Dynamic Student Cognitive Modeling and Performance Prediction in Multi-Agent Teaching Simulations

Anonymous ACL submission

Abstract

In multi-agent teaching simulation scenarios, large language models (LLMs) exhibit an inherent assistant-oriented bias, leading them to generate overly advanced responses when acting as student agents, which limits their ability to accurately reflect real students' cognitive states and learning behaviors. To address this limitation, we model students' cognitive graphs and propose **GraphLR-MPP**, a **Graph**-structured **L**earning **R**eport Enhanced **M**ath **P**erformance **P**redictor, which leverages GraphRAG-generated learning reports to train the model for predicting students' math problem-solving performance. Experimental results demonstrate that our method outperforms existing in-context learning (ICL) approaches and other supervised fine-tuning (SFT) methods. Furthermore, we introduce a Multi-Agent Teaching Intervention Trial that simulates the dynamic updating of students' cognition under instructional interventions, providing a scalable foundation for future agent-based teaching simulation experiments.

1 Introduction

To date, LLM-driven agents have demonstrated strong capabilities in simulating human behavior across domains such as medicine, law, education, and gaming. In educational settings, agent-based simulations typically involve interactions among multiple roles (e.g., teachers, students, teaching assistants, and peers) to support problem solving. Due to their inherently assistant-oriented nature, LLMs perform particularly well when simulating teachers or assistants. However, this same characteristic makes it difficult for them to realistically model students, especially lower-performing students. Instead of reproducing authentic mistakes, they often generate overly correct responses (He-Yueya et al., 2024; Markel et al., 2023; Aher et al., 2023), as illustrated in Figure 1.

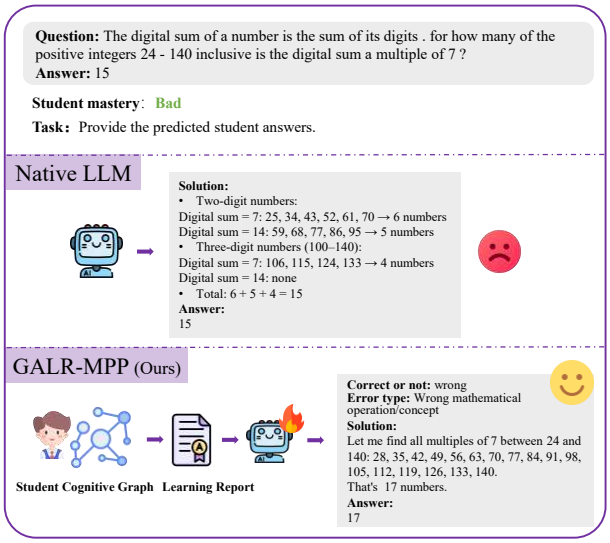

Figure 1: Comparison between a native LLM and our method when predicting the answer of a low-performing student.

Motivated by these observations, we propose a GraphRAG-based dynamic cognitive modeling framework for student simulation. Specifically, we construct individualized student cognitive graphs from historical problem-solving records, where each node represents a knowledge concept with its mastery level, and edges encode relationships between concepts. Related concepts are further clustered into communities, and each community is summarized as a learning report reflecting the student's mastery over that subset of knowledge. The resulting cognitive graph provides a structured representation of evolving student cognition and serves as the basis for predicting subsequent performance.

Current LLM-based approaches for student performance prediction mainly rely on in-context learning (ICL), using either mastery over isolated knowledge components or retrieved records based on textual similarity. However, knowledge-point-based methods fail to capture interactions among

concepts, often leading to unstable predictions. In contrast, similarity-based retrieval relies on surface-level overlap without modeling underlying structures, introducing conceptually irrelevant yet textually similar examples. These limitations highlight the need for structured and dynamic student representations.

Both approaches rely on limited and relatively static features, restricting contextual information and leading to oversimplified predictions. To address this, we propose **GraphLR-MPP**, a graph-enhanced math performance prediction framework that incorporates graph-structured learning reports derived from student cognitive graphs. These reports capture richer student characteristics, including knowledge summaries, practice performance, and evolving mastery states. Through supervised fine-tuning, GraphLR-MPP learns implicit relations between learning reports and student responses, enabling more personalized, accurate, and stable predictions. We evaluate our method on `MathEDU` and the large-scale knowledge tracing benchmark `Nips2020`. Results show consistent improvements over both ICL-based and SFT baselines, particularly in error-type prediction.

Building on this fine-grained student modeling framework, we further develop a multi-agent teaching simulation environment to study two instructional interventions: Explicit Systematic Instruction and Metacognitive Strategies. During intervention, students' cognitive states are dynamically updated via interactions with the teacher agent. This enables simulation of adaptive learning trajectories under different pedagogical strategies. Results show both interventions effectively improve student performance, consistent with findings from real-world educational studies.

Our main contributions are as follows:

- We propose a GraphRAG-based method for constructing dynamic student cognitive models that capture relationships among knowledge concepts and students' cognitive profiles.
- We introduce GraphLR-MPP, a graph-enhanced framework for student math performance prediction that significantly outperforms existing baselines.
- We design a multi-agent teaching intervention simulation with dynamically updated cognitive graphs, providing insights into adaptive educational strategies.

2 Related Work

2.1 Agent-based Educational Simulations

Large language model (LLM)–driven agents have demonstrated human-like behaviors across domains including social interaction, medicine, law, software development, and gaming (Park et al., 2023; Li et al., 2024; Fan et al., 2024; Qian et al., 2024; Wang et al., 2023). In education, recent work leverages multi-agent and cognitively grounded LLM simulations to model classroom dynamics, including learner modeling, Theory-of-Mind–based peer interaction, adaptive teaching strategies, and personality-aware student simulation (Yuan et al., 2025; Gao et al., 2025b; Sanyal et al., 2025; Liu et al., 2024b; Markel et al., 2023; Jinxin et al., 2023; Zhang et al., 2024). Building on these settings, several studies further focus on mathematics education, where agent-based tutoring systems explore instructional strategies for problem solving and dialogue generation (Liu et al., 2024a; Yue et al., 2024; Liu et al., 2025; Macina et al., 2023; Ding et al., 2024). In this paper, we focus on student cognitive modeling and teacher–student interactions for mathematical word problems.

2.2 Student Cognitive Modeling

Student cognition modeling aims to infer learners' knowledge states and cognitive processes for personalized learning and performance prediction. Early models like BKT (Corbett and Anderson, 1994) and deep learning approaches (DKT (Piech et al., 2015), DKVMN (Zhang et al., 2017), SAKT (Pandey and Karypis, 2019), graph-based KT (Nakagawa et al., 2019)) estimate mastery from interaction records, but rely on predefined concepts, limiting reasoning and error pattern capture. With large language models, role-playing agents enable more expressive student modeling. Recent works like Embracing Imperfection (Wu et al., 2025) use static graphs with ICL, Agent4Edu (Gao et al., 2025a) focuses on generative data augmentation, and MathVC (Yue et al., 2024) simulates classrooms via LLM role-playing. Yet most remain static without fine-grained dynamic cognitive graphs or real-world validation. To address this, we construct a continuously updated fine-grained dynamic student cognitive graph with mastery-embedded nodes, hierarchical clustering, and LLM-generated reports, supporting dynamic updating and SFT-based stable prediction.

3 Methods

Our research focuses on modeling students' dynamic cognition and predicting their mathematical problem-solving performance. We adopt a GraphRAG-based approach to construct student cognition and generate learning reports by clustering related knowledge concepts, which are then used to train GraphLR-MPP. Finally, we demonstrate a multi-agent teaching simulation to validate the effectiveness of instructional interventions through dynamically updated student cognition. The overall framework is shown in Figure 2.

3.1 Student Cognitive Graph Construction

Given a student's historical problem-solving records, we construct a personalized knowledge graph for the student. This process is divided into three steps: entity extraction, relation extraction, and community clustering.

Entity Extraction. Firstly, an LLM is applied to each historical record to extract all knowledge points involved in the corresponding problem. Each knowledge point is assigned one of the predefined mathematical knowledge types: *Concept*, *Theorem*, *Method*, *Skill*, or *Model*. Based on the student's performance on the problem, the student's mastery of each knowledge point is further inferred as either *Good* or *Bad*, and a textual description of the mastery status is generated accordingly. Each knowledge point, together with its mathematical knowledge type and the student's mastery information, forms a single node in the knowledge graph.

This process is formally defined as:

$$\mathbf{V}_i = \Pi_v(\mathbf{R}_i) = \{v_{i1}, v_{i2}, \ldots, v_{ik}\} \quad (1)$$

where R_i denotes the i-th historical problem-solving record of a student, including the problem statement, the student's solution process and answer, and the corresponding error analysis. Each node v_{ij} is represented as a tuple $v_{ij} = (n_{ij}, t_{ij}, m_{ij})$, where n_{ij} is the name of the knowledge point, t_{ij} denotes its mathematical knowledge type, and m_{ij} represents the student's mastery information, including the mastery level and its textual description.

Relation Extraction. After extracting the knowledge points corresponding to each record, we further employ an LLM to infer the relationships between these knowledge points. According to definitions in mathematics education, we categorize the relationships into the following five types:

- Prerequisite_of: One knowledge point is a prerequisite for learning another.
- Equivalent_of: Two concepts describe the same mathematical object or property.
- Contains_in: One knowledge point is a special case or subset of another.
- Parallel_to: Two knowledge points belong to different branches of the same category.
- Application_of: A knowledge point is an application of another.

Formally, the set of extracted relationships can be expressed as:

$$\mathbf{E}_i = \Pi_e(\mathbf{V}_i) = \{e_1, e_2, \ldots, e_m\} \quad (2)$$

Each relationship e_j is represented as a tuple $e_j = (s_j, t_j, r_j, d_j, w_j)$, where s_j and t_j denote the source and target entities, r_j indicates the relationship type, d_j provides a textual description of the relationship, and $w_j \in [0, 1]$ represents the strength of the relationship between the source and target entities.

