Rewarding How Models Think Pedagogically: Integrating Pedagogical Reasoning and Thinking Rewards for LLMs in Education

Anonymous ACL submission

Abstract

Large language models (LLMs) are increasingly deployed as intelligent tutoring systems in education, yet existing reinforcement learning (RL) approaches for training LLM tutors optimize only the final response and leave the model's internal thinking process unsupervised. We introduce PedagogicalRL-Thinking, a framework that extends pedagogical alignment to the thinking phase of reasoning LLMs through two components: (1) Thinking Reward, which rewards the pedagogical quality of reasoning traces, and (2) Pedagogical Reasoning Prompting, which guides internal reasoning with domain-specific educational theory rather than generic instructions. The combination of both components outperforms all ablated variants, with each contributing complementary gains over generic prompting and response-only RL, and our 8B model approaches the performance of much larger frontier zero-shot models on student-improvement outcomes. Moreover, models trained only on mathematics tutoring also improve on unseen educational benchmarks while preserving the base model's factual knowledge. Quantitative and qualitative analyses confirm that rewarding pedagogical thinking systematically reshapes reasoning traces toward more structured, instructionally beneficial appropriate decision-making. These findings point toward LLM tutors that scaffold student learning rather than simply provide answers.

1 Introduction

Large language models (LLMs) are increasingly deployed as intelligent tutoring systems, offering scalable one-on-one educational support across diverse subject domains (Kasneci et al., 2023; Roll and Wylie, 2016). However, a fundamental challenge persists: LLM tutors often provide direct answers rather than guiding students through the problem-solving process, undermining the pedagogical value of the interaction between human and artificial intelligence (AI) (MacNeil et al., 2023; Tack et al., 2023). This tendency reflects a misalignment between the model's optimization objective, typically next-token prediction (i.e., answer), and the educational goal of fostering student understanding through scaffolded guidance (i.e., process).

Recent work has begun addressing this challenge through reinforcement learning (RL) approaches. Notably, PedagogicalRL (Dinucu-Jianu et al., 2025a) formulates tutoring as an RL problem with rewards for solution correctness and pedagogical quality (Dinucu-Jianu et al., 2025b; Lee et al., 2025). However, such approaches still focus exclusively on the *visible response*, the text presented to students, while ignoring the model's internal *thinking process*. This limitation is particularly notable given the emergence of reasoning-specialized LLMs (DeepSeek-AI, 2025) that explicitly generate intermediate reasoning traces before producing final responses.

This paper addresses a central research question: Can applying pedagogical rewards to the thinking process, in addition to visible responses, improve LLM tutor performance? We hypothesize that rewarding pedagogically appropriate reasoning, not just pedagogically appropriate outputs, will produce tutors that more consistently guide students without revealing answers.

We introduce PedagogicalRL-Thinking[1], a framework that extends pedagogical reinforcement learning to the thinking phase of reasoning LLMs (Figure 1). Our approach integrates two complementary components: (1) Pedagogical Reasoning Prompting, which replaces generic tutoring instructions with domain-specific prompts grounded in mathematics education theory (Polya's four-step problem-solving framework (Polya, 1945)); and (2)

[1] https://anonymous.4open.science/r/pedagogical_reasoning_submission-2787/README.md

