Beyond Pedagogical Principles: Multi-Horizon Preference Optimization for Efficient Socratic Tutoring

Anonymous ACL submission

Abstract

The development of LLM-based tutor agents faces challenges in simultaneously ensuring adherence to pedagogical principles and achieving optimal pedagogical effectiveness, particularly in dynamic, multi-turn interactions. Existing methods are often constrained by static data or sparse reward signals in online settings. To address this gap, we propose **M**ulti-**H**orizon **P**reference **O**ptimization (**MHPO**), a novel framework that iteratively refines tutor agents using a multi-horizon reward function within a dynamic teacher-student simulation environment. Specifically, this reward function is designed to capture both turn-level pedagogical quality and trajectory-level pedagogical effectiveness, which is estimated via Monte Carlo rollouts. We further investigate two distinct strategies to aggregate these rewards for policy optimization. Our experiments demonstrate that MHPO significantly enhances base model performance, achieving a superior balance between principles and effectiveness compared to various baselines.

1 Introduction

Given their powerful capabilities, Large Language Models (LLMs) are increasingly being applied across the educational landscape (Cheng et al., 2024; Liu et al., 2024b; Zhang et al., 2025; Chu et al., 2025). A prominent area of this research focuses on developing LLMs as personalized tutor agents. The goal is to create tutor agents that achieve pedagogical alignment—acting not as answer providers, but as guides (Liu et al., 2024a; Ding et al., 2024; Kargupta et al., 2024). A primary technique for this is Socratic questioning, a form of educational scaffolding where a sequence of carefully crafted questions prompts students to navigate their own path to understanding (Wood et al., 1976; Shridhar et al., 2022). This interactive process is highly valued for its ability to foster critical thinking and genuine self-discovery.

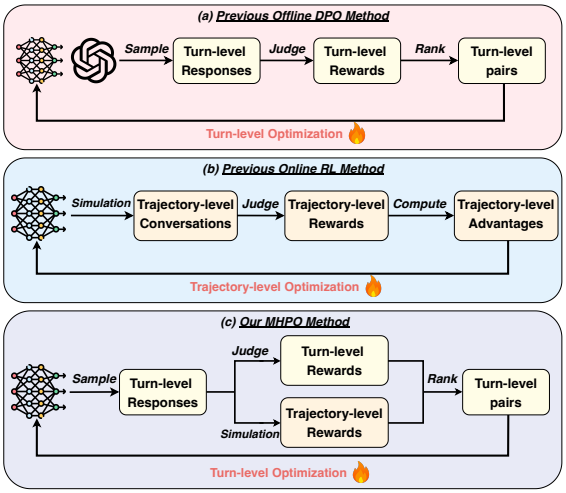

Figure 1: Overview of RL paradigms for tutor agent. Unlike previous methods (a) offline DPO and (b) online RL, our MHPO method (c) integrates multi-horizon reward while employing fine-grained turn-level optimization.

The ultimate goal of tutoring is to improve learning outcomes and critical thinking (Cohen et al., 1982; Ritter et al., 2007). Expert educators achieve this with not only sound pedagogy but also remarkable efficiency. They can precisely focus on the most critical points and provide targeted guidance that accelerates understanding (Anghileri, 2006; Paul and Elder, 2007). Consequently, true pedagogical effectiveness must be gauged by both the learning outcome and the efficiency of the process—a holistic standard exemplified by human expertise. However, many current AI tutors prioritize strict adherence to pedagogical principles while neglecting interactional efficiency.

A foundational and widely adopted method for training LLM-based tutor agents is Supervised Fine-Tuning (SFT) (Macina et al., 2023; Liu et al., 2024a; Ding et al., 2024). However, this method struggles with adaptability because its reliance on static datasets cannot prepare the agent for the dy-

namic and unpredictable nature of live student interactions. To explicitly align the tutor agent's policy with pedagogical principles, recent work has turned to offline preference optimization techniques like Direct Preference Optimization (DPO) (Sonkar et al., 2024; Scarlatos et al., 2025; Kumar and Lan, 2024; Rafailov et al., 2023). These methods typically construct turn-level preference pairs by using responses generated from different tutor agents, which are then evaluated and compared against predefined pedagogical criteria. However, this paradigm introduces significant limitations: it often relies on external, more powerful models for generating high-quality data and lacks feedback on the final outcome of an interactive dialogue. To address this lack of interactive feedback, an online RL approach has recently been introduced to optimize the tutor agent through interaction with a simulated student (Dinucu-Jianu et al., 2025). However, the critical challenge of this online paradigm is its reliance on sparse, trajectory-level rewards. This design evaluates the dialogue only as a whole, making it impossible to measure how a turn-level response impacts the final student outcome, and creating a severe credit assignment problem.

In this paper, we propose **M**ulti-**H**orizon **P**reference **O**ptimization (**MHPO**) framework to jointly optimize for both pedagogical principles and pedagogical effectiveness. Our framework achieves this by optimizing the tutor agent with a multi-horizon reward function derived from interactions within a dynamic teacher-student simulation environment. Specifically, this reward function is designed to achieve dual objectives by combining two complementary components. The first is a short-term, turn-level reward, where an external LLM-as-a-Judge assesses immediate pedagogical quality. The second is a long-term, trajectory-level reward. In contrast to previous methods, this long-term component, estimated via Monte Carlo rollouts within our simulation, allows us to explicitly estimate how a turn-level response will impact the overall pedagogical effectiveness. Furthermore, we explore two distinct strategies to fuse these multi-dimensional rewards: Linear Scalarization and Strict Dominance. Based on this fused reward, we then optimize the tutor agent's policy using an iterative DPO framework, a scheme designed to mitigate distribution shift and ensure continuous improvement.

We evaluate our MHPO framework using two backbone models on the MathDial dataset(Macina et al., 2023). The results demonstrate that our method significantly enhances the base models' performance, outperforming baselines by balancing pedagogical principles and pedagogical effectiveness. Moreover, we conduct further analysis to validate the efficacy of our framework and provide a deeper understanding of our two reward aggregation strategies. In summary, our main contributions are:

- We propose MHPO, a novel framework that integrates a dynamic teacher-student simulation and a multi-horizon reward function to jointly optimize for pedagogical principles and pedagogical effectiveness.
- Experiments demonstrate that MHPO consistently enhances base models and outperforms baselines by effectively balancing pedagogical principles and pedagogical effectiveness.
- We provide a comprehensive analysis of two distinct reward aggregation strategies, offering critical insights into their performance characteristics and the trade-offs between data quantity and quality.

2 Related Work

LLM-based Tutor Agents. Early Intelligent Tutoring Systems (ITS) provided instruction but were often rule-based, lacking the generative capabilities for open-ended dialogue (Graesser et al., 2001; Aleven et al., 2003; Graesser et al., 2005; Wollny et al., 2021). The advent of Large Language Models (LLMs) has fundamentally transformed this landscape, making it possible for the first time to create tutor agents capable of fluent, dynamic, and pedagogically rich interactions (Dinucu-Jianu et al., 2025; Liu et al., 2025).

A primary approach to deploying LLMs as tutor agents involves constructing scaffolded prompts. For instance, the CLASS framework (Sonkar et al., 2023) utilizes problem decomposition, step-by-step scaffolding, and error diagnosis within its prompts. Similarly, Bridge (Wang et al., 2024) guides the LLM's tutoring by defining strategies and intents from student errors. Furthermore, many approaches employ Supervised Fine-Tuning (SFT) with high-quality, curated data to enhance the tutor agent's pedagogical capabilities (Macina et al., 2023; Liu et al., 2024a).