Community Clustering. After constructing the entity–relation graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, we perform community clustering to identify coherent groups of knowledge points. Specifically, following the community partition method in Edge et al. (2024), we apply a hierarchical Leiden algorithm to the largest connected component of the graph to obtain multi-level community partitions. This process can be expressed as:

$$\mathcal{C} = \Phi_{\text{Leiden}}(\mathcal{G}) \quad (3)$$

where $\mathcal{C} = \{c_{l,k}\}$ denotes the set of communities across different hierarchical levels l, where k denotes the community index, and each $c_{l,k}$ corresponds to a subset of knowledge point nodes. A single node may belong to different communities at different levels, enabling multi-granularity representations of the student's cognitive structure.

Based on the resulting community structure, we further generate a learning report for each community. For a given community $c_{l,k}$, its internal entities and relations are first aggregated and serialized into a textual description, which is then fed into an LLM to produce a structured learning report:

$$\mathcal{R}_{l,k} = \Pi_r(c_{l,k}) \quad (4)$$

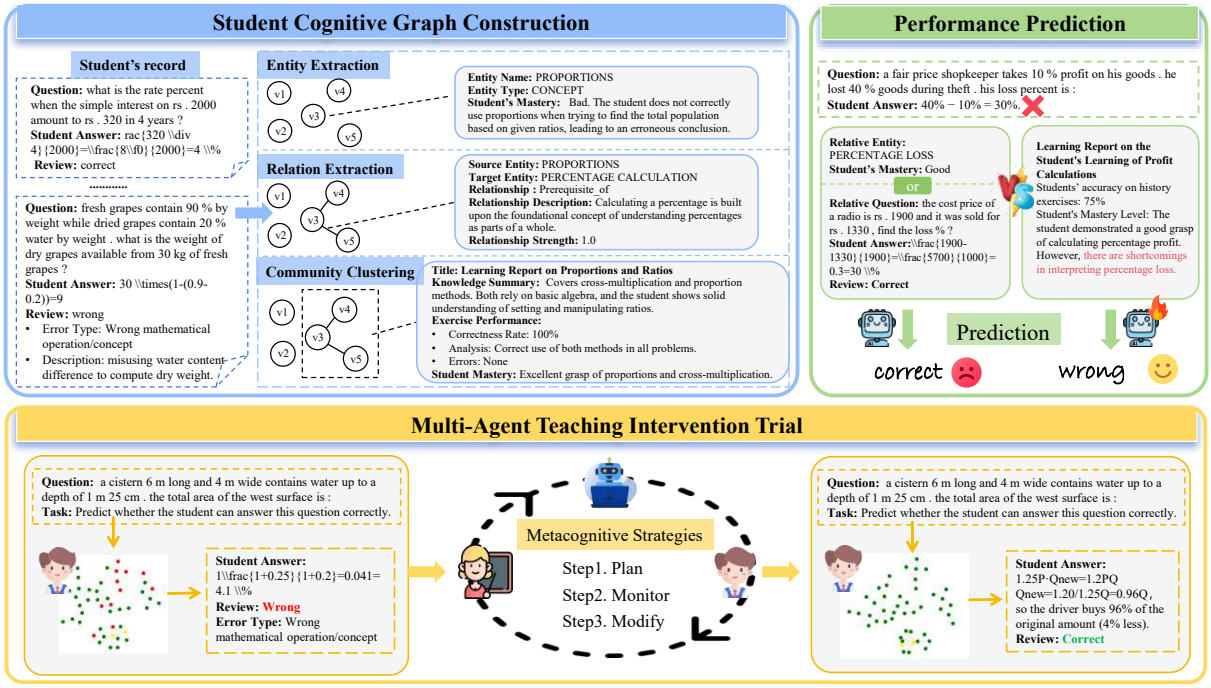

Figure 2: The complete framework of this study, consisting of three components: Student Cognitive Graph Construction, Performance Prediction, and Multi-Agent Teaching Intervention Trial.

The reports are generated in a top-down manner across hierarchy levels, allowing higher-level community reports to incorporate information from previously generated lower-level reports. Each learning report summarizes the student's past performance on exercises related to the corresponding knowledge points, including the overall accuracy, detailed results for each knowledge point, and the main types of errors. Based on this, a concise summary of the student's mastery for the knowledge section is generated, providing teachers with a systematic evaluation of the student's learning status.

3.2 Student Performance Prediction

Existing LLM-based student performance prediction methods rely on ICL with limited and discrete retrieved information, which often introduces bias. As shown in Figure 2, such retrieval may misleadingly suggest strong mastery and lead to incorrect predictions. In contrast, learning reports provide a structured representation of students' knowledge states, but directly applying ICL with them tends to produce overly pessimistic predictions. Therefore, we adopt a SFT strategy for more reliable performance prediction.

Specifically, we fine-tune a general LLM on a student performance prediction dataset D_s. Each training example consists of a structured input sequence $X = \{X_p, X_r\}$, where X_p denotes the problem statement together with the corresponding learning report, and $X_r = (x_1, x_2, \ldots, x_L)$ represents the student performance annotations (e.g., answer correctness and error types), serialized as target tokens. The model is trained to maximize the likelihood of the student response tokens conditioned on the preceding context. The training objective is defined as:

$$\mathcal{L} = -\sum_{i=1}^{L} \log p_\theta\big(x_i \mid X_p, X_{r,<i}\big), \quad x_i \in X_r, \tag{5}$$

where L is the length of the target sequence, x_i denotes the i-th token in the student performance sequence, and $X_{r,<i}$ represents the previously generated student performance tokens before position i.

By minimizing this negative log-likelihood over all samples in D_s, the model learns an explicit mapping from historical exercise accuracy and learning reports to actual student responses, resulting in the final predictive model, **GraphLR-MPP**, which achieves more accurate and calibrated predictions than ICL-based approaches.

3.3 Multi-Agent Teaching Intervention Trial

Most existing student cognitive modeling approaches in teaching scenario simulation with

LLM-Based agents rely on fixed and static representations. However, in real-world settings, students' cognition always evolves over time. Motivated by this observation, we design a multi-agent teaching intervention trial to capture such dynamic changes.

3.3.1 Task Design and Hypotheses

Specifically, we first construct the student cognitive model using the method described in Section 3.1. Subsequently, the teacher agent applies two instructional intervention strategies (Explicit, Systematic Instruction and Metacognitive Strategies) to re-teach the problems that the student previously answered incorrectly. The former delivers concepts or procedures in a structured and sequential manner, consisting of lesson orientation, initial instruction, guided practice, and independent practice. While the latter guide students to regulate their problem-solving through planning, monitoring, and modification.

After the whole intervention process is completed, the student's cognitive graph is updated accordingly. To evaluate the effectiveness of the intervention strategies, we assess the student's performance on a test set of previously unseen problems before and after the intervention. Since both the instructional intervention strategies are evidence-based practices (EBPs), we expect that they can effectively improve student performance. Accordingly, we formulate the following two hypotheses:

- H1: Explicit, Systematic Instruction enhances students' cognitive states and improves their accuracy in mathematical problem solving.
- H2: Metacognitive Strategies enhance students' cognitive states and improve their accuracy in mathematical problem solving.

By testing whether these hypotheses hold, we aim to demonstrate the effectiveness of our student modeling and simulation framework, thereby providing empirical support for the reliability of using this framework for future simulated teaching trial.

3.3.2 Multi-agent Collaboration Framework

We simulate the instructional intervention process through multi-round interactions between teacher and student agents. In addition, we introduce a controller to manage the session flow, which determines whether to proceed to the next phase based on the current interaction. The overall interaction procedure is presented in Algorithm 1.

Algorithm 1 Multi-agent collaboration algorithm

Require: Student S, Teacher T, Controller C, Maxturn m, Teacher's response r^T, Student's response r^S, Controller's response r^C, Dialogue history H, Current intervention step k
Input: Intervention Strategy I, Question q, Correct Answer CA, Student Answer SA, Review R
Output: Dialogue History H

$T \leftarrow (I, q, CA, SA, R)$
$S \leftarrow (q, SA)$
$k = 0$
$T \leftarrow I^{(0)}$
T generate first turn guidance r_0^T
$H \leftarrow H \cup \{r_0^T\}$
for $i = 1$ **to** m **do**
  S generate r_i^S
  $H \leftarrow H \cup \{r_i^S\}$
  $C \leftarrow H^{(k)}$
  C decides whether to stay, move to next step, or end.
  if "end" in r_i^C **then**
    break
  else if "next step" in r_i^C **then**
    $k = k + 1$
    $T \leftarrow I^{(k)}$
  end if
  T generate i-th turn guidance r_i^T
  $H \leftarrow H \cup \{r_i^T\}$
end for
return H

4 Experimental setup

4.1 Datasets

Our experiments are primarily conducted on the `MathEDU` dataset (Hsu et al., 2025), which records authentic problem-solving processes of six real students on MathQA problems (Amini et al., 2019) covering topics such as Geometry and Probability. We select it as our main testbed because it provides rich expert annotations, including correctness, error types (e.g., *Wrong Mathematical Operation/Concept*, *Calculation Error*, *Incomplete Answer*), error equations, and revision advice. Since the original `MathEDU` only contains problem IDs, we merge the records with textual problem statements and correct answers from MathQA, followed by cleaning steps such as removing duplicates, unmatched entries, and unnecessary Chinese fields. Each student has 600–700 exercise records with correctness ranging from 70% to 90%.