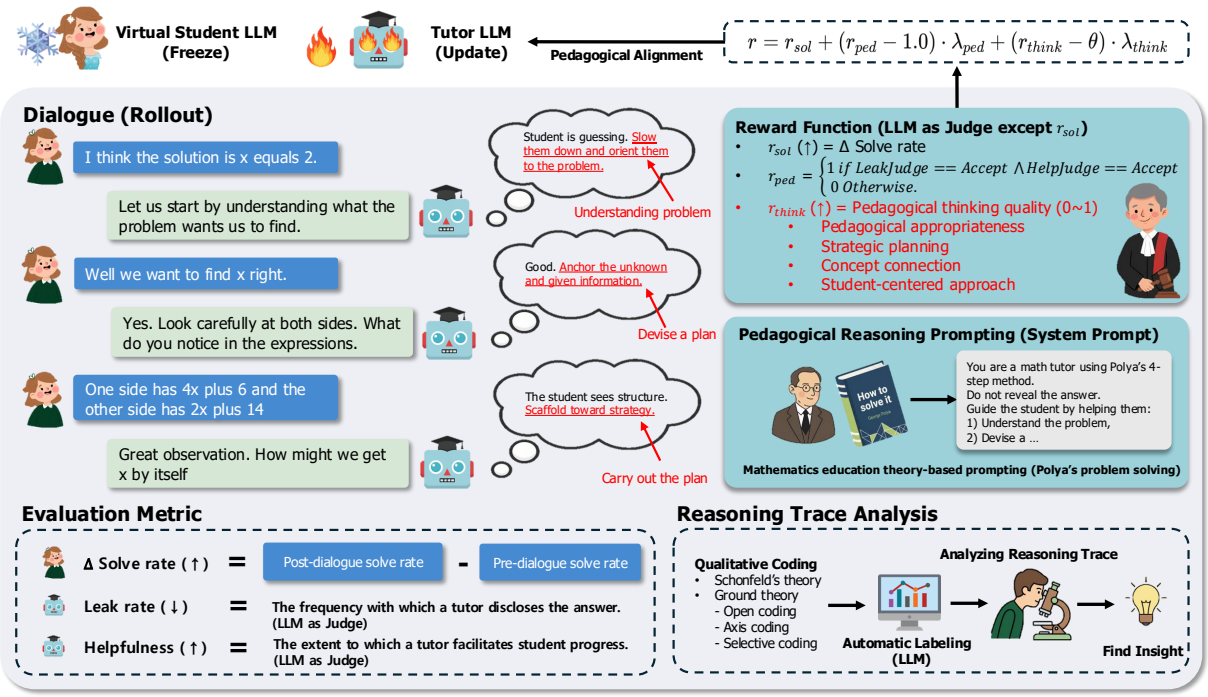

Figure 1: Overview of PedagogicalRL-Thinking. A frozen virtual student LLM interacts with an updatable tutor LLM through multi-turn dialogues. Our framework integrates two components. (1) Pedagogical Reasoning Prompting guides the tutor's internal reasoning using Polya's four-step problem-solving method. (2) Thinking Reward explicitly evaluates the pedagogical quality of reasoning traces (r_{think}). The composite reward function combines solve rate improvement (r_{sol}), pedagogical appropriateness (r_{ped}), and thinking quality (r_{think}). We evaluate tutors on Δ solve rate, leak rate, and helpfulness, and analyze reasoning traces through qualitative coding.

Thinking Reward, an LLM-based evaluation of the pedagogical quality of reasoning traces, providing explicit training signal for the thinking process.

1.1 Contributions

Our main contributions are:

- We propose Thinking Reward, the first pedagogical reward applied to the internal thinking process of reasoning LLMs, rewarding how a model reasons rather than only what it outputs.
- We introduce Pedagogical Reasoning Prompting grounded in Polya's framework, which outperforms generic tutoring prompts on both thinking quality and visible response quality.
- We show that dialogue-based RL on mathematics tutoring transfers to unseen educational benchmarks while preserving factual knowledge.
- We provide an 82-code educational codebook that exposes systematic behavior differences across conditions with statistical validation.

Together, these contributions enable an 8B model trained with our method to approach the performance of much larger frontier zero-shot models on student-improvement outcomes.

2 Method

2.1 Problem Formulation

We formulate LLM tutoring as a Markov Decision Process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R)$ over multi-turn tutor-student dialogues. At each turn t, the state $s_t \in \mathcal{S}$ comprises the dialogue history including the math problem, all previous student messages, and tutor responses. The action $a_t \in \mathcal{A}$ is the tutor's response, consisting of a thinking component a_t^{think} enclosed in `<think>` tags and a visible component a_t^{visible} shown to the student. The reward $r_t = R(s_t, a_t)$ is a composite signal evaluating solution correctness, pedagogical quality, and thinking quality (defined in Section 2.2). The goal is to learn a tutor policy π_θ that maximizes expected cumulative reward:

$$\pi^* = \arg\max_{\pi_\theta} E_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T} R(s_t, a_t) \right], \quad (1)$$

where τ denotes a complete tutoring dialogue rolled out under π_θ.

2.2 Reward Design

Our reward function consists of three components, each addressing a distinct aspect of tutoring quality.

For solution correctness (r_{sol}), following (Dinucu-Jianu et al., 2025a), we evaluate whether the tutoring dialogue improves the student's problem-solving ability. After the dialogue concludes, the student model attempts the problem $K = 8$ times, and we compute the average success rate as r_{sol}.

For pedagogical quality (r_{ped}), we employ two LLM-based judges (GPT-4o-mini) to assess pedagogical appropriateness. The Leak Judge determines whether the tutor revealed the answer or solution steps that the student should discover independently. The Help Judge assesses whether the tutor's response provides meaningful educational guidance. We set $r_{\text{ped}} = 1.0$ if both judges accept the response, and $r_{\text{ped}} = 0.0$ otherwise.

For thinking quality (r_{think}), our key contribution is explicitly rewarding the thinking process. An LLM judge (GPT-4o-mini) evaluates the content within `<think>` tags for pedagogical appropriateness of reasoning, consideration of student understanding, and metacognitive awareness in planning responses. The score $r_{\text{think}} \in [0, 1]$ reflects the overall pedagogical quality of the thinking trace.

The composite reward combines all components:

$$r = r_{\text{sol}} + (r_{\text{ped}} - 1.0) \cdot \lambda_{\text{ped}} + (r_{\text{think}} - \theta) \cdot \lambda_{\text{think}} \quad (2)$$

where $\lambda_{\text{ped}} = 0.75$ penalizes pedagogically inappropriate responses, $\lambda_{\text{think}} = 0.3$ weights the thinking reward, and $\theta = 0.6$ sets the thinking quality threshold.