Reinforcement Learning (RL) has emerged as a promising approach to further enhance tutor agents'

capabilities. One paradigm employs offline preference optimization techniques like Direct Preference Optimization (DPO) (Sonkar et al., 2024; Scarlatos et al., 2025). These methods typically refine a policy using static preference data. To incorporate the interactive feedback missing from static datasets, an online RL approach has also been explored, where the agent learns directly from interaction with a simulated student (Dinucu-Jianu et al., 2025). However, while these prior methods have made significant strides in aligning agents with pedagogical principles, they have largely overlooked the critical dimension of interactional efficiency.

Reinforcement Learning for LLM. RL was first introduced to the training of LLMs for aligning them with human preferences (Ouyang et al., 2022). This foundational approach uses a reward model to guide policy optimization via an online algorithm, typically Proximal Policy Optimization (PPO) (Schulman et al., 2017). More recently, novel online RL algorithms like Group Relative Policy Optimization (GRPO) have been proposed to enhance reasoning capabilities (Shao et al., 2024; Yu et al., 2025). While these online methods have achieved significant success in verifiable domains, they often suffer from a critical credit assignment problem when applied to long-trajectory, multi-turn interactions (Yu et al., 2024; Dong et al., 2025; Dinucu-Jianu et al., 2025).

As a simpler and more stable offline method, DPO bypasses the need for an explicit reward model, thus achieving significant success across various domains by directly optimizing policies on pairwise preference data (Rafailov et al., 2023; Kumar and Lan, 2024; Scarlatos et al., 2025). The DPO framework continues to be refined, with recent variants like SimPO removing the reference model and ORPO integrating instruction tuning (Meng et al., 2024; Hong et al., 2024).

Despite its success in training tutor agents, the typical application of DPO relies on static preference data that is distilled from powerful external models and lacks interactive feedback from students (Sonkar et al., 2024; Scarlatos et al., 2025). In contrast to these prior works, our method integrates a dynamic simulation environment into the preference optimization loop, shaping a multi-horizon reward to enable the joint optimization of turn-level pedagogical principles and trajectory-level pedagogical effectiveness.

3 Method

Figure 2 illustrates our proposed **MHPO** framework. It begins with the initialization of both the tutor and student agents (Sec. 3.1). Subsequently, we sample candidate responses and compute their multi-horizon rewards (Sec. 3.2). Finally, we perform iterative policy optimization via Direct Preference Optimization (DPO) to refine the tutor agent (Sec. 3.3).

3.1 Interaction Framework Initialization

We formulate the tutoring process as a goal-oriented dialogue. Each dialogue is grounded in a specific background $\mathcal{B}$ (e.g., a problem statement and student profile) and is driven by a clear goal $\mathcal{G}$: guiding the student to a correct and comprehensive solution. The dialogue comprises a sequence of turns, indexed by t. Each turn consists of a student utterance u_t and a tutor response a_t. The tutor agent's policy, denoted as π_θ, aims to generate pivotal responses to achieve goal $\mathcal{G}$ efficiently. At turn t, the policy observes the current state $s_t = (\mathcal{B}, C_t)$, where $C_t = (u_0, a_0, \ldots, u_{t-1}, a_{t-1}, u_t)$ represents the dialogue history, and then generates the next action a_t:

$$a_t \sim \pi_\theta(\cdot \mid s_t). \tag{1}$$

Tutor Agent Initialization. We initialize the tutor agent with foundational pedagogical skills through a sequential two-stage supervised fine-tuning (SFT) process. We first fine-tune the base language model on a specialized dataset D_{socratic}. This stage enables the model to generate Socratic responses that productively guide students towards a solution without directly revealing the answer. Subsequently, the resulting model is further fine-tuned on D_{judge} to enhance its judgment capability for accurately verifying student solutions. This ability is critical for identifying a successful dialogue outcome. Each stage optimizes the policy parameters θ by minimizing the standard autoregressive loss on its respective dataset:

$$\mathcal{L}_{\text{SFT}}(\theta; \mathcal{D}) = -\mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\log \pi_\theta(y|x)\right]. \tag{2}$$

Student Agent Simulation. To facilitate iterative reinforcement of the tutor policy, we introduce a student agent π_ϕ to provide feedback within a simulated environment. The student agent is fine-tuned on $D_{simulation}$ to simulate a diverse range of student proficiency levels, thereby providing realistic and challenging interactions. Given the current

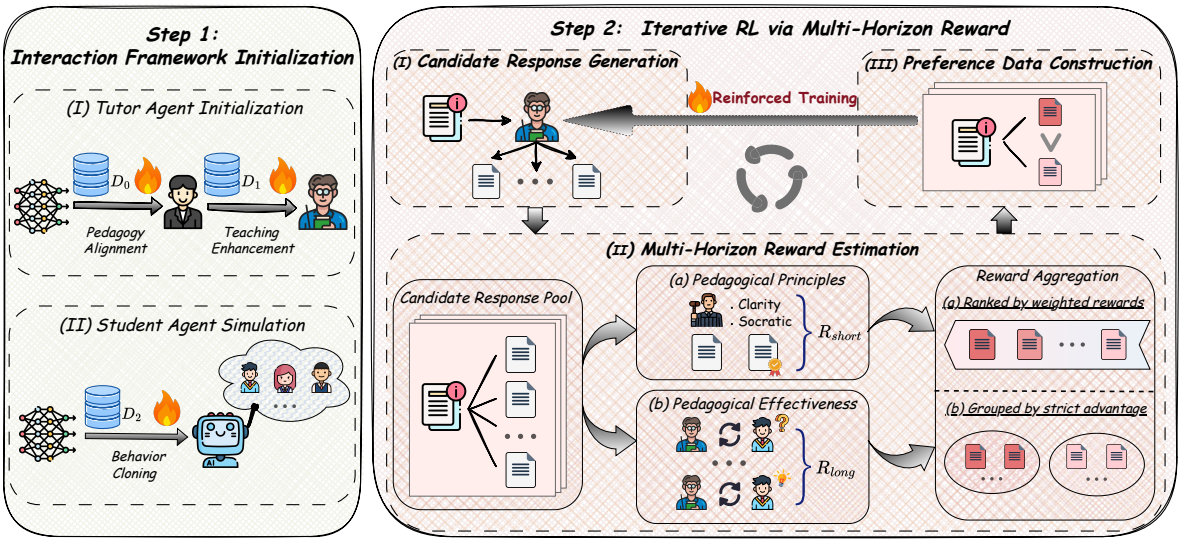

Figure 2: The overall pipeline of our MHPO framework. This iterative RL framework begins by initializing tutor and student agents. It then continuously refines the tutor agent through DPO, utilizing preference data derived from multi-horizon rewards within a dynamic simulation environment.

state s_t and the tutor agent's action a_t, the student agent generates its response u_{t+1} as the environment's feedback:

$$u_{t+1} \sim \pi_\phi(\cdot \mid s_t, a_t). \tag{3}$$

This feedback drives the transition of the dialogue to the next state s_{t+1}.

3.2 Multi-horizon Reward Modeling

To effectively guide the tutor agent's policy, we formulate a multi-horizon reward function $\mathcal{R}$ that evaluates candidate actions by combining two complementary components: (1) a short-term, turn-level reward $\mathcal{R}_{\text{short}}$ reflecting adherence to pedagogical principles, and (2) a long-term, trajectory-level reward $\mathcal{R}_{\text{long}}$ measuring overall pedagogical effectiveness, which encompasses both student outcomes and dialogue efficiency. For a given dialogue state s_t, we first sample a set of N distinct candidate responses $\{a_t^{(i)}\}_{i=1}^N$ from the tutor agent's policy π_θ. Each candidate $a_t^{(i)}$ is then evaluated using the reward function $\mathcal{R}(s_t, a_t^{(i)})$.