To validate generalizability on a larger scale, we additionally evaluate on the NeurIPS 2020 Education Challenge dataset (`Nips2020`) (Wang et al., 2020), which was further preprocessed to convert problem images into textual descriptions (Wang et al., 2025). We randomly select 50 students and split their records into training and test sets at a 7:3 ratio. The training set supports knowledge graph construction and model fine-tuning, while the test set is used for prediction accuracy evaluation. Details of both datasets are provided in Appendix A.1.

4.2 Metrics

We evaluate prediction accuracy across both datasets. Specifically, we report the overall accuracy of different methods in predicting students' answer outcomes and assess model performance on correct and incorrect answers using Precision, Recall, and F1-score. For the `MathEDU` dataset, we place particular emphasis on error type prediction accuracy via an LLM-based evaluation that checks consistency with ground-truth error types. The `NIPS2020` dataset is excluded from this evaluation due to the lack of error type annotations. See Appendix A.2 for more details.

4.3 Baselines

We compare our method with five ICL baselines and two supervised SFT baselines. For ICL, we include the entity-record retrieval method of (Wu et al., 2025), which uses five relevant entities and the most relevant learning record as context, along with its random-retrieval and similarity-retrieval variants. We also evaluate an entities-only baseline and a learning report baseline, where original learning reports are directly used as context to examine the necessity of fine-tuning. All ICL methods are evaluated using `GPT-4o mini` (OpenAI, 2024a) and `GPT-4o` (OpenAI, 2024b) as backbones, with additional experiments conducted on `Qwen2.5-7B-Instruct` (Yang et al., 2024), `Claude-4-Sonnet` (Anthropic, 2025), and `Gemini-2.0-Flash` (Google DeepMind, 2024) to assess cross-family robustness. The two SFT baselines are fine-tuned using either five relevant entities or one relevant record. Details are provided in Appendix A.3.

4.4 Implementation Details

For each selected student, we split their learning records into training and test sets with a 7:3 ratio. In the graph construction stage, we construct student cognitive knowledge graphs from the training data using the GraphRAG framework (Edge et al., 2024). Specifically, GPT-4o mini is used to extract entities and relations from learning records, and the Leiden algorithm is applied to cluster related knowledge concepts into communities. For each resulting community, GPT-4o mini further generates a structured learning report that summarizes the student's knowledge state.

For all SFT baselines, models are fine-tuned on the corresponding training data. We initialize the model with Qwen2.5-7B-Instruct and fine-tune it for five epochs using LoRA (Hu et al., 2022), with rank $r = 8$ and scaling factor $\alpha = 32$. We use a batch size of 2, a learning rate of 2×10^{-4}, a cosine learning-rate scheduler, and a warm-up ratio of 0.1.

During evaluation, we report the overall prediction accuracy on both the `MathEDU` and `Nips2020` datasets. All ICL methods are evaluated using GPT-4o mini and GPT-4o as backbones. On the `MathEDU` dataset, we further test them with Qwen2.5-7B-Instruct, Claude-4-Sonnet, and Gemini-2.0-Flash as additional backbones. Since `Nips2020` lacks detailed annotations for student answer reviews, error-type prediction is conducted only on `MathEDU`.

For multi-agent simulation, the student and teacher agents are powered by GPT-4o, while the controller agent uses GPT-4o mini. The generation temperature is set to 0.7 for the student and teacher agents to encourage diverse interactions, and 0.3 for the controller agent to ensure more stable decision-making. We conduct the Multi-Agent Teaching Intervention Trial on `MathEDU`, where we examine how instructional interventions update students' cognitive knowledge graphs and whether these updates lead to improved answer accuracy.

5 Results

5.1 Student Performance Prediction

5.1.1 Overall Prediction Accuracy

On the `MathEDU` dataset, Table 1 compares GraphLR-MPP with seven baselines, using OpenAI-family models as backbones for the ICL methods. GraphLR-MPP achieves the highest accuracy of 0.748 and consistently outperforms all SFT and ICL baselines. Most ICL methods are biased toward correct-answer predictions, yielding high F1 on correct cases (> 0.75) but low F1 on incorrect cases (0.3–0.4). The Report ICL baseline exhibits the opposite trend, improving recall

Methods		SFT			ICL									
		GraphLR-MPP	Entities	Record	Similarity		Entities+Record		Entities		Random Record		Learning Report	
Base Model		Qwen2.5-7B-Instruct			4o-mini	4o	4o-mini	4o	4o-mini	4o	4o-mini	4o	4o-mini	4o
Overall	Acc.	**0.748**	0.677	0.707	0.696	0.668	0.696	0.713	0.701	0.710	0.670	0.667	0.57	0.468
Correct	Prec.	0.787	0.759	0.756	0.799	0.786	0.764	0.770	0.767	0.770	0.757	0.766	0.792	**0.801**
	Rec.	**0.888**	0.820	0.887	0.786	0.757	0.828	0.853	0.839	0.852	0.807	0.778	0.554	0.379
	F1	**0.833**	0.786	0.813	0.787	0.766	0.793	0.808	0.799	0.807	0.778	0.769	0.649	0.496
Wrong	Prec.	**0.480**	0.327	0.341	0.438	0.391	0.395	0.429	0.407	0.420	0.352	0.364	0.320	0.310
	Rec.	0.316	0.250	0.165	0.466	0.428	0.304	0.309	0.303	0.306	0.303	0.359	0.580	**0.735**
	F1	0.380	0.276	0.221	**0.431**	0.390	0.342	0.357	0.344	0.348	0.309	0.353	0.405	0.415

Table 1: Performance comparison between SFT and various ICL strategies on the `MathEDU` dataset. Best results are highlighted in bold, and the second-best results are underlined.

Methods		SFT			ICL									
		GraphLR-MPP	Entities	Record	Similarity		Entities+Record		Entities		Random Record		Learning Report	
Base Model		Qwen2.5-7B-Instruct			4o-mini	4o	4o-mini	4o	4o-mini	4o	4o-mini	4o	4o-mini	4o
Overall	Acc.	**0.659**	0.573	0.572	0.553	0.552	0.584	0.572	0.570	0.570	0.519	0.517	0.564	0.555
Correct	Prec.	0.445	0.500	**0.523**	0.489	0.489	0.496	0.491	0.482	0.484	0.467	0.472	0.497	0.489
	Rec.	0.438	0.549	0.581	0.656	**0.693**	0.585	0.586	0.552	0.553	0.585	0.610	0.539	0.525
	F1	0.424	0.511	0.538	0.550	**0.564**	0.529	0.527	0.508	0.509	0.507	0.518	0.498	0.487
Wrong	Prec.	**0.570**	0.519	0.451	0.533	0.532	0.545	0.533	0.527	0.532	0.518	0.520	0.527	0.532
	Rec.	**0.627**	0.465	0.421	0.363	0.325	0.448	0.437	0.457	0.459	0.392	0.370	0.488	0.482
	F1	**0.578**	0.481	0.417	0.422	0.397	0.482	0.471	0.483	0.486	0.436	0.421	0.489	0.480

Table 2: Performance comparison between SFT and various ICL strategies on the `Nips2020` dataset. Best results are highlighted in bold, and the second-best results are underlined.

on wrong answers at the expense of correct-answer performance. In contrast, GraphLR-MPP leverages structured learning reports through fine-tuning and achieves more balanced performance on both types of predictions. It also surpasses the two SFT baselines on all metrics. The high correlation between GPT-4o and GPT-4o mini predictions ($\rho \approx 0.99$) further indicates the consistency of ICL results. The complete per-student prediction results and additional backbone results are provided in Appendix B.1.1.

Similar trends are observed on the `Nips2020` dataset. As shown in Table 2, our GraphLR-MPP method again achieves the highest overall accuracy (0.659) and delivers the best performance on wrong answer prediction across Precision, Recall, and F1-score. In addition, students' actual answer correctness significantly affects prediction accuracy across all methods. Figure 11 and Figure 12 demonstrate that GraphLR-MPP consistently outperforms the baselines across all correctness levels, with particularly robust performance for both low-correctness (<0.45) and high-correctness (>0.65) students. A more detailed analysis of this trend is provided in Appendix B.2.

5.1.2 Error-type Prediction

We further evaluate the effectiveness of different methods in predicting students' incorrect answers on the `MathEDU` dataset. Specifically, for the problems that the model correctly predicts as incorrect, we examine the proportion of cases where the predicted error type is consistent with the ground-truth error type. The results are shown in Figure 3. Our GraphLR-MPP method achieves an accuracy of 82.4% in error-type prediction, substantially outperforming all baseline approaches. While the ICL method using a randomly selected historical record as context attains only around 0.35 accuracy, highlighting its limited reliability for capturing nuanced student errors. These results demonstrate that our approach, by leveraging structured learning reports, provides a significantly more faithful and reliable approximation of students' actual error behaviors

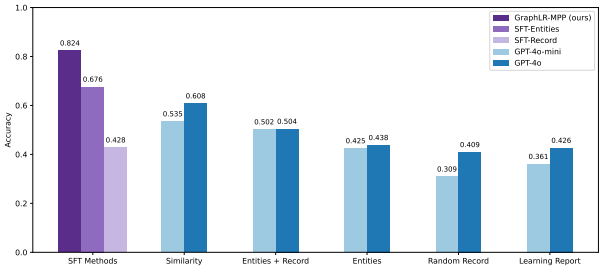

Figure 3: the proportion of consistent error types.

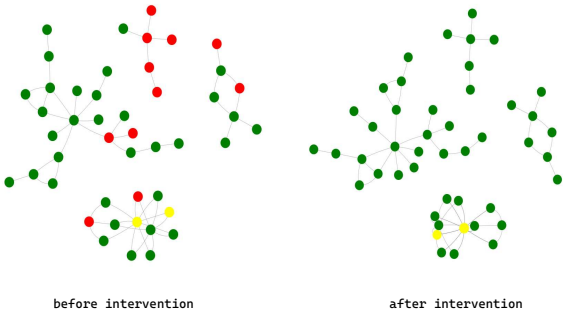

Figure 4: Student's cognitive graph before and after intervention. Red indicates not mastered, green indicates mastered, and yellow indicates partially mastered.