2.3 Pedagogical Reasoning Prompting

Pedagogical Reasoning Prompting is grounded in Polya's four-step problem-solving method (Polya, 1945), instructing the tutor to guide students through: (1) understanding the problem, (2) devising a plan, (3) carrying out the plan, and (4) looking back. For example: "*Guide the student by helping them understand the problem, devise a plan, carry out the plan step by step, and verify whether the answer makes sense.*" The full prompts are provided in Appendix A.

2.4 Training Algorithm

We use Group Relative Policy Optimization (GRPO) (Shao et al., 2024) implemented in the veRL framework. GRPO computes advantages by comparing outcomes within groups of sampled responses, avoiding the need for a separate value model. For each training step, we sample a batch of 16 problems, generating 8 complete dialogue rollouts per problem for a total of 128 rollouts per batch. Each dialogue is limited to a maximum of 16 turns. Training uses 4 GPUs with hyperparameters detailed in Appendix C.

3 Experiments

3.1 Experimental Setup

We use BigMath (Albalak et al., 2025) for training and evaluation. Following prior work (Dinucu-Jianu et al., 2025a,b), Llama-3.1-8B-Instruct serves as the student simulator. We filter problems by student solve rate (1-60%), yielding 10,000 training and 500 evaluation problems.

For thinking-enabled conditions we use DeepSeek-R1-0528-Qwen3-8B (an 8B reasoning-specialized model that emits explicit `<think>` traces; hereafter Qwen3-8B), and Qwen2.5-7B-Instruct for the NoThink baseline. Full configuration is in Appendix C.

NoThink reproduces the PedagogicalRL paradigm (Dinucu-Jianu et al., 2025a) on Qwen2.5-7B (response-only RL without thinking) and serves as a prior-work baseline. The remaining four conditions all use Qwen3-8B with thinking enabled and form a 2×2 ablation: *Think NoReward* / *Think Reward* use standard prompting with and without the thinking reward, while *Ped. Think NoReward* / *Ped. Think Reward* use Pedagogical Reasoning Prompting with and without it.

We report three tutoring-quality metrics on BigMath: Delta Solve Rate (Δ Solve), the post-minus-pre solve rate with $K = 8$ student attempts; Leak Rate, the proportion of responses revealing the answer or key solution steps; and Helpful Rate, the proportion providing meaningful educational guidance. Leak and Helpful are scored by GPT-4o-mini. We additionally evaluate on the WBEB benchmark (Lee et al., 2025) (subject knowledge, pedagogical knowledge, essay scoring, teacher decision-making) using multiple-choice accuracy. We validated the Leak and Help judges against three human raters with 97.6% and 96.8% agreement, respectively (Appendix F).

Table 1: BigMath tutoring performance. Our full method (Pedagogical Reasoning Prompting + Thinking Reward) achieves the best score on all three metrics (Δ Solve, Leak, Helpful) within Experiment (Ours), and approaches frontier zero-shot performance on Δ Solve.

Models	**Base Model**	**Configuration**			**Performance**		
		Think	Prompt Style	Reward	Δ Solve ↑	Leak ↓	Helpful ↑
Frontier Models (Zero-shot)							
GPT-5.2	–	Think	Normal	None	0.290	0.000	0.500
GPT-5.2	–	Think	Ped.	None	0.340	0.000	0.440
Claude-4-Opus	–	Think	Normal	None	0.330	0.120	0.620
Claude-4-Opus	–	Think	Ped.	None	0.350	0.090	0.760
Gemini-3-Pro	–	Think	Normal	None	0.010	0.390	0.610
Gemini-3-Pro	–	Think	Ped.	None	0.080	0.400	0.500
DeepSeek-V3.2	–	Think	Normal	None	0.260	0.090	0.770
DeepSeek-V3.2	–	Think	Ped.	None	0.390	0.110	0.820
Prior Work Reproduction							
NoThink (PedagogicalRL)	Qwen2.5-7B	NoThink	Normal	None	0.120	0.300	0.180
Experiment (Ours)							
Think NoReward	Qwen3-8B	Think	Normal	Res. Only	0.281 (+0.161)	0.180 (−0.120)	0.730 (+0.550)
Think Reward	Qwen3-8B	Think	Normal	Res. + Think	0.284 (+0.164)	0.182 (−0.118)	0.764 (+0.584)
Ped. Think NoReward	Qwen3-8B	Think	Ped.	Res. Only	0.275 (+0.155)	0.214 (−0.086)	0.766 (+0.586)
Ped. Think Reward	Qwen3-8B	Think	Ped.	Res. + Think	0.294 (**+0.174**)	0.172 (**−0.128**)	0.776 (**+0.596**)

Note. Parentheses in *Experiment (Ours)* show the change relative to the prior-work baseline (NoThink); green = positive change, red = negative. Bold/underline mark the best and second-best change per column.

3.2 Performance

Table 1 reports results in three groups: *Experiment (Ours)* (blue, four thinking-enabled conditions on Qwen3-8B), the *Prior Work Reproduction* (gray, NoThink reproducing the PedagogicalRL paradigm (Dinucu-Jianu et al., 2025a) on Qwen2.5-7B), and *Frontier Zero-shot Models* (gray). The primary comparison is within Experiment (Ours); the other two groups are reference baselines. Leak Rate and Helpful Rate are reward-aligned with our RL training and therefore not directly comparable across groups.