Turn-level Reward. This reward component $\mathcal{R}_{\text{short}}(s_t, a_t^{(i)})$ provides immediate feedback on pedagogical quality by evaluating the tutor agent's action from the perspective of a skilled educator. Inspired by the recent success of the LLM-as-a-Judge paradigm, we estimate this value by prompting a powerful external LLM to quantify the response's alignment with established pedagogical principles. The judge specifically assesses aspects such as clarity, conciseness, and the effectiveness of Socratic scaffolding that avoids revealing the final solution. Detailed prompt can be found in Appendix E.

Trajectory-level Reward. The second component $\mathcal{R}_{\text{long}}(s_t, a_t^{(i)})$ is introduced to measure the expected value of the pedagogical effectiveness given a candidate response $a_t^{(i)}$. We approximate this value via Monte Carlo rollouts, leveraging the interactive simulation between the tutor agent and student agent. Specifically, for each state-action pair $(s_t, a_t^{(i)})$, we simulate M dialogue trajectories. Each trajectory continues until the problem is successfully resolved or a maximum turn limit T_{max} is reached. The reward $\mathcal{R}_{\text{long}}$ is then computed by penalizing the mean number of turns from the current state to the end of each trajectory, thereby capturing both student outcome and dialogue efficiency.

Reward Aggregation. Having defined the short-term and long-term reward components, which can be represented as a vector $\mathbf{r} = [\mathcal{R}_{\text{short}}, \mathcal{R}_{\text{long}}]$, we aggregate this vector into a unified reward $\mathcal{R}$, using one of two distinct strategies: Linear Scalarization (LS) or Strict Dominance (SD).

LS adopts a trade-off-based view. It projects the vector $\mathbf{r}$ onto a single scalar reward, allowing a high score in one dimension to compensate for a lower score in another. The aggregated reward is computed as a weighted sum:

$$\mathcal{R} = \mathcal{R}_{\text{short}} + \lambda \cdot \mathcal{R}_{\text{long}}, \tag{4}$$

where the hyperparameter $\lambda \in \mathbb{R}^+$ explicitly controls the balance between pedagogical principles and pedagogical effectiveness.

In contrast, SD adopts a stringent view of preference, generating a binary reward signal. The motivation for this approach is that preference-based algorithms like DPO rely on relative rankings rather than absolute reward values. Specifically, inspired by GRPO, which computes an advantage function within a candidate group (Shao et al., 2024), we compute the binary reward by comparing a candidate's reward vector $\mathbf{r}$ to the group's mean reward vector $\bar{\mathbf{r}}_{\text{group}}$. The resulting binary reward $\mathcal{R}$ is defined as:

$$\mathcal{R} = \begin{cases} 1, & \text{if } \mathbf{r} \succ \bar{\mathbf{r}}_{\text{group}} \\ 0, & \text{otherwise} \end{cases}, \tag{5}$$

where the $\succ$ denotes element-wise greater than.

3.3 Iterative Reinforcement Learning

With the multi-horizon reward function defined, the tutor agent's policy is refined via an iterative DPO framework. We adopt DPO for its superior training stability and lower resource consumption compared to online reinforcement learning approaches. To mitigate the critical issue of distribution shift inherent in offline reinforcement learning algorithms like DPO, our framework employs iterative reinforcement through a simulated interaction environment. This iterative process progressively enhances the tutor agent's adherence to pedagogical principles and its overall pedagogical effectiveness.

In each training iteration j, we construct a new preference dataset $D_{\text{DPO}}^{(j)}$ to refine the tutor agent's policy. For a current state $s_t = (\mathcal{B}, C_t)$, given candidate responses $\{a_t^{(i)}\}_{i=1}^N$ and their corresponding aggregated rewards $\{\mathcal{R}^{(i)}\}_{i=1}^N$, we compare all candidate pairs, designating the higher-reward response as chosen a_t^w and the lower as rejected a_t^l. The resulting dataset $D_{\text{DPO}}^{(j)}$ is composed of all such preference pairs (s_t, a_t^w, a_t^l) collected across states in the iteration j. We optimize the tutor agent's policy π_θ using this dataset via the DPO loss:

$$\mathcal{L}_{\text{DPO}} = -\mathbb{E}_{(s_t, a_t^w, a_t^l) \sim D_{\text{DPO}}^{(j)}} \left[\log \sigma \left(\beta \log \frac{\pi_\theta(a_t^w|s_t)}{\pi_{\text{ref}}(a_t^w|s_t)} - \beta \log \frac{\pi_\theta(a_t^l|s_t)}{\pi_{\text{ref}}(a_t^l|s_t)} \right) \right]. \tag{6}$$

4 Experiments

4.1 Experimental Setup

Datasets. We conduct our experiments on the MathDial dataset (Macina et al., 2023). This dataset comprises 2,861 one-on-one tutoring dialogues with 14,197 conversational turns in total. Each dialogue is grounded in a student's initial incorrect solution to a mathematical reasoning problem from the GSM8k dataset (Cobbe et al., 2021), with the pedagogical objective being for the tutor to guide the student to the correct answer. We adopt the original 80%/20% training and test split provided by the dataset. All experimental datasets are based on the original MathDial training set. Detailed statistics of the datasets are provided in Appendix A.

Implementation Details. For the student agent, we employ Qwen2.5-7B-Instruct (Team, 2024) as the base model. For the tutor agent, we primarily utilize two widely adopted large language models: Qwen2.5-7B-Instruct and Llama3.1-8B-Instruct (Dubey et al., 2024) as base models. For both SFT stages, we use a learning rate of 1e-5, a batch size of 32, and train for 3 epochs. In the iterative DPO training, for a given background and context, we sample 2 responses from the policy model. We then use GPT-4o to evaluate the pedagogical quality of these responses. Simultaneously, for each sampled response, we conduct 2 multi-turn Monte Carlo rollouts until the student successfully solves the problem or the interaction reaches 10 turns. For the iterative DPO training, we set β to 0.2 and train for 1 epoch. All training procedures are implemented using the Llama-Factory framework (Zheng et al., 2024) with LoRA fine-tuning (Hu et al., 2022). More implementation details can be seen in Appendix B.

Baselines. We evaluate our MHPO method, trained on two distinct backbone models, by comparing it against a diverse set of baselines employing different methodologies: (1) Vanilla, the base model with a standard, non-Socratic prompt; (2) Socratic Prompting (Liu et al., 2024a), the base model guided by a detailed Socratic prompt; (3) SFT, the base model fine-tuned using SFT on the dialogue dataset; (4) Tutor-DPO (Sonkar et al., 2024), an offline RL method using preference pairs; and (5) Tutor-RL (Dinucu-Jianu et al., 2025), an online RL approach that optimizes based on trajectory-level rewards.