Figure 5: Error rate changes for Student1 and Student5 before and after ESI and MS interventions (light: before; dark: after).

compared to random or similarity-based baselines. Individual student error-type predictions are shown in the Appendix B.1.2.

5.2 Simulated Teaching Intervention

We simulate the teaching intervention process in an interactive agent-based environment using `Student1` and `Student5` from the `MathEDU` dataset as examples. `Student1` and `Student5` originally contain 125 and 127 incorrect responses, respectively. Intelligent agents sequentially act as the two students, while a teacher agent provides re-instruction for each incorrect item using two intervention strategies: Explicit Systematic Instruction (ESI) and Metacognitive Strategies (MS).

We then evaluate the simulated students' post-intervention performance. For `Student1`, 93 items were corrected after ESI, while 114 items were corrected following MS. Similarly, `Student5` corrected 102 and 112 items under the two intervention strategies, respectively. Based on the updated response records, we reconstruct the corresponding knowledge graphs. As illustrated in Figure 4, red nodes denote misconceptions or incorrect knowledge states, whereas green nodes indicate mastered concepts. The transition from red to green reflects the refinement of students' cognitive states after intervention.

Given the test set, we retrieve problem-relevant learning reports based on both the initial and post-intervention knowledge graphs, and use the fine-tuned model to predict students' responses. As shown in Figure 5, error rates for `Student1` decreased by 58.54% and 65.85% after ESI and MS, respectively, while the reductions for `Student5` were 53.12% and 71.88%. These findings suggest that both interventions improve mathematics problem-solving performance, with Metacognitive Strategies showing stronger effectiveness, which offers empirical support for real-world instructional design. Intervention cases and cognitive graph update visualizations are shown in Appendix B.3.

6 Conclusion

This paper constructs student cognitive graphs based on the GraphRAG framework, leveraging LLM to extract knowledge concepts and the relationships among them from students' historical learning records, which are represented as nodes and edges in the graph. Closely related concepts are further clustered to generate learning reports that reflect students' mastery of specific knowledge areas. Based on these learning reports and students' answer records, we fine-tune a base model to obtain a student performance prediction model GraphLR-MPP, achieving superior performance over existing baselines, particularly in error-type prediction. Finally, through a Multi-Agent Teaching Intervention Trial, we simulate the dynamic evolution of students' cognition under instructional interventions by updating the knowledge graph, thereby providing a unified framework and empirical reference for future simulated teaching experiments.

Limitations

A potential limitation of this work is the limited availability of richly annotated real student interaction data. Existing studies on multi-agent educational simulations often rely on synthetic datasets, which may not fully capture the complexity of real-world learning behaviors. To mitigate this, we conduct our main experiments on the `MathEDU` dataset, derived from authentic student problem-solving records. To further examine generalizability, we also evaluate on the large-scale `Nips2020` dataset. However, the `Nips2020` dataset lacks teacher review annotations, which limits further exploration of error analysis. Nevertheless, extensive experiments show that it maintains strong predictive performance while offering promising insights into more fine-grained performance prediction.

Ethics Statements

The data used in this work are publicly available open-source data, and thus do not raise any ethical or moral concerns.

Use of AI Assistants

We primarily used AI tools to assist with writing polishing and language refinement.

References

Gati V Aher, Rosa I Arriaga, and Adam Tauman Kalai. 2023. Using large language models to simulate multiple humans and replicate human subject studies. In *International conference on machine learning*, pages 337–371. PMLR.

Aida Amini, Saadia Gabriel, Shanchuan Lin, Rik Koncel-Kedziorski, Yejin Choi, and Hannaneh Hajishirzi. 2019. Mathqa: Towards interpretable math word problem solving with operation-based formalisms. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: Human language technologies, volume 1 (long and short papers)*, pages 2357–2367.

Anthropic. 2025. System card: Claude Opus 4 and Claude Sonnet 4. Technical report, Anthropic.

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

Yuyang Ding, Hanglei Hu, Jie Zhou, Qin Chen, Bo Jiang, and Liang He. 2024. Boosting large language models with socratic method for conversational mathematics teaching. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 3730–3735.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Zhihao Fan, Jialong Tang, Wei Chen, Siyuan Wang, Zhongyu Wei, Jun Xi, Fei Huang, and Jingren Zhou. 2024. Ai hospital: Interactive evaluation and collaboration of llms as intern doctors for clinical diagnosis. *arXiv preprint arXiv:2402.09742*.

Weibo Gao, Qi Liu, Linan Yue, Fangzhou Yao, Rui Lv, Zheng Zhang, Hao Wang, and Zhenya Huang. 2025a. Agent4edu: Generating learner response data by generative agents for intelligent education systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 23923–23932.

Xian Gao, Zongyun Zhang, Ting Liu, and Yuzhuo Fu. 2025b. Onlinemate: An llm-based multi-agent companion system for cognitive support in online learning. *arXiv preprint arXiv:2509.14803*.

Google DeepMind. 2024. Introducing Gemini 2.0: Our new AI model for the agentic era. https://blog.google/innovation-and-ai/models-and-research/google-deepmind/google-gemini-ai-update-december-2024/. Accessed: 2026-05-26.

Joy He-Yueya, Noah D Goodman, and Emma Brunskill. 2024. Evaluating and optimizing educational content with large language model judgments. *arXiv preprint arXiv:2403.02795*.

Wei-Ling Hsu, Yu-Chien Tang, and An-Zi Yen. 2025. Mathedu: Towards adaptive feedback for student mathematical problem-solving. *arXiv preprint arXiv:2505.18056*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Shi Jinxin, Zhao Jiabao, Wang Yilei, Wu Xingjiao, Li Jiawen, and He Liang. 2023. Cgmi: Configurable general multi-agent interaction framework. *arXiv preprint arXiv:2308.12503*.

Junkai Li, Siyu Wang, Meng Zhang, Weitao Li, Yunghwei Lai, Xinhui Kang, Weizhi Ma, and Yang Liu. 2024. Agent hospital: A simulacrum of hospital with evolvable medical agents. *arXiv preprint arXiv:2405.02957*.

Ben Liu, Jihai Zhang, Fangquan Lin, Xu Jia, and Min Peng. 2025. One size doesn't fit all: A personalized conversational tutoring agent for mathematics instruction. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 2401–2410.

Jiayu Liu, Zhenya Huang, Tong Xiao, Jing Sha, Jinze Wu, Qi Liu, Shijin Wang, and Enhong Chen. 2024a. Socraticlm: Exploring socratic personalized teaching with large language models. *Advances in Neural Information Processing Systems*, 37:85693–85721.

Zhengyuan Liu, Stella Xin Yin, Geyu Lin, and Nancy Chen. 2024b. Personality-aware student simulation for conversational intelligent tutoring systems. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 626–642.

Jakub Macina, Nico Daheim, Sankalan Chowdhury, Tanmay Sinha, Manu Kapur, Iryna Gurevych, and Mrinmaya Sachan. 2023. Mathdial: A dialogue tutoring dataset with rich pedagogical properties grounded in math reasoning problems. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5602–5621.

Julia M Markel, Steven G Opferman, James A Landay, and Chris Piech. 2023. Gpteach: Interactive ta training with gpt-based students. In *Proceedings of the tenth acm conference on learning@ scale*, pages 226–236.

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

OpenAI. 2024a. GPT-4o mini: Advancing cost-efficient intelligence. https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/. Accessed: 2026-05-26.

OpenAI. 2024b. GPT-4o system card. *Preprint*, arXiv:2410.21276.

Shalini Pandey and George Karypis. 2019. A self-attentive model for knowledge tracing. *arXiv preprint arXiv:1907.06837*.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pages 1–22.

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

Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, and 1 others. 2024. Chatdev: Communicative agents for software development. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15174–15186.

Debdeep Sanyal, Agniva Maiti, Umakanta Maharana, Dhruv Kumar, Ankur Mali, C Lee Giles, and Murari Mandal. 2025. Investigating pedagogical teacher and student llm agents: Genetic adaptation meets retrieval augmented generation across learning style. *arXiv preprint arXiv:2505.19173*.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2023. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*.

Zichao Wang, Angus Lamb, Evgeny Saveliev, Pashmina Cameron, Yordan Zaykov, José Miguel Hernández-Lobato, Richard E Turner, Richard G Baraniuk, Craig Barton, Simon Peyton Jones, and 1 others. 2020. Instructions and guide for diagnostic questions: The neurips 2020 education challenge. *arXiv preprint arXiv:2007.12061*.

Ziwei Wang, Jie Zhou, Qin Chen, Min Zhang, Bo Jiang, Aimin Zhou, Qinchun Bai, and Liang He. 2025. Llm-kt: aligning large language models with knowledge tracing using a plug-and-play instruction. *arXiv preprint arXiv:2502.02945*.

Tao Wu, Jingyuan Chen, Wang Lin, Mengze Li, Yumeng Zhu, Ang Li, Kun Kuang, and Fei Wu. 2025. Embracing imperfection: Simulating students with diverse cognitive levels using llm-based agents. *arXiv preprint arXiv:2505.19997*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Yu Yuan, Lili Zhao, Wei Chen, Guangting Zheng, Kai Zhang, Mengdi Zhang, and Qi Liu. 2025. Simulating human-like learning dynamics with llm-empowered agents. *arXiv preprint arXiv:2508.05622*.