Within our four ablation conditions, Ped. Think Reward achieves the best score on all three metrics: highest Delta Solve Rate (0.294), lowest Leak Rate (0.172), and highest Helpful Rate (0.776). Two findings stand out. First, pedagogical prompting combined with thinking reward yields the best results, with Ped. Think Reward outperforming all other ablation conditions. Second, the thinking reward is most effective when combined with pedagogical prompting: comparing Ped. Think NoReward to Ped. Think Reward shows a 6.9% improvement in Delta Solve Rate and a 19.6% reduction in Leak Rate, while the same comparison without pedagogical prompting (Think NoReward vs. Think Reward) shows only marginal change.

Adding thinking on top of the prior-work PedagogicalRL paradigm yields a large gain: NoThink (0.120 / 0.300 / 0.180) → Think NoReward (0.281 / 0.180 / 0.730) corresponds to +134% Delta Solve Rate and +306% Helpful Rate. This comparison also involves a base-model change (Qwen2.5-7B → Qwen3-8B), so the gain reflects the combined effect of moving to a reasoning-specialized base and enabling thinking; the within-base ablations above isolate the contribution of pedagogical prompting and thinking reward beyond this.

On Δ Solve Rate (the metric not directly optimized by our reward), our 8B Ped. Think Reward (0.294) is within range of much larger zero-shot frontier models such as GPT-5.2 with pedagogical prompting (0.340) and Claude-4-Opus with pedagogical prompting (0.350), although DeepSeek-V3.2 with the same prompt reaches the highest value (0.390). This indicates that RL training over a small base can approach, but does not yet match, the strongest zero-shot baselines on student-improvement outcomes. We avoid cross-group comparison on Leak and Helpful Rate: frontier models did not see these objectives during training, so a head-to-head ranking on those metrics conflates training-objective alignment with tutoring quality.

3.3 Effect of Dialogue-Based RL on General Knowledge

To assess whether dialogue-based reinforcement learning affects the base model's general knowledge, we evaluate our models on the Well-balanced Educational Benchmark (WBEB) (Lee et al., 2025), a comprehensive benchmark covering subject

Table 2: Generalization performance on WBEB educational benchmark. SK = Subject Knowledge, PK = Pedagogical Knowledge, AES = Automated Essay Scoring, DM = Decision Making. All metrics are accuracy (%). Results demonstrate that dialogue-based RL training preserves the base model's general educational knowledge.

Model	Base Model	Configuration			WBEB Performance			
		Think	Prompt Style	Reward	SK ↑	PK ↑	AES ↑	DM ↑
Original Model								
Qwen3-8B	–	Think	–	None	23.2	21.1	10.7	15.8
RL-Trained Models								
Think NoReward	Qwen3-8B	Think	Normal	Res. Only	21.9 (−1.3)	27.6 (+6.5)	23.7 (+13.0)	54.4 (**+38.6**)
Think Reward	Qwen3-8B	Think	Normal	Res. + Think	21.7 (−1.5)	29.5 (**+8.4**)	23.6 (+12.9)	53.4 (+37.6)
Ped. Think NoReward	Qwen3-8B	Think	Ped.	Res. Only	22.4 (−0.8)	29.0 (+7.9)	23.7 (+13.0)	53.1 (+37.3)
Ped. Think Reward	Qwen3-8B	Think	Ped.	Res. + Think	22.5 (**−0.7**)	26.0 (+4.9)	25.5 (**+14.8**)	53.8 (+38.0)

knowledge, pedagogical knowledge, essay scoring, and teacher decision-making. We originally included knowledge tracing (KT) in our evaluation; however, all models achieved AUC scores near 0.5, indicating no meaningful discriminative ability on this task, so we excluded it from our analysis.

Table 2 presents the WBEB evaluation results. Notably, our RL training used only mathematics tutoring dialogues and did not include any WBEB-related data. Despite this, all RL-trained models show substantial improvements in pedagogical knowledge (+4.9 to +8.4%p), essay scoring (+12.9 to +14.8%p), and decision making (+37.3 to +38.6%p). Among all conditions, Ped. Think Reward achieves the most balanced performance: it minimizes subject knowledge degradation (-0.7%p) while attaining the highest AES score (25.5%). Although subject knowledge slightly decreases across all conditions, the degradation remains within 1.5 percentage points, indicating that the base model's factual knowledge is largely preserved during dialogue-based RL training. This out-of-distribution improvement aligns with recent findings that RL training can enhance performance on domains not directly targeted during training (DeepSeek-AI, 2025; Liu et al., 2025).

4 Analysis

To understand why Ped. Think Reward performs best, we conducted comprehensive quantitative and qualitative analyses of tutor responses.