Backbone	Method	Ped. Pri. ↑	Success rate ↑	Avg. Turns ↓	Avg. Tokens ↓
Qwen2.5-7B-Instruct	Vanilla	52.71	75	3.47	224.15
	Socratic	68.13	38	7.74	276.34
	SFT	66.22	45	7.49	147.80
	Tutor-RL	63.02	50	6.62	374.25
	MHPO-LS (Ours)	69.87	63	6.59	154.90
	MHPO-SD (Ours)	72.66	60	7.09	143.79
Llama3.1-8B-Instruct	Vanilla	48.71	73	4.00	192.66
	Socratic	67.79	32	8.18	495.36
	SFT	65.72	48	7.57	132.11
	Tutor-DPO	61.76	46	6.88	434.09
	MHPO-LS (Ours)	67.49	66	6.69	133.80
	MHPO-SD (Ours)	68.53	60	6.87	164.34

Table 1: Performance comparison of different methods across two backbones. MHPO significantly enhances different base model performance, achieving a superior balance between principles and effectiveness compared to various baselines.

Metrics. We evaluate the models on two primary dimensions: adherence to pedagogical principles *(Ped. Pri.)* and pedagogical effectiveness. Following prior works (Scarlatos et al., 2025; Dinucu-Jianu et al., 2025), we assess *Ped. Pri.* on clarity, Socratic guidance, and other key criteria by GPT-4o. Pedagogical effectiveness is measured by simulating interactions between each tutor agent and the simulated student agent, recording the *Success Rate* and Interaction Efficiency. The latter metric encompasses both the average number of turns *(Avg. Turns)* and the average total tokens *(Avg. Tokens)* generated by the tutor per dialogue. More detailed evaluation settings can be found in the Appendix C.

4.2 Main Results

Base models struggle to balance principles with effectiveness. As shown in Table 1, base models without fine-tuning fail to achieve a balance between adhering to pedagogical principles and achieving effective teaching outcomes. On one hand, the Vanilla model achieves a high success rate of 75% on Qwen2.5-7B and 73% on Llama3.1-8B, respectively, with a low number of rounds. This suggests the model often defaults to providing the correct answer directly—a strategy that is effective for task completion but pedagogically unsound, as reflected in its extremely low pedagogical principles (52.71 and 48.71). On the other hand, simply adding a Socratic prompt significantly improves the model's adherence to pedagogical principles, boosting the score to over 67 on both backbones. However, this severely compromises pedagogical effectiveness. The success rate plummets to 38% and 32%, respectively, while the interaction becomes excessively verbose and lengthy, as indicated by the sharp increase in turns and tokens. These findings highlight a critical challenge: base models tend to operate at two extremes. They are either effective but unprincipled answer-givers or principled but ineffective conversationalists, failing to achieve the crucial balance required for expert tutoring.

MHPO can enhance both principles and effectiveness concurrently. The results in Table 1 demonstrate that our proposed MHPO framework successfully overcomes the trade-off that constrains the baseline methods, concurrently enhancing adherence to pedagogical principles and pedagogical effectiveness. Across both the Qwen2.5-7B and Llama3.1-8B backbones, our MHPO variants consistently achieve the highest Pedagogical Principles, with MHPO-SD on the Qwen backbone reaching up to 72.66. Unlike the Socratic baseline, this significant improvement in pedagogical quality does not come at the cost of a lower success rate. In fact, our methods substantially boost the Success rate to as high as 63% (for MHPO-LS on Qwen) and 66% (for MHPO-LS on Llama), a considerable improvement over other tuned baselines

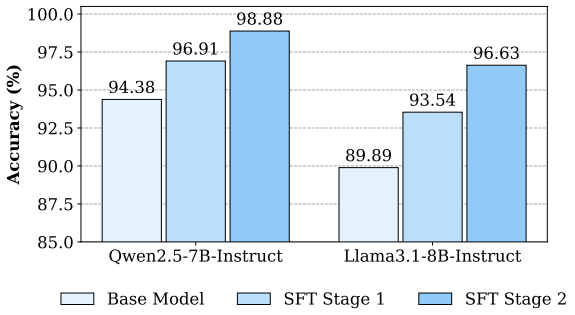

Figure 3: Effectiveness of the two-stage SFT process in improving the tutor agent's judgment accuracy.

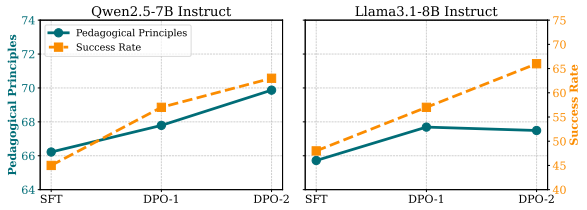

Figure 4: Progressive performance improvement across DPO iterations.

Strategy	Success Rate↑	Avg. Turns↓	Avg. Tokens↓
SFT	45	7.49	147.80
Avg. SR	54	7.51	149.44
Avg. Turns	63	6.59	154.90

Table 2: Impact of different trajectory-level reward strategies (Avg. SR: Average Success Rate).

like SFT and Tutor-DPO/RL, which linger in the 45-50% range. Crucially, these gains in principles and effectiveness are not achieved at the expense of interactional efficiency. On the contrary, our methods demonstrate highly competitive performance in both dialogue length and token usage. For instance, MHPO-SD is the most token-efficient method on the Qwen backbone (143.79). On the Llama backbone, MHPO-LS (133.80) achieves its high success rate while remaining nearly as concise as the strong SFT baseline and significantly leaner than other RL-based approaches. This demonstrates that by explicitly modeling and optimizing for the dual objectives of turn-level quality and trajectory-level effectiveness, our MHPO framework learns a superior and more balanced policy than other methods.

5 Analysis

Validity analysis of framework initialization. A foundational requirement for our reinforcement learning framework is the tutor's ability to reliably recognize a successful dialogue outcome, as this directly informs the trajectory-level reward signal. The efficacy of our initialization process is validated in Figure 3. The first stage, fine-tuning on D_{socratic}, equips the agent with general pedagogical skills and provides a substantial initial boost in judgment accuracy. Building upon this foundation, the second stage explicitly fine-tunes the agent on D_{judge}, further enhancing its capability to discern correct final answers. This sequential process proves highly effective, elevating the final judgment accuracy to 98.88% for Qwen2.5-7B and 96.63% for Llama3.1-8B. By achieving such high reliability, we ensure a clean and accurate reward signal for the subsequent RL optimization, minimizing noise and enabling more stable policy improvement.

Iterative refinement progressively boosts performance. To overcome the limitations of static offline training and mitigate the critical issue of distribution shift, our framework employs an iterative DPO optimization process. This approach allows the tutor agent to continually refine its policy by learning from preference data generated from its own ever-improving actions. The effectiveness of this strategy is clearly demonstrated in Figure 4. Over just two iterations, the agent achieves substantial gains, with the Success Rate for Qwen2.5-7B climbing from 45.0% to 63.0% and for Llama3.1-8B from 48.0% to 66.0%. Critically, these gains in effectiveness do not degrade pedagogical quality; in fact, the Pedagogical Principles also shows a steady increase. This confirms the efficacy of our framework's virtuous cycle: the improving agent policy continually generates higher-quality preference data, fueling its own progressive refinement.