Murong Yue, Wenhan Lyu, Jennifer Suh, Yixuan Zhang, and Ziyu Yao. 2024. Mathvc: An llm-simulated multi-character virtual classroom for mathematics education. *arXiv preprint arXiv:2404.06711*.

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.

Zheyuan Zhang, Daniel Zhang-Li, Jifan Yu, Linlu Gong, Jinchang Zhou, Zhiyuan Liu, Lei Hou, and Juanzi Li. 2024. Simulating classroom education with llm-empowered agents. *arXiv preprint arXiv:2406.19226*.

A Experimental Details

A.1 Dataset Details

We conduct our primary experiments on the MathEDU dataset for overall performance prediction, error-type prediction, and instructional intervention trials. It consists of a large number of problem-solving records from six students. The number of records and the corresponding answer correctness of each student are summarized in Table 3. Each record contains the problem and its ground-truth answer, the student's solution process, teacher reviews, etc. An illustrative example is provided in Figure 6.

Table 3: Statistics of Student Performance in MathEDU.

Student	Total	Incorrect	Accuracy
Student1	668	186	72.16%
Student2	685	85	87.59%
Student3	676	194	71.30%
Student4	660	163	75.30%
Student5	652	195	70.09%
Student6	630	99	84.29%

Problem id: 34421
Problem: of the total amount that jill spent on a shopping trip , excluding taxes , she spent 25 percent on clothing , 25 percent on food , and 50 percent on other items . if jill paid a 10 percent tax on the clothing , no tax on the food , and an 2 percent tax on all other items , then the total tax that she paid was what percent of the total amount that she spent , excluding taxes ?

Student id: 1
Student answer: 0.06
Student process: 0.5 \\times 0.1+0.5 \\times 0.02=0.06

Review:
Correct or not: wrong
Error counts: 1
Error type: Comprehension error
Error equation: 0.5 \\times 0.1
Teacher advice: The student did not notice that Jill only paid tax on clothing, so the student incorrectly calculated the tax on food, leading to an error. The correct answer should be 0.25*1+0.5*0.02=0.035=3.5%

Figure 6: Sample from the MathEDU Dataset.

In the extended experiments, we adopt the Nips2020 dataset, a large-scale knowledge tracing dataset containing answer records from 5,310 students. It includes 110 questions and 17 annotated knowledge concepts, together with textual information for both questions and concepts. An example is shown in Figure 7. Since Nips2020 lacks detailed diagnostic information such as problem-solving processes, error types, and teacher feedback, it is used only for overall performance prediction. We randomly selected 50 students for experimentation.

Problem id: 644
Problem: Jo says the diagram above represents 3÷18=6. Paul says the diagram represents 6×3=18 Who is correct ?
A: Neither is correct
B: Only Paul
C: Only Jo
D: Both Jo and Paul
Subject: Maths, Number, Basic Arithmetic, Mental Multiplication and Division
Correct answer: B

Student id: 2216
Student answer: D
Answer time: 2019-10-06 08:26:00.000

Review:
Correct or not: wrong

Figure 7: Sample from the Nips2020 Dataset.

A.2 Metrics Details

In the Error Prediction setting, we evaluate model performance from multiple perspectives, including overall answer outcome prediction and fine-grained error type prediction. The following describes each evaluation metric in detail.

A.2.1 Overall Accuracy

Overall accuracy measures the proportion of student answers whose correctness (correct or incorrect) is predicted correctly. It is defined as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (6)$$

where *TP* and *TN* denote the numbers of true positives and true negatives, *FP* and *FN* represent false positives and false negatives, respectively.

A.2.2 Precision, Recall, and F1-score

We report Precision, Recall, and F1-score to evaluate model performance on both correct answer prediction and incorrect answer prediction. These metrics are computed in a binary classification setting by alternately treating correct answers and incorrect answers as the positive class. Precision is defined as:

$$\text{Precision} = \frac{TP}{TP + FP} \quad (7)$$

where *TP* denotes the number of correctly predicted positive instances and *FP* denotes the number of incorrectly predicted positive instances.

Recall is defined as:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{8}$$

where *FN* denotes the number of positive instances that are incorrectly predicted as negative.

The F1-score is the harmonic mean of Precision and Recall:

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{9}$$

When correct answers are treated as the positive class, these metrics measure the model's ability to identify correct responses. Conversely, when incorrect answers are treated as the positive class, they evaluate the model's effectiveness in detecting student errors.

A.2.3 Error Type Precision Accuracy

Beyond predicting whether an answer is correct, we place particular emphasis on the accuracy of error type prediction. Let $E = \{e_1, e_2, \ldots, e_N\}$ denote the set of student responses with annotated ground-truth error types. The error type prediction accuracy is defined as:

$$\text{ErrorTypeAcc} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(\hat{e}_i = e_i) \tag{10}$$

where $\hat{e}_i$ denotes the predicted error type for the i-th response, e_i is the corresponding ground-truth error type, and $\mathbb{I}(\cdot)$ is the indicator function. Since error types often involve nuanced reasoning and implicit misconceptions, we adopt an LLM-based evaluation protocol to assess whether the predicted error types are semantically consistent with the ground-truth annotations.

A.3 Baselines Details

We compare our proposed method against a comprehensive set of baselines, including five ICL approaches and two SFT methods, to thoroughly evaluate the effectiveness of our proposed `GraphLR-MPP` in predicting student performance.

Among the ICL baselines, we generate predictions by providing different forms of contextual information as input. Specifically, *Entities+Record* is based on the method proposed by Wu et al. (2025), *Similarity* and *Random Record* are comparison baselines introduced in the same work. We additionally introduce the *Entities* setting to isolate the effect of entity-level information without learning records, and the *Learning Report* baseline to compare the use of learning reports in ICL and SFT methods.

- *Similarity*, which retrieves the most semantically similar historical problem-solving record for a given question using text embeddings generated by the `BAAI/bge-m3` model.
- *Entities+Record*, which retrieves the five most relevant entities for a given query and then selecting a historical learning record that covers the maximum number of these entities as the in-context example.
- *Entities*, which uses only the retrieved entities as context, isolating the effect of entity-level information without incorporating any learning records.
- *Random Record*, which randomly selects a record from past learning records as an in-context reference.
- *Learning Report*, which directly uses raw learning reports as context.

Our method fine-tunes the model to predict student performance using learning reports generated via GraphRAG-based clustering. To compare the effectiveness of this approach with other SFT methods, we introduce two SFT baselines.

- SFT (Entities): the model is fine-tuned using the five most relevant entities as input.
- SFT (Record): the model is fine-tuned using a single relevant learning record as input.

A.4 Method Details

A.4.1 Details of Student Cognitive Graph Construction

We provide detailed instructions for extracting entities and relations from student problem solving records, which describe the task goal and specify concrete steps, including identifying knowledge points in the problem and identifying relationships between knowledge points, as well as the required output format.

After extracting all entities and relations from the complete set of student records, we apply the Leiden algorithm to cluster related knowledge points into communities. For each community, we prompt

an LLM to generate a learning report that reflects the student's mastery of the corresponding knowledge points. We further give a detailed description of the task goal and the report structure, which includes a title, a summary of the relevant knowledge points, the student's past learning performance, and an assessment of the student's level of mastery of that knowledge community.

For both tasks, we present representative input and output examples. The complete prompts are shown in Figure 8 and 9, respectively.

```
-Goal-
Given a math problem, the student's answer, solution process, and error
analysis, extract all the related knowledge points and all relationships
among them.
Note: The extracted knowledge points should be common mathematical
concepts, not error types or specific problem instances.

-Steps-
1.Identify knowledge points in the problem.
For each identified knowledge point, extract the following information:
- entity_name: Name of the knowledge point, capitalized.
- entity_type: One of the following types: [{entity_types}], Note:
    a) Concept: Mathematical object or definition (e.g., set, vector).
    b) Theorem: Mathematical principle or law (e.g., Pythagorean theorem).
    c) Method: General problem-solving approach (e.g., substitution
method).
    d) Skill: Computational or operational ability (e.g., arithmetic).
    e) Model: Mathematical abstraction of real-world problems (e.g.,
linear regression model).
- entity_master_level: Assess students' mastery of knowledge points based
on their mistakes. Choose from 'Good' or 'Bad'.
- entity_master_description: Describe how well the student understand and
master this knowledge point using a sentence.
Format each knowledge point as ("entity"{tuple_delimiter}<entity_name>{
tuple_delimiter}<entity_type>{tuple_delimiter}<entity_master_level>{tup
le_delimiter}<entity_master_description>)
2.Identify relationships between knowledge points.
From the entities identified in step 1, identify all pairs of
(source_entity, target_entity) that are *clearly related* to each other.
The relationship should be functional, aiding learners in understanding
the knowledge. We define 4 types of the relations and you must choose one:
    a) Prerequisite_of: One knowledge point is a prerequisite for
learning the other (e.g., understanding limits is required before
learning derivatives)
    b) Equivalent_of: Two concepts describe the same mathematical object
or property (e.g., even numbers = multiples of 2)
    c) Contains_in: One knowledge point is a special case or subset of
another (e.g., a square is a type of rectangle)
    d) Parallel_to: Two knowledge points belong to different branches of
the same category (e.g., area formulas for a circle and a rectangle)
    e) Application_of: A knowledge point is an application of another
(e.g., Discount Rate is an application of Percentage.)
For each pair of related entities, extract the following information:
- source_entity: name of the source entity, as identified in step 1
- target_entity: name of the target entity, as identified in step 1
- relationship_type: type of the relationship, as defined above
- relationship_description: explanation as to why you think the source
entity and the target entity are related to each other
- relationship_strength: a numeric score (0-1) indicating strength of the
relationship between the source entity and target entity
Format each relationship as
("relationship"{tuple_delimiter}<source_entity>{tuple_delimiter}<target
_entity>{tuple_delimiter}<relationshp_type>{tuple_delimiter}<relationsh
ip_description>{tuple_delimiter}<relationship_strength>)
3. Return output in English as a single list of all the entities and
relationships identified in steps 1 and 2. Use **{record_delimiter}** as
the list delimiter.
4. When finished, output {completion_delimiter}

######################
-Examples-
######################
Input:
Problem:
an error 4 % in excess is made while measuring the side of a square . the
percentage of error in the calculated area of the square is

Student Answer:
1.0816

Student Process:
(1+0.04) \\times(1+0.04)=1.0816

Correct or Not:
wrong

Error Analysis:
{{
    "error_counts": 1,
    "error": [
        {{
            "error_type": "Unfinished answer",
            "error_equation": "None",
            "teacher_advice_en": "The student only calculated the
            percentage of error in the measurement of the side of the
            square. The question asks for the percentage of error in the
            calculated area of the square. Therefore, the correct answer
            should be 1.0816 - 1 = 0.0816 = 8.16%."
        }}
    ]
}}

Output:
("entity"{tuple_delimiter}"Percentage Error"{tuple_delimiter}"Concept"{
tuple_delimiter}"Good"{tuple_delimiter}"The student correctly understands
how to calculate the percentage error for a single measurement, such as
the side of a square."){record_delimiter}
("entity"{tuple_delimiter}"Error Propagation in Area"{tuple_delimiter}"
Method"{tuple_delimiter}"Bad"{tuple_delimiter}"The student does not
correctly apply the percentage error to derived quantities like area, as
shown by only computing the side length error and not the area
error."){tuple_delimiter}
("relationship"{tuple_delimiter}"Percentage
Error"{tuple_delimiter}"Error Propagation in
Area"{tuple_delimiter}"Prerequisite_of"{tuple_delimiter}"Understanding
percentage error in a single measurement is necessary before applying
error propagation to derived quantities like
area."{tuple_delimiter}"0.9"){completion_delimiter}

######################
-Real Data-
######################
Input:
{input_text}

Output:
```