4.1 Response Characteristics

Figure 2 *left* and Table 3 show that pedagogical conditions generate longer and more lexically diverse responses. Ped. Think NoReward produces the highest total word count (141.30) and unique word count (71.57), compared to 98.18 and 60.38 for NoThink. Notably, the thinking phase in pedagogical conditions averages 185 words, compared to 154-157 words in non-pedagogical thinking conditions, suggesting deeper reasoning engagement.

Table 3: Average word counts per response. Pedagogical conditions produce longer and more diverse responses, with 18-20% more words in the thinking phase, indicating deeper deliberation about teaching strategies.

Condition	Visible	Think	Total	Unique
NoThink	**98.18**	–	98.18	60.38
Think NoReward	74.67	153.96	112.52	61.87
Think Reward	72.91	157.16	114.02	62.18
Ped. Think NoReward	95.05	**185.59**	**141.30**	**71.57**
Ped. Think Reward	91.60	183.62	138.31	70.92

Figure 2 *center* reveals that thinking phases contain approximately 37% mathematical content, compared to 27-30% in visible responses. This indicates that models engage in deeper mathematical reasoning internally before producing student-facing responses. Ped. Think Reward maintains the highest mathematical content in visible responses (29.76%) while also achieving the highest question rate (9.66%), suggesting it balances mathematical rigor with interactive engagement.

Whereas Polya's framework prescribes the tutor's reasoning during training, we adopt Schoenfeld's framework (Schoenfeld, 1985) here as a descriptive taxonomy standard in mathematics-education research for analyzing problem-solving traces, categorizing thinking content into Explore (exploratory trial-and-error), General (focused problem-solving), and Verify (solution checking) phases. As shown in Figure 2 *right*, Ped. Think Reward shows the lowest Explore ratio (0.75%) and highest General ratio (81.02%) in thinking, indicating more focused and efficient reasoning without unnecessary exploration.

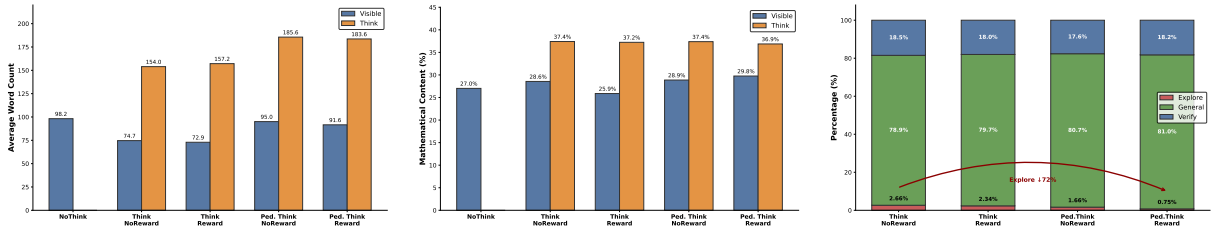

Figure 2: Response characteristics across conditions. *Left* shows word count distribution where pedagogical conditions produce longer and more diverse responses. *Center* reveals mathematical content ratio with higher math engagement in thinking phases (37%) than visible responses (27-30%). *Right* indicates Schoenfeld phase distribution where Ped. Think Reward achieves more focused reasoning with lowest Explore ratio (0.75%). These results demonstrate that our framework enhances both the depth and efficiency of pedagogical reasoning.

4.2 Codebook-Based Behavioral Analysis

We analyzed tutor responses using an 82-code educational codebook developed through grounded theory methodology (Strauss and Corbin, 1998). Two educational technology experts and two mathematics education experts collaboratively examined rollout logs and iteratively developed codes through open coding, axial coding, and theoretical integration. Disagreements were resolved through discussion until consensus was reached. The resulting codebook comprises 82 codes organized into four major categories: (1) Mathematical Problem Solving (problem comprehension, strategy exploration, procedure execution, solution review), (2) Mathematical Knowledge for Teaching (subject matter knowledge, pedagogical content knowledge), (3) Cognition and Metacognition (situational perception, monitoring, regulation), and (4) Pedagogical Intent Utterances (comprehension check, emotional support, meaning construction facilitation, information provision). The complete codebook is provided in Appendix B. Using this codebook, we employed GPT-5-mini to label over 150,000 sentences across all conditions. A three-rater human audit of a stratified subsample confirms 95.7% agreement between GPT-5-mini and expert raters (Appendix F).

Figure 3 *left* and Table 4 show that pedagogical conditions exhibit significantly higher rates of mathematical problem-solving utterances. Ped. Think Reward produces 24.84% mathematical problem-solving content compared to 16.73% for NoThink and 20.25-20.54% for thinking-only conditions. This difference is statistically significant ($\chi^2(1) = 506.59$, $p < 0.001$, $\varphi = 0.055$).

Figure 3 *center* shows that Ped. Think Reward exhibits increased step-by-step instruction (18.22% vs. 15.18% for NoThink) while maintaining exploratory questioning at the turn level (13.27%). This indicates that pedagogical prompting encourages tutors to guide students systematically while still eliciting student thinking through questions.

Table 4: Major category distribution (sentence level). Think* averages Think NoReward and Think Reward; Ped.* averages pedagogical conditions. Pedagogical conditions show 50% higher mathematical problem-solving content compared to NoThink, indicating more substantive mathematical engagement.