Process-oriented reward outperforms outcome-oriented reward. We analyze the formulation of our trajectory-level reward $\mathcal{R}_{\text{long}}$ to identify the most effective learning signal for the tutor agent, with results presented in Table 2. A conventional approach is to directly optimize for the final outcome, measured by the average success rate (Avg. SR). This strategy yields a notable 9-point improvement in success rate over the SFT baseline. However, this gain comes at a slight cost to conversational efficiency, as both average turns and tokens increase. In stark contrast, when optimizing for an efficiency-focused strategy—minimizing the average number of turns (Avg. Turns)—we observe a far more compelling result. This approach not

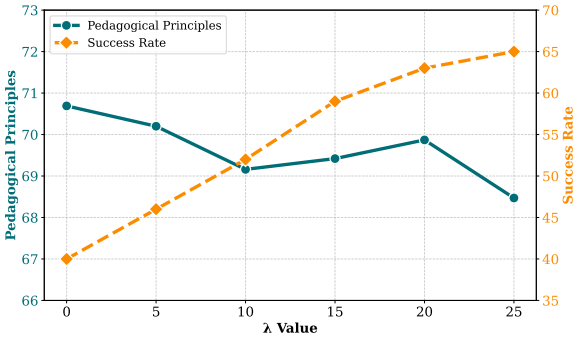

Figure 5: Impact of the trade-off hyperparameter λ on principles and effectiveness.

only successfully reduces the dialogue length from 7.49 to 6.59 turns but also produces a substantially larger 18-point gain in Success Rate. This finding provides a critical insight: a reward that incentivizes efficiency appears to be a more powerful and effective learning signal than one that solely targets the final outcome. This insight suggests a strong coupling between instructional conciseness and pedagogical success, implying that training an agent to be efficient is a shortcut to making it effective.

Balancing principles and effectiveness with λ. We analyze the impact of the hyperparameter λ, which balances the short-term principles reward ($\mathcal{R}_{\text{short}}$) and the long-term effectiveness reward ($\mathcal{R}_{\text{long}}$) in our MHPO-LS method. As illustrated in Figure 5, the results reveal a clear trade-off. Optimizing solely for principles (λ=0) yields a high Pedagogical Principles (70.69) but a very low Success Rate (40.0%), confirming that adherence to principles alone is insufficient for effective tutoring. As λ increases, the Success Rate demonstrates a strong and consistent upward trend, climbing from 40.0% to its peak of 65.0% at λ=25. Crucially, this significant gain in effectiveness is achieved with only a minor decrease in the Pedagogical Principles, which remains at a high level (above 68.0). This analysis demonstrates our framework's ability to effectively control the quality-effectiveness trade-off and identifies an optimal λ region that maximizes success while maintaining strong pedagogical principles.

Data quantity vs. quality in reward aggregation. We investigate the trade-off between data quantity and quality by comparing our Linear Scalarization (LS) and Strict Dominance (SD) strategies. Table 3 reveals the core difference: SD's stringent filtering yields a smaller, higher-quality dataset (3,778 pairs), while LS generates a larger, more varied set (7,106 pairs). Figure 6 illustrates how this trade-off impacts performance. For Success Rate (right panel), SD proves more data-efficient, leading at a 50% data scale (60% vs. 58%). However, LS scales better with more data, ultimately achieving a higher peak of 63% (MHPO-LS on Qwen). For Pedagogical Principles (left panel), SD's data quality advantage is clear, as it consistently achieves a superior score (peaking at 72.66). In essence, SD guarantees high pedagogical principles, especially from limited data, while LS unlocks higher pedagogical effectiveness when larger datasets are available.

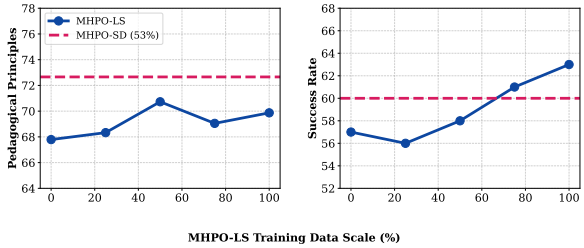

Figure 6: Performance of MHPO-LS and MHPO-SD across different data scales.

Strategy	Chosen		Rejected		Num
	Avg. T.	Avg. Pri.	Avg. T.	Avg. Pri.	
LS	3.74	60.79	7.84	58.20	7,106
SD	3.82	71.14	7.98	46.16	3,778

Table 3: Statistics of preference data generated by different reward aggregation strategies (Avg. T.: Average Turns; Avg. Pri.: Average Pedagogical Principles).

6 Conclusion

In this paper, we introduced MHPO, a novel framework for training LLM-based tutor agents that balances pedagogical principles with pedagogical effectiveness. Our multi-horizon reward function uses Monte Carlo rollouts to mitigate the credit assignment problem. Experiments show MHPO significantly outperforms baselines, with our analysis revealing that an efficiency-focused reward provides a superior learning signal for overall effectiveness. This framework provides a clear path toward AI tutors that not only deliver guidance aligned with core pedagogical principles but also demonstrate superior pedagogical effectiveness in achieving learning outcomes.

Limitations

While our findings demonstrate the promise of the MHPO framework, we acknowledge two primary limitations related to computational resource constraints.

Scale of model. Due to computational resource constraints, our study validates the proposed MHPO framework on models within the 7-8B parameter scale (Qwen2.5-7B and Llama3.1-8B). While our results show significant and consistent performance gains at this scale, the framework's effectiveness on substantially larger models remains an open question. Future work should focus on applying and evaluating MHPO on large-scale models to confirm its scalability and impact.

Scale of Monte Carlo rollouts. Another limitation related to computational cost is the scale of the sampling process for our Monte Carlo rollouts. To maintain tractability, our experiments were conducted with a limited number of candidate responses (N=2) and simulation rollouts (M=2) for estimating the long-term reward. While this configuration was sufficient to demonstrate the framework's effectiveness over baselines, a more extensive search with a larger N and a more stable value estimation with a larger M could potentially unlock further performance gains. Future research should investigate the impact of scaling these parameters on the final policy.

Ethics Statement

Our research on AI tutors, which utilizes the public MathDial dataset in compliance with its CC BY-SA 4.0 license, is motivated by the goal of enhancing educational accessibility. However, we acknowledge associated ethical risks. These include the potential for student over-reliance on AI and the possibility of flawed guidance. Furthermore, our framework is built upon Large Language Models (LLMs), which are known to reflect societal biases and can generate incorrect information. While our pedagogical fine-tuning mitigates these issues, it does not eradicate the inherent risk of the models producing biased or unsafe content.

References

Vincent Aleven, Elmar Stahl, Silke Schworm, Frank Fischer, and Raven Wallace. 2003. Help seeking and help design in interactive learning environments. *Review of educational research*, 73(3):277–320.

Julia Anghileri. 2006. Scaffolding practices that enhance mathematics learning. *Journal of Mathematics Teacher Education*, 9(1):33–52.

Cheng Cheng, Guanhao Zhao, Zhenya Huang, Yan Zhuang, Zhaoyuan Pan, Qi Liu, Xin Li, and Enhong Chen. 2024. Towards explainable computerized adaptive testing with large language model. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 2655–2672.

Zhendong Chu, Shen Wang, Jian Xie, Tinghui Zhu, Yibo Yan, Jinheng Ye, Aoxiao Zhong, Xuming Hu, Jing Liang, Philip S Yu, and 1 others. 2025. Llm agents for education: Advances and applications. *arXiv preprint arXiv:2503.11733*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Peter A Cohen, James A Kulik, and Chen-Lin C Kulik. 1982. Educational outcomes of tutoring: A meta-analysis of findings. *American educational research journal*, 19(2):237–248.

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.

David Dinucu-Jianu, Jakub Macina, Nico Daheim, Ido Hakimi, Iryna Gurevych, and Mrinmaya Sachan. 2025. From problem-solving to teaching problem-solving: Aligning llms with pedagogy using reinforcement learning. *arXiv preprint arXiv:2505.15607*.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, and 1 others. 2025. Agentic reinforced policy optimization. *arXiv preprint arXiv:2507.19849*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Arthur C Graesser, Patrick Chipman, Brian C Haynes, and Andrew Olney. 2005. Autotutor: An intelligent tutoring system with mixed-initiative dialogue. *IEEE Transactions on Education*, 48(4):612–618.