Figure 8: Prompt for extracting entities and relations.

```
You are an AI assistant that helps a human teacher generate students'
Learning Report.

# Goal
Given a set of entities and their relationships, where each entity
represents a knowledge point and the student's mastery of it, and
relationships describe the connections between these knowledge points.
Your task is to write a report analyzing the student's understanding of
this section of content.
This report is intended to provide teachers with a systematic evaluation
of the student's learning performance at this stage.

# Report Structure
This report should include the following sections:
- Title: Learning Report on the Student's Learning of <xx>. Replace xx
with a concise 2~3 word summary of the relevant knowledge category.
- KC Summary: Provide a summary of the key knowledge points in this
section and briefly explain how they are interconnected.
- Exercise Performance: Describe the student's performance on exercises
related to these knowledge points, including the overall accuracy rate,
detailed results for each specific knowledge point, and a classification
of the main types of errors (e.g., Comprehension errors, Wrong
mathematical operations/concepts, and Arithmetical mistakes, etc.).
- Student Mastery: Based on the exercise performance, write one or two
concise sentences summarizing the student's overall mastery of this
knowledge section.

Return output as a well-formed JSON-formatted string with the following
format:
    {{
        "title": ,
        "kc_summary": ,
        "Exercise Performance": [
            {{
                "Correctness Rate":
                "Detailed Analysis":
                "Error Types":
            }}
        ]
        "Student Mastery":
    }}


######################
-Examples-
######################
Text:
```
Entities:
```csv
id,entity,type,description
1,DISCOUNT RATE,concept,The student correctly understands the concept of
discount rate and its application in calculating savings
2,TOTAL SAVINGS CALCULATION,method,The student correctly calculates the
total savings by comparing the discounted price to the original price
3,SALES DISCOUNT,concept,The student correctly identifies that the total
savings come from the discounts applied to the jeans
4,ORIGINAL PRICE,concept,The student correctly uses the original prices
of the jeans in the calculations

```
Relationships:
```csv
id,source,target,type,description
1,DISCOUNT RATE,TOTAL SAVINGS CALCULATION,Prerequisite_of,Both concepts
deal with the idea of reduction in cost, just from different perspectives:
total savings vs. individual discount rate
2,DISCOUNT RATE,SALES DISCOUNT,Application_of,The sales discount is a
form of discount rate applied to the jeans
3,DISCOUNT RATE,ORIGINAL PRICE,Parallel_to,The discount rate is applied
to the original price of the jeans
```
```
```

```
```
Output:
{{
    "title": "Learning Report on the Student's Learning of <Sales
    Discounts and Savings>",
    "kc_summary": "This section covers four key knowledge points: Discount
    Rate, Original Price, Sales Discount, and Total Savings Calculation.
    The Discount Rate is a prerequisite concept for understanding both
    Sales Discount and Total Savings Calculation. The Original Price is
    used as the base value to apply the Discount Rate, forming the Sales
    Discount. The Total Savings Calculation applies the concept of the
    Discount Rate and Sales Discount to determine the overall savings.
    These relationships show a logical progression: Discount Rate and
    Original Price → Sales Discount → Total Savings Calculation.",
    "Exercise Performance": [
        {{
            "Correctness Rate": "100%",
            "Detailed Analysis": "The student correctly identified the
            discount rate, applied it to the original price, and
            accurately calculated the sales discount and total savings
            for all exercises.",
            "Error Types": "None observed."
        }}
    ],
    "Student Mastery": "The student demonstrates a solid understanding of
    the relationships among discount rate, sales discount, original
    price, and total savings, and can accurately apply these concepts in
    calculations."
}}

######################
- Real Data-
######################
Use the following text for your answer. Do not make anything up in your
answer.

Text:
```
{input_text}
```

Output:
```

Figure 9: Prompt for Generating Learning Report.

### A.4.2 Details of Instructional Intervention Strategies

We select *Explicit Instruction* and *Metacognitive Strategies* as the two instructional intervention strategies for multi-agent simulation experiments. *Explicit, Systematic Instruction* is organized into four steps. The name and description of each step are as follows:

- Step1. Orientation to the Lesson. Connect this question to prior learning and review previously taught concepts.
- Step2. Initial Instruction. Point out the difficult aspects of the problem and check students' understanding of them.
- Step3. Teacher-Guided Practice. Provide scaffolding and guide the student through solving the problem step by step. Avoid giving the full answer at once.
- Step4. Independent Practice. Have students re-solve the initial problem independently and check their answers.

*Metacognitive Strategies* is organized into three steps. The name and description of each step are as follows:

- Step1. Plan. Guide students to analyze the problem, identify what the problem is asking, and plan a strategy to solve the problem by themself."
- Step2. Monitor. Prompt students to check their steps as they work and verify that their approach and reasoning make sense."
- Step3. Modify. Support students in revising their plan or choosing an alternative strategy if their current approach is ineffective, or if their answer seems unreasonable. Encourage students to explain why a modification is needed.

# B More Experimental Results and Analysis

## B.1 Complete Student Performance Prediction Results on MathEDU

### B.1.1 Overall Prediction Results

The complete per-student prediction results of our GraphLR-MPP method compared with five ICL baselines and two additional SFT baselines on the MathEDU dataset are reported in Tables 4–9. As shown, GraphLR-MPP consistently achieves the highest overall accuracy across all six students, demonstrating strong robustness at the individual level.

To further address whether the comparison is affected by different LLM families, we conduct additional ICL experiments with Qwen2.5-7B-Instruct, Claude-4-Sonnet, and Gemini-2.0-Flash as backbones. As shown in Table 10, GraphLR-MPP still achieves the best overall accuracy and remains competitive across most metrics. Although some ICL baselines obtain higher recall or F1 on wrong-answer prediction, they often sacrifice performance on correct-answer prediction, leading to lower overall accuracy. These results further confirm that the advantage of GraphLR-MPP is not limited to a specific LLM family.

### B.1.2 Error-type Prediction Results

Error-type prediction is a key aspect of our analysis, as it directly reflects the model's ability to capture nuanced student misconceptions. Table 11 reports the accuracy of error-type consistency for each of the six students achieved by GraphLR-MPP and all baseline methods. To provide a more intuitive comparison, we further visualize the results using a heatmap in Figure 10. As shown, our method significantly outperforms the other approaches on almost all students, highlighting its superior capability in fine-grained error understanding.
```

Methods		SFT			ICL									
		GraphLR-MPP	Entities	Record	Similarity		Entities+Record		Entities		Random Record		Learning Report	
Base Model		Qwen2.5-7B-Instruct			4o-mini	4o	4o-mini	4o	4o-mini	4o	4o-mini	4o	4o-mini	4o
Overall	Acc.	**0.71**	0.645	0.655	0.649	0.595	0.615	0.683	0.6	0.688	0.645	0.634	0.551	0.439
Correct	Prec.	**0.765**	0.713	0.705	0.746	0.706	0.709	0.739	0.697	0.729	0.727	0.731	0.74	0.729
	Rec.	0.845	0.824	**0.873**	0.746	0.711	0.754	0.838	0.746	**0.873**	0.789	0.746	0.542	0.303
	F1	**0.803**	0.765	0.78	0.746	0.709	0.73	0.785	0.721	0.795	0.757	0.739	0.626	0.428
Wrong	Prec.	**0.519**	0.359	0.333	0.429	0.339	0.352	0.477	0.321	0.486	0.388	0.4	0.356	0.322
	Rec.	0.392	0.23	0.148	0.429	0.333	0.302	0.333	0.27	0.27	0.311	0.381	0.571	**0.746**
	F1	0.446	0.28	0.205	0.429	0.336	0.325	0.393	0.293	0.347	0.345	0.39	0.439	**0.45**

Table 4: Comparison of methods in predicting `Student1`'s performance. Best results are in bold, second best are underlined.