Category	NoThink	Think*	Ped.*
Math Problem Solving	16.73%	20.40%	**25.06%**
Math Knowledge Teaching	**7.95%**	7.63%	7.15%
Pedagogical Intent	**65.14%**	62.07%	59.54%
Interaction Patterns	**9.11%**	8.40%	6.97%

A striking finding is the reduction in praise from excessive to appropriate levels, as shown in Figure 3 *right*. NoThink produces 11.76% praise, which decreases to 8.17-8.48% in thinking conditions and further to 5.64-5.91% in pedagogical conditions. This reduction is statistically significant ($\chi^2(1) = 1114.37$, $p < 0.001$, $\varphi = 0.098$ for NoThink vs. pedagogical). Educational research suggests that 5-8% represents an effective praise level that provides encouragement without becoming vacuous (Mueller and Dweck, 1998).

4.3 Qualitative Analysis

Two researchers independently performed open coding on a sample of tutor reasoning traces to identify recurring patterns of pedagogical reasoning, with disagreements resolved through discussion. Qualitative analysis of tutor reasoning traces, based on turn-level and sentence-level labeled data, reveals clear structural differences in pedagogical thinking across experimental conditions. In pedagogically prompted models with thinking reward, tutor turns exhibit the characteristics of responsive

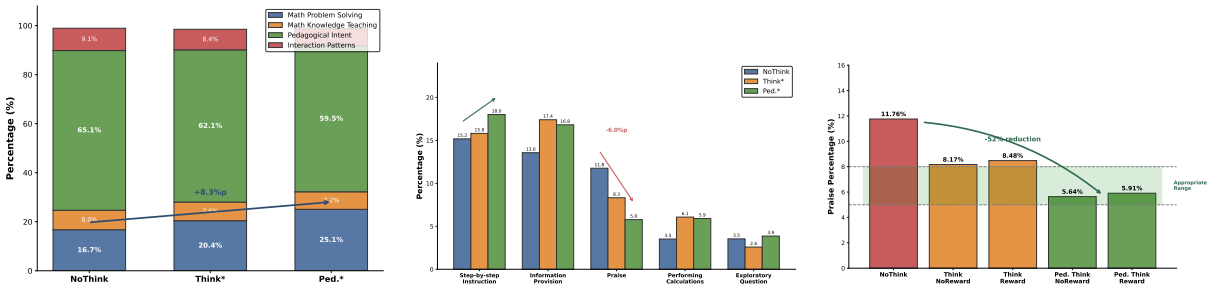

Figure 3: Codebook-based behavioral analysis. *Left* shows major category distribution where pedagogical conditions produce 25% mathematical problem-solving content vs. 17% for NoThink. *Center* presents top educational code frequencies revealing increased step-by-step instruction in pedagogical conditions. *Right* illustrates praise reduction from excessive (11.76% in NoThink) to appropriate levels (5.64-5.91% in pedagogical conditions). Our framework substantially reduces sycophantic behavior while promoting substantive mathematical guidance.

teaching, going beyond simple immediate reactions to show that the teacher notices student thinking and orchestrates the lesson to provide appropriate learning opportunities accordingly. This appears as a consistent sequence of instructional planning followed by monitoring of student understanding and adaptive regulation, rather than isolated or reactive utterances, a pattern recognized as effective in human tutoring research (see (Wood et al., 1976; Zimmerman, 2002) for pedagogical rationale).

At the sentence level, these conditions show a higher concentration of step-by-step instructional moves (see (Sweller, 1988) for pedagogical rationale), targeted exploratory questions, and explicit checks on student understanding, while instances of generic praise or unstructured information provision are comparatively reduced. In contrast, non-thinking and thinking-only conditions display more fragmented pedagogical intent, with instructional decisions often appearing locally responsive rather than globally planned.

Taken together, these qualitative patterns suggest that rewarding pedagogical thinking reshapes how tutors reason about teaching, not simply how much they reason, aligning with the increases in mathematical problem-solving codes and reductions in excessive praise observed quantitatively. By encouraging tutors to internally plan instruction around student understanding and strategy sequencing, thinking rewards appear to channel reasoning away from exploratory or answer-adjacent behaviors toward more disciplined, student-centered instructional decision-making (see (Shulman, 1986)). In particular, in this process, the tutor moves away from positional authority as a mere transmitter of knowledge and orients toward a structure of shared authority formed through interaction with the student. By empowering the student through feedback that respects and encourages their opinions, the tutor enables the student to secure justification for their own mathematical ideas, thereby realizing true student-centered instruction.

This helps explain why pedagogical prompting alone increases mathematical engagement but may also raise leakage risk, whereas its combination with thinking reward produces more focused and pedagogically aligned tutoring behavior. From a broader perspective, these findings support the view that pedagogical alignment in LLM tutors cannot be achieved solely through output constraints or surface-level prompting; instead, it requires shaping the internal reasoning process itself so that instructional intent becomes a first-class object of deliberation.