Arthur C Graesser, Kurt VanLehn, Carolyn P Rosé, Pamela W Jordan, and Derek Harter. 2001. Intelligent tutoring systems with conversational dialogue. *AI magazine*, 22(4):39–39.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. Orpo: Monolithic preference optimization without reference model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11170–11189.

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.

Priyanka Kargupta, Ishika Agarwal, Dilek Hakkani-Tur, and Jiawei Han. 2024. Instruct, not assist: Llm-based multi-turn planning and hierarchical questioning for socratic code debugging. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 9475–9495.

Nischal Ashok Kumar and Andrew Lan. 2024. Improving socratic question generation using data augmentation and preference optimization. In *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*, pages 108–118.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626.

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, Geyu Lin, Hui Li Tan, Huayun Zhang, Yanfeng Lu, Xiaoxue Gao, Stella Xin Yin, He Sun, Hock Huan Goh, Lung Hsiang Wong, and 1 others. 2025. Singakids: A multilingual multimodal dialogic tutor for language learning. *arXiv preprint arXiv:2506.02412*.

Zhengyuan Liu, Stella 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.

Yu Meng, Mengzhou Xia, and Danqi Chen. 2024. Simpo: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Richard Paul and Linda Elder. 2007. Critical thinking: The art of socratic questioning. *Journal of developmental education*, 31(1):36.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741.

Steven Ritter, John R Anderson, Kenneth R Koedinger, and Albert Corbett. 2007. Cognitive tutor: Applied research in mathematics education. *Psychonomic bulletin & review*, 14(2):249–255.

Alexander Scarlatos, Naiming Liu, Jaewook Lee, Richard Baraniuk, and Andrew Lan. 2025. Training llm-based tutors to improve student learning outcomes in dialogues. In *International Conference on Artificial Intelligence in Education*, pages 251–266. Springer.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Kumar Shridhar, Jakub Macina, Mennatallah El-Assady, Tanmay Sinha, Manu Kapur, and Mrinmaya Sachan. 2022. Automatic generation of socratic subquestions for teaching math word problems. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4136–4149.

Shashank Sonkar, Naiming Liu, Debshila Mallick, and Richard Baraniuk. 2023. Class: A design framework for building intelligent tutoring systems based on learning science principles. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1941–1961.

Shashank Sonkar, Kangqi Ni, Sapana Chaudhary, and Richard Baraniuk. 2024. Pedagogical alignment of large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 13641–13650.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Rose Wang, Qingyang Zhang, Carly Robinson, Susanna Loeb, and Dorottya Demszky. 2024. Bridging the novice-expert gap via models of decision-making: A case study on remediating math mistakes. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2174–2199.

Sebastian Wollny, Jan Schneider, Daniele Di Mitri, Joshua Weidlich, Marc Rittberger, and Hendrik Drachsler. 2021. Are we there yet?-a systematic literature review on chatbots in education. *Frontiers in artificial intelligence*, 4:654924.

David Wood, Jerome S Bruner, and Gail Ross. 1976. The role of tutoring in problem solving. *Journal of child psychology and psychiatry*, 17(2):89–100.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, and 1 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*.

Yuanqing Yu, Zhefan Wang, Weizhi Ma, Shuai Wang, Chuhan Wu, Zhiqiang Guo, and Min Zhang. 2024. Steptool: Enhancing multi-step tool usage in llms through step-grained reinforcement learning. *arXiv preprint arXiv:2410.07745*.

Xueqiao Zhang, Chao Zhang, Jianwen Sun, Jun Xiao, Yi Yang, and Yawei Luo. 2025. Eduplanner: Llm-based multi-agent systems for customized and intelligent instructional design. *IEEE Transactions on Learning Technologies*.

Yaowei Zheng, Richong Zhang, Junhao Zhang, YeYanhan YeYanhan, and Zheyan Luo. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410.

A Datasets Details

The MathDial dataset comprises 2,861 one-on-one tutoring dialogues with 14,197 conversational turns in total. Each dialogue is grounded in a student's initial incorrect solution to a mathematical reasoning problem from the GSM8k dataset, with the pedagogical objective being for the tutor to guide the student to the correct answer. We adopt the original 80%/20% training and test split provided by the dataset.

The tutor agent's initial SFT stages utilize specifically constructed datasets D_{socratic} derived from the MathDial training data for Socratic guidance. It contains 14,905 entries, where each entry corresponds to a tutor's turn in a dialogue. Each entry comprises the following content: Question, Ground Truth Answer, Student Answer, Student Profile, the preceding Conversation history, and the target Ground Truth tutor response. Additionally, a separate dataset D_{judge} is specifically constructed to enhance the tutor's judgment capability. This dataset is derived from student turns within the training set, categorized by whether dialogues were successfully completed or remained unfinished. This dataset comprises 3,388 entries. A subset of 356 entries from this dataset is specifically reserved as a test set to validate the judgment accuracy of the tutor agent. Each entry in D_{judge} includes the following fields: Question, Ground Truth Answer, and Student Response. The student agent is trained on the dataset $D_{simulation}$, which comprises 13,406 samples of student profiles and their dialogue turns from the MathDial training set.

B Implementation Details

For both SFT stages, we use a learning rate of 1e-5, a batch size of 32, and train for 3 epochs. In the iterative DPO training, for a given background and context, we sample 2 responses from the policy model. We then use GPT-4o to evaluate the pedagogical quality of these responses. Simultaneously, for each sampled response, we conduct 2 multi-turn Monte Carlo rollouts until the student successfully solves the problem or the interaction reaches 10 turns. For DPO training, we use a batch size of 32, set β to 0.2, and train for 1 epoch. The learning rate for the Qwen2.5-7B-Instruct backbone is 8e-7 for both iterations. For the Llama3.1-8B-Instruct backbone, we apply a learning rate of 8e-7 for the first iteration, which is then reduced to 8e-8 for the second. All training procedures are implemented using the Llama-Factory framework (Zheng et al., 2024) with LoRA fine-tuning (Hu et al., 2022), where the alpha and rank are set to 16, and the dropout rate is 0.05. For inference, we set the model temperature to 0.7 and top-p to 0.95. We utilize vLLM (Kwon et al., 2023) to accelerate the inference process. All our experiments are conducted on 4×24GB RTX 4090 GPUs.

λ Value	Chosen		Rejected		Num
	Avg. T.	Avg. Pri.	Avg. T.	Avg. Pri.	
0	5.81	74.96	5.71	46.06	6,763
5	4.77	70.39	6.79	48.12	6,991
10	4.04	65.46	7.55	53.36	6,915
15	3.87	62.54	7.71	56.36	7,183
20	3.74	60.79	7.84	58.20	7,106
25	3.74	59.72	7.85	59.08	7,229

Table 4: Statistics of preference data generated by different λ Values (Avg. T.: Average Turns; Avg. Pri.: Average Pedagogical Principles).

C Evaluation Details

For pedagogical effectiveness evaluation, we use the Qwen2.5-7B-Instruct backbone student model to simulate the environment and employ GPT-4o-mini to assess whether the dialogue is successful. The evaluation involves randomly sampling 100 scenarios from the test set, and the model to be evaluated interacts based on the simulated environment of the student model. The dialogue continues until the student correctly solves the problem or reaches the maximum of 10 turns. The average total tokens (Avg. Tokens) across all tutor turns in a dialogue is calculated using the Qwen2.5-7B-Instruct tokenizer. For the evaluation of pedagogical principles, we use the candidate model to generate corresponding actions based on the states in the test set, and employ GPT-4o for assessment according to predefined detailed prompts. The prompt used for this evaluation is shown in Figure 11.