Methods		SFT			ICL									
		GraphLR-MPP	Entities	Record	Similarity		Entities+Record		Entities		Random Record		Learning Report	
Base Model		Qwen2.5-7B-Instruct			4o-mini	4o	4o-mini	4o	4o-mini	4o	4o-mini	4o	4o-mini	4o
Overall	Acc.	**0.845**	0.801	0.825	0.757	0.728	0.825	0.83	0.825	0.83	0.796	0.796	0.607	0.447
Correct	Prec.	0.901	0.896	0.89	0.914	0.901	0.903	0.908	0.903	0.908	**0.918**	0.909	0.888	0.908
	Rec.	**0.93**	0.881	0.919	0.805	0.784	0.903	0.903	0.903	0.903	0.849	0.859	0.643	0.427
	F1	**0.915**	0.888	0.904	0.856	0.838	0.903	0.905	0.903	0.905	0.882	0.883	0.746	0.581
Wrong	Prec.	0.133	0.083	0	0.163	0.111	0.143	0.182	0.143	0.182	**0.2**	0.161	0.083	0.109
	Rec.	0.095	0.095	0	0.333	0.238	0.143	0.19	0.143	0.19	0.333	0.238	0.286	**0.619**
	F1	0.111	0.089	0	0.219	0.152	0.143	0.186	0.143	0.186	**0.25**	0.192	0.129	0.186

Table 5: Comparison of methods in predicting `Student2`'s performance. Best results are in bold, second best are underlined.

Methods		SFT			ICL									
		GraphLR-MPP	Entities	Record	Similarity		Entities+Record		Entities		Random Record		Learning Report	
Base Model		Qwen2.5-7B-Instruct			4o-mini	4o	4o-mini	4o	4o-mini	4o	4o-mini	4o	4o-mini	4o
Overall	Acc.	**0.725**	0.667	0.676	0.706	0.676	0.691	0.696	0.686	0.681	0.627	0.672	0.461	0.48
Correct	Prec.	0.76	0.723	0.718	0.789	0.748	0.753	0.755	0.752	0.756	0.714	0.753	0.727	**0.791**
	Rec.	**0.897**	0.862	**0.897**	0.8	0.821	0.841	0.848	0.834	0.814	0.793	0.8	0.386	0.366
	F1	**0.823**	0.786	0.789	0.795	0.783	0.795	0.799	0.791	0.784	0.752	0.776	0.505	0.5
Wrong	Prec.	**0.545**	0.355	0.348	0.491	0.422	0.452	0.463	0.442	0.438	0.302	0.42	0.299	0.328
	Rec.	0.305	0.186	0.136	0.475	0.322	0.322	0.322	0.322	0.356	0.22	0.356	0.644	**0.763**
	F1	0.391	0.244	0.195	**0.483**	0.365	0.376	0.38	0.373	0.393	0.225	0.385	0.409	0.459

Table 6: Comparison of methods in predicting `Student3`'s performance. Best results are in bold, second best are underlined.

Methods		SFT			ICL									
		GraphLR-MPP	Entities	Record	Similarity		Entities+Record		Entities		Random Record		Learning Report	
Base Model		Qwen2.5-7B-Instruct			4o-mini	4o	4o-mini	4o	4o-mini	4o	4o-mini	4o	4o-mini	4o
Overall	Acc.	**0.698**	0.593	0.633	0.673	0.618	0.648	0.673	0.668	0.658	0.653	0.658	0.603	0.387
Correct	Prec.	0.754	0.75	0.726	0.811	0.793	0.752	0.763	0.765	0.745	0.744	0.776	0.854	**0.857**
	Rec.	**0.878**	0.673	0.81	0.728	0.653	0.782	0.81	0.796	0.816	0.81	0.755	0.558	0.204
	F1	**0.811**	0.71	0.765	0.767	0.716	0.767	0.785	0.78	0.779	0.775	0.766	0.675	0.33
Wrong	Prec.	0.357	0.284	0.2	**0.403**	0.346	0.304	0.349	0.348	0.289	0.282	0.357	0.369	0.287
	Rec.	0.192	0.365	0.135	0.519	0.519	0.269	0.288	0.308	0.212	0.212	0.385	0.731	**0.904**
	F1	0.25	0.319	0.161	0.454	0.415	0.286	0.316	0.327	0.244	0.242	0.37	**0.49**	0.435

Table 7: Comparison of methods in predicting `Student4`'s performance. Best results are in bold, second best are underlined.

Methods		SFT			ICL									
		GraphLR-MPP	Entities	Record	Similarity		Entities+Record		Entities		Random Record		Learning Report	
Base Model		Qwen2.5-7B-Instruct			4o-mini	4o	4o-mini	4o	4o-mini	4o	4o-mini	4o	4o-mini	4o
Overall	Acc.	**0.707**	0.608	0.656	0.649	0.683	0.605	0.62	0.649	0.649	0.62	0.566	0.588	0.629
Correct	Prec.	**0.699**	0.649	0.671	0.638	0.673	0.622	0.626	0.642	0.643	0.623	0.591	0.676	0.685
	Rec.	0.855	0.821	**0.889**	**0.889**	0.863	0.786	0.829	0.872	0.863	0.846	0.778	0.61	0.65
	F1	**0.769**	0.725	0.765	0.743	0.757	0.694	0.713	0.739	0.737	0.717	0.672	0.641	0.667
Wrong	Prec.	**0.726**	0.447	0.581	0.69	0.709	0.561	0.6	0.674	0.667	0.609	0.490	0.484	0.564
	Rec.	0.511	0.246	0.261	0.33	0.443	0.364	0.341	0.352	0.364	0.318	0.284	0.556	**0.602**
	F1	**0.6**	0.318	0.36	0.446	0.545	0.441	0.435	0.463	0.471	0.418	0.36	0.517	0.582

Table 8: Comparison of methods in predicting `Student5`'s performance. Best results are in bold, second best are underlined.

Methods		SFT			ICL									
		GraphLR-MPP	Entities	Record	Similarity		Entities+Record		Entities		Random Record		Learning Report	
Base Model		Qwen2.5-7B-Instruct			4o-mini	4o	4o-mini	4o	4o-mini	4o	4o-mini	4o	4o-mini	4o
Overall	Acc.	**0.804**	0.749	0.794	0.739	0.709	0.794	0.774	0.779	0.754	0.678	0.678	0.608	0.427
Correct	Prec.	0.84	0.825	0.823	**0.898**	0.893	0.842	0.83	0.84	0.839	0.817	0.836	0.865	0.833
	Rec.	0.922	0.857	**0.935**	0.747	0.708	0.903	0.89	0.883	0.844	0.753	0.727	0.584	0.325
	F1	**0.879**	0.841	0.875	0.816	0.79	0.871	0.859	0.861	0.841	0.784	0.778	0.698	0.467
Wrong	Prec.	**0.6**	0.436	0.583	0.451	0.416	0.559	0.5	0.514	0.455	0.333	0.354	0.326	0.252
	Rec.	0.4	0.378	0.311	0.711	0.711	0.422	0.378	0.422	0.444	0.422	0.511	0.689	**0.778**
	F1	0.48	0.405	0.406	**0.552**	0.525	0.481	0.43	0.463	0.449	0.373	0.418	0.443	0.38

Table 9: Comparison of methods in predicting `Student6`'s performance. Best results are in bold, second best are underlined.

Table 10: Extended experimental results with additional LLM backbones on the `MathEDU` dataset. Qwen, Claude, and Gemini denote Qwen2.5-7B-Instruct, Claude-4-Sonnet, and Gemini-2.0-Flash, respectively. The best result in each row is highlighted in bold.

Category	Metric	GraphLR-MPP	ICL-Similarity			ICL-Entities+Record			ICL-Entities			ICL-Random Record			ICL-Learning Report		
		Qwen	Qwen	Claude	Gemini	Qwen	Claude	Gemini	Qwen	Claude	Gemini	Qwen	Claude	Gemini	Qwen	Claude	Gemini
Overall	Acc.	**0.748**	0.660	0.626	0.635	0.650	0.626	0.645	0.631	0.611	0.596	0.650	0.567	0.660	0.601	0.350	0.365
Correct	Prec.	0.787	0.716	**0.844**	0.774	0.715	0.811	0.750	0.701	0.789	0.717	0.729	0.821	0.770	0.707	0.812	0.842
	Rec.	**0.888**	0.825	0.570	0.676	0.831	0.606	0.739	0.824	0.606	0.697	0.796	0.486	0.732	0.732	0.092	0.113
	F1	**0.833**	0.778	0.681	0.722	0.769	0.694	0.745	0.757	0.685	0.707	0.761	0.611	0.751	0.720	0.165	0.199
Wrong	Prec.	**0.480**	0.382	0.430	0.418	0.368	0.423	0.413	0.306	0.404	0.338	0.396	0.387	0.441	0.321	0.310	0.315
	Rec.	0.316	0.213	0.754	0.541	0.230	0.672	0.426	0.180	0.623	0.361	0.311	0.754	0.492	0.295	**0.951**	**0.951**
	F1	0.380	0.274	**0.548**	0.471	0.283	0.519	0.419	0.227	0.490	0.349	0.349	0.511	0.465	0.308	0.468	0.473

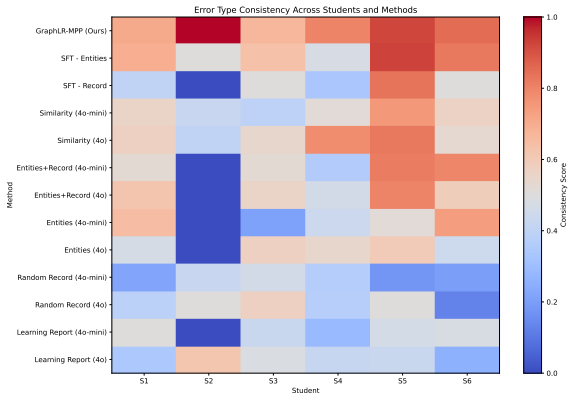

Figure 10: Heatmap of Error Type Predictions for Six Students.

