on question q;

    $nc_q^o \leftarrow$ number of learners who did *not* select option o on question q;

    $C_q^o \leftarrow \frac{c_q^o}{c_q^o + nc_q^o}$;

    $NC_q^o \leftarrow \frac{nc_q^o}{c_q^o + nc_q^o}$;

  end for

end for

// Calculating the average score of learners who selected or unselected each option

for $q = 1, 2, \ldots, |Q|$ **do**

  for $o = 1, 2, \ldots, |O|$ **do**

    $I_q^o \leftarrow$ Set of learners who selected option o on question q;

    $NI_q^o \leftarrow$ Set of learners who did not select option o on question q;

    $\bar{x}_{c_q}^o \leftarrow \frac{1}{c_q^o}\sum_{i\in I_q^o}\bar{x}_i$;

    $\bar{x}_{nc_q}^o \leftarrow \frac{1}{nc_q^o}\sum_{i\in NI_q^o}\bar{x}_i$;

  end for

end for

// Calculating the weight of each option for each question

for $q = 1, 2, \ldots, |Q|$ **do**

  for $o = 1, 2, \ldots, |O|$ **do**

    $w_q^o \leftarrow \frac{\bar{x}_{c_q}^o - \bar{x}_{nc_q}^o}{S_x} \cdot \sqrt{C_q^o \cdot NC_q^o}$;

  end for

end for

return W

F Example of System Messages and Predefined Template for LOKT

F.1 LOKT

F.1.1 System Message

```
You are an advanced knowledge tracing expert. You track and predict the learner's knowledge states by
     considering their problem-solving history, associated Knowledge Components (KCs), selected
    options, and the weights of those selected options.
```

F.1.2 Predefined Template

```
Analyze the learner's problem-solving history and predict their performance on the next question.
learner's problem-solving history:
- Question ID sequence: {question_ids}
- KC ID sequence: {kc_ids} (Note: This is a list of KC IDs associated with the next question. For
    example, if next_kc_id is [3, 72], it means the next question involves KC IDs 3 and 72.)
- Selected option sequence: {option_sequence}
- Selected option weights: {option_weights}
- Correctness sequence: {answer_sequence}

Next question details:
- Next question ID: {next_question_id}
- Next question's KC ID: {next_kc_id}

Based on the above information, predict whether the learner will answer the next question (ID: {
    next_question_id}, KC ID: {next_kc_id}) correctly ('correct') or wrongly ('wrong').

Consider the following when making your prediction:
- The learner's overall correctness pattern.
- The complexity and difficulty levels of the questions and KC IDs.
- How the selected options reflect their weight on the learner's understanding and confidence.
- Recent trends in the learner's performance.
- The learner's progression and knowledge improvement over time.
- The learner's current knowledge state for each KC and how it matches the KC IDs in the next
    question.
- Ignore any NaN values in the option weights.

Output only the single word ['correct' or 'wrong']. No other words or punctuation should be included.
```

F.2 LOKT without option and option weight

F.2.1 System Message

```
You are an advanced knowledge tracing expert. You track and predict knowledge states by considering
     the learner's problem-solving history, KCs.
```

F.2.2 Predefined Template

```
Analyze the learner's problem-solving history and predict their performance on the next question.
learner's problem-solving history:
- Question ID sequence: {question_ids}
- KC ID sequence: {kc_ids} (Note: This is a list of KC IDs associated with the next question. For
    example, if next_kc_id is [3, 72], it means the next question involves KC IDs 3 and 72.)
- Correctness sequence: {answer_sequence}

Next question details:
- Next question ID: {next_question_id}
- Next question's KC ID: {next_kc_id}

Based on the above information, predict whether the learner will answer the next question (ID: {
    next_question_id}, KC ID: {next_kc_id}) correctly ('correct') or wrongly ('wrong').

Consider the following when making your prediction:
- The learner's overall correctness pattern.
- The complexity and difficulty levels of the questions and KC IDs.
- Recent trends in the learner's performance.
- The learner's progression and knowledge improvement over time.
```

```
- The learner's current knowledge state for each KC and how it matches the KC IDs in the next
    question.

Provide a detailed assessment of how these factors influence the learner's knowledge state and the
    likelihood of correctly answering the next question.

Output only the single word ['correct' or 'wrong']. No other words or punctuation should be included.
```

F.3 LOKT without option weight

F.3.1 System Message

```
You are an advanced knowledge tracing expert. You track and predict knowledge states by considering
    the learner's problem-solving history, selected options, KCs.
```

F.3.2 Predefined Template

```
Analyze the learner's problem-solving history and predict their performance on the next question.
learner's problem-solving history:
- Question ID sequence: {question_ids}
- KC ID sequence: {kc_ids} (Note: This is a list of KC IDs associated with the next question. For
    example, if next_kc_id is [3, 72], it means the next question involves KC IDs 3 and 72.)
- Selected option sequence: {option_sequence}
- Correctness sequence: {answer_sequence}

Next question details:
- Next question ID: {next_question_id}
- Next question's KC ID: {next_kc_id}

Based on the above information, predict whether the learner will answer the next question (ID: {
    next_question_id}, KC ID: {next_kc_id}) correctly ('correct') or wrongly ('wrong').

Consider the following when making your prediction:
- The learner's overall correctness pattern.
- The complexity and difficulty levels of the questions and KC IDs.
- How the selected options reflect their weight on the learner's understanding and confidence.
- Recent trends in the learner's performance.
- The learner's progression and knowledge improvement over time.
- The learner's current knowledge state for each KC and how it matches the KC IDs in the next
    question.

Output only the single word ['correct' or 'wrong']. No other words or punctuation should be included.
```