D Supplementary Experimental Results

Table 4 illustrates the impact of the hyperparameter λ on the characteristics of the generated preference data used for DPO training. As the value of λ increases, placing more weight on the long-term efficiency reward, the model increasingly prefers responses that lead to shorter dialogues. This is evident in the Chosen pairs, where the Average Turns (Avg. T.) steadily decrease from 5.81 to 3.74. This gain in efficiency comes at the expected cost

of a relaxed adherence to turn-level pedagogical principles, as shown by the decreasing Average Principles (Avg. Pri.) score for the Chosen responses.

E Prompts

This appendix details the prompts used in our framework. Figure 7 and Figure 8 present the prompts of the tutor agent and the student agent, respectively. The prompt utilized to evaluate the correctness of the student's answer is presented in Figure 9. Figure 10 illustrates the prompt for calculating the pedagogical principles reward. Figure 11 shows the prompt employed to evaluate how well responses adhere to pedagogical principles.

F Case Study

In this section, we present a comparative case study to provide a concrete illustration of the behavioral differences between our MHPO-trained tutor agent (Success Example) and a baseline agent (Failure Example) when addressing the same scenario.

As shown in Figure 12, the MHPO agent demonstrates a remarkable ability to balance pedagogical principles with interactional efficiency. After the student explains their flawed reasoning, the tutor, in Turn 3, immediately pinpoints the student's single core misconception—confusing "cups per day" with "cups per meal." It then poses a targeted Socratic question to correct this specific error. This concise and direct guidance, focused on the root of the problem, allows the agent to steer the student back to the correct reasoning path and achieve a successful resolution in a short number of turns. This behavior emulates the targeted efficiency of an expert human educator.

In stark contrast, Figure 13 highlights a baseline agent that prioritizes turn-level pedagogical principles at the expense of pedagogical effectiveness. The baseline tutor begins with overly broad questions like "What do you think about the calculation...?", which fail to provide any meaningful guidance to the confused student. Subsequently, the agent gets stuck in a repetitive and unproductive loop, repeatedly asking "Why...?" without offering a new perspective. Ultimately, it abandons the original problem's context entirely, derailing the conversation into a confusing sub-problem. This long and inefficient dialogue fails to correct the student's error, perfectly illustrating how a narrow focus on "asking questions" without strategic foresight compromises the ultimate goal of learning.

This side-by-side comparison validates that our MHPO framework learns a more sophisticated and effective tutoring policy, one that is capable of diagnosing and addressing student errors with the precision and efficiency of an expert.

Prompt for Tutor Agent

System
You are a teacher that always responds in the Socratic style. You will be given a math question, the ground truth answer, the student's incorrect answer and the student's profile along with the context of the conversation, generating response to guide the student to the correct answer. Your response should be concise, clear, and focused on helping the student understand the problem better.

Input
Question: {question}
Ground Truth Answer: {gt_answer}
Student Answer: {stu_answer}
Student Profile: {stu_profile}
Conversation: {conversation}

Figure 7: Prompt for tutor agent.

Prompt for Student Agent

System
You are a student who has made an incorrect attempt at solving a math question. You will be given the math question, the ground truth answer, your incorrect answer, and your profile along with the context of the conversation. Your task is to reflect on your previous attempt and try to arrive at the correct answer by engaging in a conversation with the teacher.

Input
Question: {question}
Ground Truth Answer: {gt_answer}
Student Answer: {stu_answer}
Student Profile: {stu_profile}
Conversation: {conversation}

Figure 8: Prompt for student agent.

Prompt for Response Judge

System
You are a judge that determines whether the student's response is correct or not. You will be given the question, the ground truth answer and student's response. Your task is to determine if the student's response is correct or not. If it is correct, return True, otherwise return False.

Input
Question: {question}
Ground Truth Answer: {gt_answer}
Student Response: {student_response}

Figure 9: Prompt for response judge.

Prompt for Reward Modeling

System
You will be given a math question, the Ground Truth answer, the student's incorrect answer, the student's profile, the conversation context, the Ground Truth teacher response, and two different teacher responses (Teacher Response A and Teacher Response B).

Attention
Your task is to independently rate each teacher response with the Ground Truth teacher response in "Efficient Socratic Guidance" on a 0-100 scale according to the criteria below.
Assess each response solely based on its own quality and its effectiveness in efficient Socratic guidance within the given context.
The scoring must be discriminative: assign a higher score for a concise response with rhetorical questions.
Avoid introducing external information or personal biases to ensure the objectivity and accuracy of the analysis. For example, longer responses do not necessarily indicate better quality.
Only output the two scores (two numbers, separated by a comma and in the order: Teacher Response A, Teacher Response B). Do not provide any explanation or additional text.

Evaluation Criteria
Scoring Criteria (0-100):
0-19: The response is excessively long, unclear, or off-topic. It does not ask any guiding questions or encourage student thinking. Mainly provides information in a didactic or mechanical way.
20-39: The response is somewhat wordy or unfocused, with limited clarity. Rarely uses questions or prompts, and does little to engage the student's thought process.
40-59: The response is mostly understandable and sometimes uses simple questions or prompts. However, it could be more concise or more effectively targeted to inspire the student.
60-79: The response is clear and concise, with frequent and well-placed guiding or probing questions. It effectively encourages the student to think or explain their reasoning, while remaining focused and efficient.
80-100: The response is exceptionally concise, clear, and highly focused. It skillfully uses targeted questions or prompts to inspire deep student thinking, self-reflection, or correction, with no unnecessary explanation or off-topic content.

Input
Question: {question}
Ground Truth Answer: {gt_answer}
Student Answer: {stu_answer}
Student Profile: {stu_profile}
Conversation: {conversation}
Ground Truth Teacher Response: {gt_teacher_response}
Teacher Response A: {teacher_response_A}
Teacher Response B: {teacher_response_B}

Figure 10: Prompt for rewarding responses.

Prompt for Evaluating Pedagogical Principles

System
You will be given a math question, the Ground Truth answer, the student's incorrect answer, the student's profile, the conversation context, the teacher's response, and the Ground Truth teacher's response.

Attention
Your task is to compare the teacher's response with the Ground Truth teacher's response and rate the teacher's performance in "Efficient Socratic Guidance" on a 0-100 scale according to the criteria below.
Scoring must have discrimination, giving high marks for answers close to Ground Truth. Try to differentiate between different levels as much as possible. Higher scores for concise responses with rhetorical questions.
Avoid introducing external information or personal biases to ensure the objectivity and accuracy of the analysis. For example, longer responses do not necessarily indicate better quality.
Only output the score (a single number). Do not provide any explanation.

Evaluation Criteria
Scoring Criteria (0-100):
0-19: The response is excessively long, unclear, or off-topic. It does not ask any guiding questions or encourage student thinking. Mainly provides information in a didactic or mechanical way.
20-39: The response is somewhat wordy or unfocused, with limited clarity. Rarely uses questions or prompts, and does little to engage the student's thought process.
40-59: The response is mostly understandable and sometimes uses simple questions or prompts. However, it could be more concise or more effectively targeted to inspire the student.
60-79: The response is clear and concise, with frequent and well-placed guiding or probing questions. It effectively encourages the student to think or explain their reasoning, while remaining focused and efficient.
80-100: The response is exceptionally concise, clear, and highly focused. It skillfully uses targeted questions or prompts to inspire deep student thinking, self-reflection, or correction, with no unnecessary explanation or off-topic content.