Methods	S1	S2	S3	S4	S5	S6
SFT (Qwen2.5-7B-Instruct)						
GraphLR-MPP	**0.706**	**1.000**	**0.667**	**0.800**	0.921	**0.850**
Entities	0.692	0.500	0.636	0.474	**0.929**	0.824
Record	0.400	0.000	0.500	0.333	0.833	0.500
ICL (GPT-4o mini)						
Similarity	0.556	0.429	0.393	0.519	0.750	0.563
Entities+Record	0.526	0.000	0.526	0.357	0.813	0.790
Entities	0.647	0.000	0.211	0.438	0.516	0.737
Random Record	0.222	0.500	0.462	0.364	0.179	0.200
Learning Report	0.500	0.000	0.432	0.286	0.467	0.478
ICL (GPT-4o)						
Similarity	0.571	0.400	0.546	0.778	0.822	0.531
Entities+Record	0.619	0.000	0.556	0.462	0.800	0.588
Entities	0.471	0.429	0.571	0.546	0.594	0.444
Random Record	0.385	0.000	0.571	0.368	0.500	0.129
Learning Report	0.340	0.615	0.487	0.426	0.432	0.257

Table 11: Comparison of Error Type Consistency Across Different Methods for Six Students, S1–S6 denote Student1-6.

B.2 Prediction Performance Across Varying Student Correctness Levels

To further investigate how students' actual answer correctness influences model performance, we analyze the prediction accuracy across varying student correctness levels.

As shown in Figure 11, most baseline methods exhibit a clear U-shaped trend: prediction accuracy tends to be lower for medium-correctness students and improves when student performance is either very low or very high. In contrast, our GraphLR-MPP method consistently achieves higher prediction accuracy than the baselines across the entire range.

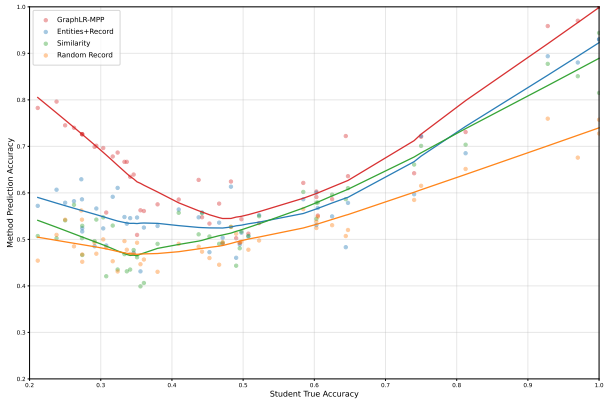

Figure 11: Prediction Accuracy of Different Methods under Different Levels of Student Answer Correctness.

To enable a more detailed analysis of GraphLR-MPP, we group students into four levels based on their overall correctness rate: Low (<0.45), Medium-Low (0.45–0.55), Medium-High (0.55–0.65), and High (>0.65). Figure 12 further shows that GraphLR-MPP maintains relatively stable and superior performance across all four groups, with particularly strong results in the high-correctness group. Overall, these results indicate that our method is more robust to varying student ability lev-

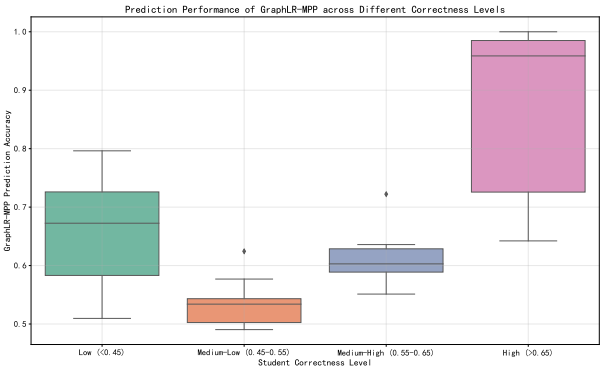

Figure 12: Prediction Performance of GraphLR-MPP across Different Student Correctness Levels.

els compared to existing SFT and ICL approaches.

B.3 Intervention Results

B.3.1 Intervention case

Here, we provide examples of teachers employing Explicit, Systematic Instruction and Metacognitive Strategies to guide students in re-solving questions they previously answered incorrectly, as shown in Figures 14 and 15, respectively.

B.3.2 Cognitive Graph Update

We visualize the complete cognitive graph of `Student1` before and after the metacognitive strategy intervention in Figure 13. The proportion of green nodes is noticeably higher than that of red nodes, indicating an improvement in the student's cognitive level after the intervention.

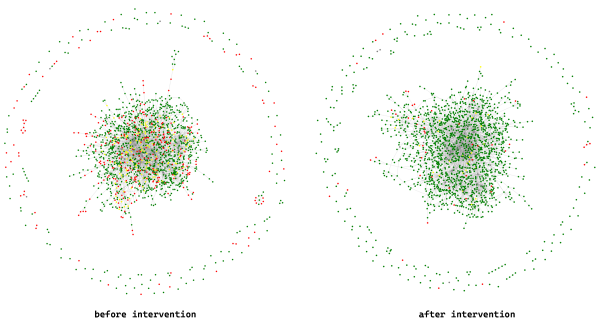

Figure 13: Complete Cognitive Graphs of Students Before and After Metacognitive Strategy Intervention. Red indicates not mastered, green indicates mastered, and yellow indicates partially mastered.

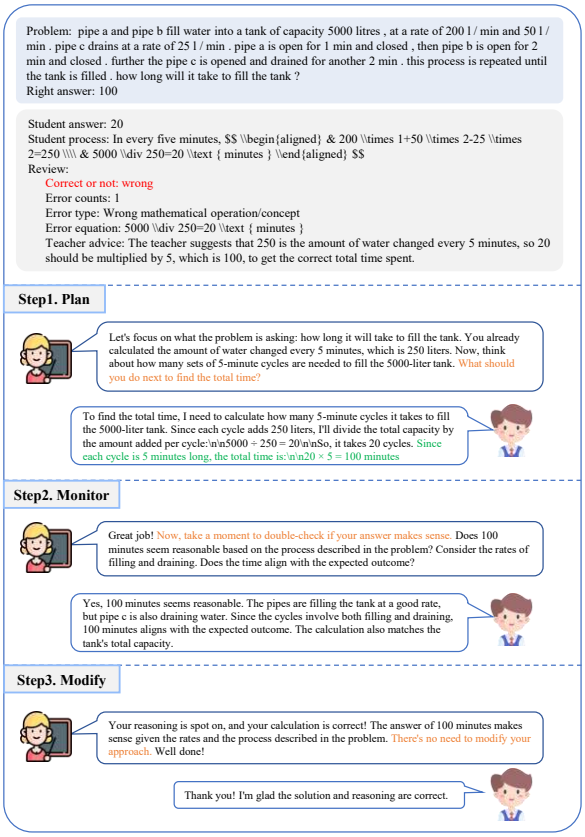

Figure 14: A Case of Teacher-Led *Metacognitive Strategy Intervention* in Students' Mathematical Problem Solving

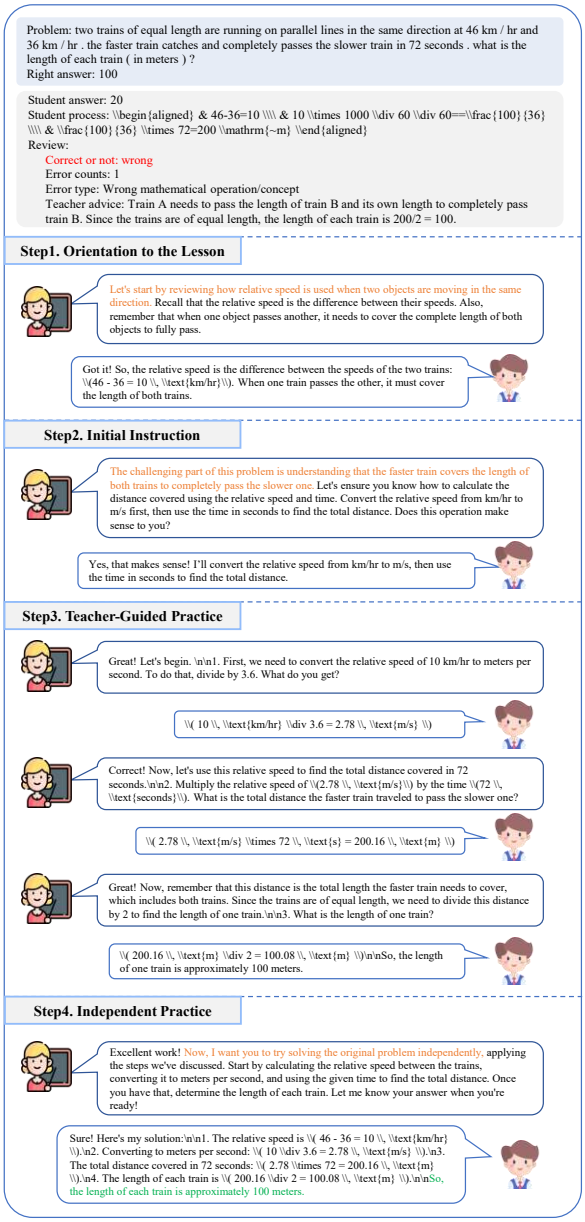

Figure 15: A Case of Teacher-Led *Explicit, Systematic Instruction* in Students' Mathematical Problem Solving