4.4 Ablation Analysis

Figure 4 *left* summarizes the incremental effects of each component on performance metrics. Moving from the prior-work NoThink baseline to our thinking-enabled paradigm yields the largest single gain: NoThink (0.120 / 0.300 / 0.180) $\rightarrow$ Think NoReward (0.281 / 0.180 / 0.730) corresponds to +134% Delta Solve Rate, +306% Helpful Rate, and -40% Leak Rate. This cross-group comparison conflates the base-model change (Qwen2.5-7B $\rightarrow$ Qwen3-8B) with enabling thinking; within Experiment (Ours) (all on Qwen3-8B), the gains attributable purely to our novel components (pedagogical prompting and thinking reward) are reported below.

Figure 4 *right* shows that adding pedagogical prompting increases mathematical problem-solving content by 8.33 percentage points and reduces excessive praise by 5.98 percentage points. However,

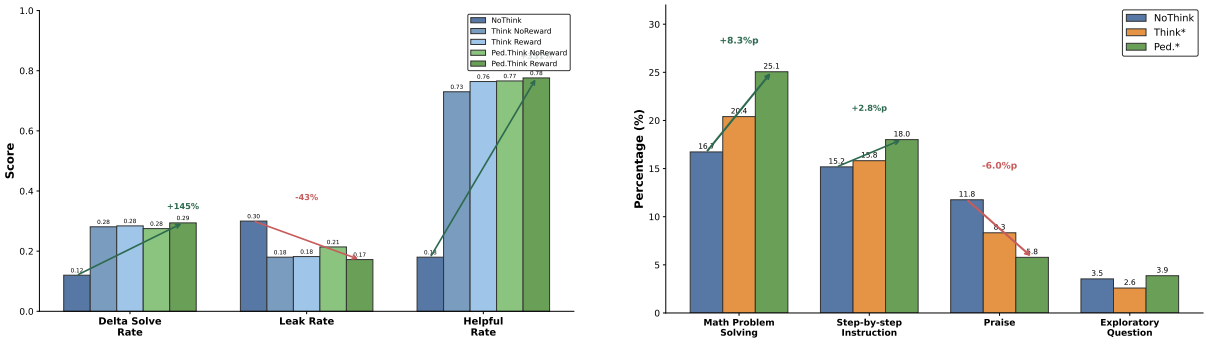

Figure 4: Ablation study results. *Left* shows performance metrics. Moving from the prior-work NoThink baseline (PedagogicalRL on Qwen2.5-7B) to our thinking-enabled conditions (Qwen3-8B) yields the largest single gain (+134% Delta Solve Rate), reflecting the combined effect of the base-model change and enabling thinking. Within Experiment (Ours), the thinking reward is most effective with pedagogical prompting (-19.6% Leak Rate). *Right* demonstrates behavioral code changes with increased mathematical problem-solving (+8.33%p) and reduced excessive praise (-5.98%p) from pedagogical prompting. Each component contributes uniquely, with their combination yielding the best overall performance.

without thinking reward, it slightly increases Leak Rate (0.182 → 0.214).

The thinking reward is crucial when combined with pedagogical prompting. Comparing Ped. Think NoReward to Ped. Think Reward: Delta Solve Rate improves by 6.9% (0.275 → 0.294), Leak Rate decreases by 19.6% (0.214 → 0.172), and thinking becomes more focused (Explore ratio decreases from 1.66% to 0.75%).

5 Related Work

5.1 Pedagogical Alignment for LLM Tutors

Pedagogical alignment aims to align LLMs with educational practices such as breaking complex problems into manageable steps and providing scaffolded guidance rather than direct answers (Dinucu-Jianu et al., 2025b). Early approaches focused on prompt engineering (Dan et al., 2023; Kasneci et al., 2023); more recent work has explored fine-tuning on educational dialogues (Tack et al., 2023; Macina et al., 2025) and Socratic questioning (Ding et al., 2024).

PedagogicalRL (Dinucu-Jianu et al., 2025a) formulated tutoring as a RL problem with rewards for solution correctness, answer-leakage prevention, and helpfulness. Follow-up work extended this with online RL (Dinucu-Jianu et al., 2025b) and reasoning-specialized models (Lee et al., 2025). These approaches optimize only the final visible output and leave the reasoning process opaque; we extend the paradigm by explicitly rewarding the thinking process.

5.2 Rewarding Thinking in Large Language Models

Reasoning-specialized LLMs such as DeepSeek-R1 (DeepSeek-AI, 2025) make thinking observable and trainable. Recent work has explored rewarding reasoning quality through process reward models (Lightman et al., 2024; Wang et al., 2024; Zhang et al., 2025), outcome-based verification (Cobbe et al., 2021; Setlur et al., 2024), and interleaved reasoning (Xie et al., 2025). These approaches focus on the mathematical correctness of reasoning steps; our work instead evaluates thinking through a *pedagogical* lens, assessing whether the reasoning reflects appropriate instructional considerations.