Input
Question: {question}
Ground Truth Answer: {gt_answer}
Student Answer: {stu_answer}
Student Profile: {stu_profile}
Conversation: {conversation}
Teacher Response: {teacher_response}
Ground Truth Teacher Response: {GT_response}

Figure 11: Prompt for evaluating pedagogical principles.

Success Example

Question
Every day, Wendi feeds each of her chickens three cups of mixed chicken feed, containing seeds, mealworms and vegetables to help keep them healthy. She gives the chickens their feed in three separate meals. In the morning, she gives her flock of chickens 15 cups of feed. In the afternoon, she gives her chickens another 25 cups of feed. How many cups of feed does she need to give her chickens in the final meal of the day if the size of Wendi's flock is 20 chickens?

Ground_Truth_Solution
1. If each chicken eats 3 cups of feed per day, then for 20 chickens they would need 3*20=60 cups of feed per day. **2.**If she feeds the flock 15 cups of feed in the morning, and 25 cups in the afternoon, then the final meal would require 60-15-25=20 cups of chicken feed. Answer: 20

Student_Incorrect_Solution
1.Wendi gives each chicken 3 cups of feed per meal, so for 20 chickens, she needs to give 3 x 20 = 60 cups of feed per meal.In total, for the day, she needs to give 60 x 3 = 180 cups of feed. **2.**She has already given 15 + 25 = 40 cups of feed in the morning and afternoon. **3.**Therefore, in the final meal, she needs to give 180 - 40 = 140 cups of feed. Answer: 140

Student_Profile
Mariana is a 7th grade student. She struggle most with understanding what the problem is asking them to do.

Conversation
Teacher: Hi Mariana, can you please walk me through your solution?
Student: Sure. I started by figuring out how much feed each chicken gets per meal. Since each chicken gets 3 cups of feed, for 20 chickens, she needs to give 3 x 20 = 60 cups of feed per meal. Then I figured out how much feed she needs for the day by multiplying 60 x 3 = 180 cups of feed. Finally, I subtracted the 40 cups of feed she has already given in the morning and afternoon, so she needs to give 180 - 40 = 140 cups of feed in the final meal.
Teacher: You have correctly figured out that each chicken needs 3 cups of feed per day, but you have multiplied it by 3 as you have said for each meal. However, the question tells us that she feeds the chickens 3 times a day, so how many cups of feed does each chicken get per day?
Student: Each chicken gets 3 cups of feed per meal, so for 20 chickens, she needs to give 3 x 20 = 60 cups of feed per day.
Teacher: That's right, so how many cups of feed would she need to give the chickens in the final meal if she has already given them 15 cups in the morning and 25 cups in the afternoon?
Student: In the final meal, she needs to give 180 - 40 = 140 cups of feed.
Teacher: Remember that she needs to give each chicken 3 cups of feed per day, so how many cups of feed would she need to give the chickens in the final meal?
Student: In the final meal, she needs to give 60 - 15 - 25 = 20 cups of feed.

Figure 12: A case study with our MHPO-trained tutor agent.

Failure Example

Question
Every day, Wendi feeds each of her chickens three cups of mixed chicken feed, containing seeds, mealworms and vegetables to help keep them healthy. She gives the chickens their feed in three separate meals. In the morning, she gives her flock of chickens 15 cups of feed. In the afternoon, she gives her chickens another 25 cups of feed. How many cups of feed does she need to give her chickens in the final meal of the day if the size of Wendi's flock is 20 chickens?

Ground_Truth_Solution
1. If each chicken eats 3 cups of feed per day, then for 20 chickens they would need 3*20=60 cups of feed per day. **2.**If she feeds the flock 15 cups of feed in the morning, and 25 cups in the afternoon, then the final meal would require 60-15-25=20 cups of chicken feed. Answer: 20

Student_Incorrect_Solution
1. Wendi gives each chicken 3 cups of feed per meal, so for 20 chickens, she needs to give 3 x 20 = 60 cups of feed per meal.In total, for the day, she needs to give 60 x 3 = 180 cups of feed. **2.**She has already given 15 + 25 = 40 cups of feed in the morning and afternoon. **3.**Therefore, in the final meal, she needs to give 180 - 40 = 140 cups of feed. Answer: 140

Student_Profile
Mariana is a 7th grade student. She struggle most with understanding what the problem is asking them to do.

Conversation
Teacher: Hello Mariana, can you please walk me through your solution?
Student: Sure. I started by calculating the total amount of feed needed for the day, which was 60 cups per meal multiplied by 3 meals, giving me a total of 180 cups of feed. Then I subtracted the 40 cups of feed that Wendi had already given in the morning and afternoon, leaving me with 140 cups of feed for the final meal.
Teacher: What do you think about the calculation for the total amount of feed for the day?
Student: I think my calculation for the total amount of feed for the day was correct. I multiplied the amount of feed per meal (3 cups) by the number of meals (3) to get the total amount of feed needed for the day (60 cups).
Teacher: Why did you multiply the number of meals by the amount of feed per meal?
Student: I multiplied the number of meals by the amount of feed per meal because I wanted to calculate the total amount of feed needed for the day. Since Wendi gives each chicken 3 cups of feed per meal and she has 20 chickens, I multiplied 3 by 20 to get the total amount of feed needed for the day (60 cups).
Teacher: Why did you multiply 3 by 20?
Student: I multiplied 3 by 20 because Wendi gives each chicken 3 cups of feed per meal and she has 20 chickens, so I wanted to calculate the total amount of feed needed for the day. Multiplying 3 by 20 gives me the total amount of feed needed for the day (60 cups).
Teacher: You see, she feeds the chickens in the morning, afternoon and evening. So if she feeds them 3 cups in the morning and 3 cups in the afternoon, how many cups does she need to feed them in the evening?
Student: Wendi needs to feed the chickens 6 cups in the evening. Since she feeds each chicken 3 cups of feed per meal and she has 20 chickens, she needs to feed them 3 x 20 = 60 cups of feed in the evening.
Teacher: Remember, she feeds them 3 cups in the morning and 3 cups in the afternoon.
Student: Yes, that's right. Wendi needs to feed the chickens 6 cups in the evening. Since she feeds each chicken 3 cups of feed per meal and she has 20 chickens, she needs to feed them 3 x 20 = 60 cups of feed in the evening.
Teacher: No, she needs to feed them 6 cups in the evening. That's 2 cups for each chicken.
Student: Oh, I see. Wendi needs to feed the chickens 2 cups in the evening. Since she feeds each chicken 3 cups of feed per meal and she has 20 chickens, she needs to feed them 2 x 20 = 40 cups of feed in the evening.
Teacher: How many cups does she already give them in the morning and afternoon?
Student: Wendi has already given the chickens 15 cups in the morning and 25 cups in the afternoon, for a total of 40 cups.
Teacher: If she gives them 2 cups each in the evening, how many cups does that add up to for the whole day?
Student: Wendi needs to give the chickens 2 cups in the evening, so that adds up to 2 x 20 = 40 cups. Adding that to the 40 cups she has already given in the morning and afternoon, that gives a total of 80 cups of feed for the day.
Teacher: No, it adds up to 60 cups for the whole day.
Student: Oh, I see. Wendi needs to give the chickens 2 cups in the evening, so that adds up to 2 x 20 = 40 cups. Adding that to the 40 cups she has already given in the morning and afternoon, that gives a total of 60 cups of feed for the day.

Figure 13: A case study with a baseline tutor agent.