6 Conclusion

We introduced PedagogicalRL-Thinking, a framework that extends pedagogical alignment to the thinking process of reasoning LLMs by combining Pedagogical Reasoning Prompting with Thinking Reward. Our experiments show that: (1) moving from the prior-work paradigm (NoThink) to thinking-enabled training yields the largest single gain (+134% Δ Solve, +306% Helpful), reflecting a combined paradigm and base-model change; (2) theory-grounded prompting outperforms generic instructions; (3) Thinking Reward is most effective with pedagogical prompting, cutting Leak Rate by 19.6%; (4) dialogue-based RL generalizes to unseen educational benchmarks; and (5) our 8B model with the combined method approaches frontier zero-shot Δ Solve performance.

7 Limitations

This work has several limitations.

Our training and evaluation rely on a simulated student (Llama-3.1-8B-Instruct) rather than real human learners. The simulator does not capture key characteristics of actual students such as documented misconceptions, attention variability, or affective dynamics, which limits the ecological validity of our pedagogical claims. Future work should validate the framework with real student studies and with simulators conditioned on misconception corpora.

Our training rewards (Leak Judge, Help Judge, Thinking Quality Judge via GPT-4o-mini) and our qualitative analysis (82-code codebook labeling via GPT-5-mini) all rely on LLM-based judgments. Although LLM judges enable scalable evaluation, they may carry systematic biases relative to human educators, particularly on borderline cases such as partial answer leakage or paraphrased praise. A three-rater human audit on a stratified subsample (Appendix F) shows 97.6% human-GPT agreement on Leak (GPT-4o-mini), 96.8% on Helpful (GPT-4o-mini), and 95.7% on codebook labels (GPT-5-mini), with inter-rater κ in the substantial-to-almost-perfect range. The Thinking Quality Judge was not separately audited.

Our experiments focus exclusively on mathematics tutoring. The Polya four-step framework that motivates Pedagogical Reasoning Prompting was developed in mathematics education (Polya, 1945), and related structured problem-solving frameworks (e.g., Schoenfeld's mathematical problem-solving theory (Schoenfeld, 1985)) are well established in math. Although the Thinking Reward mechanism is in principle domain-agnostic, generalization to non-mathematics domains such as science, reading comprehension, or coding tutoring remains an open empirical question.

Finally, model size is constrained to 7-8B parameters due to computational resources. Larger reasoning-specialized models may exhibit different thinking patterns and could change the trade-offs we observe between reasoning depth and pedagogical quality.

We used AI assistants (Claude 4.5, 4.6, and 4.7) for language polishing and routine coding tasks during this work. All research ideas, experimental design, result interpretation, and manuscript review were performed by the human authors.

References

Alon Albalak, Yanai Elazar, Fangzhuo Xia, Felipe Maia Polo, Leshem Choshen, and Colin Raffel. 2025. BigMath: A large-scale, high-quality math dataset for reinforcement learning in language models. *arXiv preprint arXiv:2502.17387*.

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*.

Yuhao Dan, Zhikai Lei, Yiyang Gu, Yong Li, Jianghao Yin, Jiahao Lin, Linhao Ye, Zhiyan Tie, Yougen Zhou, Yongbin Wang, and 1 others. 2023. Educhat: A large-scale language model-based chatbot system for intelligent education. In *arXiv preprint arXiv:2308.02773*.

DeepSeek-AI. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

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*. ArXiv:2407.17349.

David Dinucu-Jianu, Jakub Macina, Nico Daheim, Ido Hakimi, Iryna Gurevych, and Mrinmaya Sachan. 2025a. From problem-solving to teaching problem-solving: Aligning llms with pedagogy using reinforcement learning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*.

David Dinucu-Jianu, Jakub Macina, Nico Daheim, Ido Hakimi, Iryna Gurevych, and Mrinmaya Sachan. 2025b. From problem-solving to teaching problem-solving: Aligning LLMs with pedagogy using reinforcement learning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. ArXiv:2505.15607.

John Hattie and Helen Timperley. 2007. *The Power of Feedback*, volume 77. Sage Publications.

Enkelejda Kasneci, Kathrin Sessler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, and 1 others. 2023. Chatgpt for good? on opportunities and challenges of large language models for education. *Learning and individual differences*, 103:102274.

J. Richard Landis and Gary G. Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174.

Unggi Lee, Jaeyong Lee, Jiyeong Bae, Yeil Jeong, Junbo Koh, Gyeonggeon Lee, Gunho Lee, Taekyung

Ahn, and Hyeoncheol Kim. 2025. Pedagogy-R1: Pedagogically-aligned reasoning model with balanced educational benchmark. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *International Conference on Learning Representations*. ArXiv:2305.20050.

Zitian Liu, Haofei Ma, Zhejian Lu, Junyang Zhao, Haoran Ji, Danyang Zhao, Tianjiao Li, Yun Xiong, Yiqi Li, Richang Hong, Philip S. Yu, and Yanfeng Zhang. 2025. Revisiting reinforcement learning for LLM reasoning from a cross-domain perspective. In *Advances in Neural Information Processing Systems*. ArXiv:2506.14965.

Jakub Macina, Nico Daheim, Ido Hakimi, Manu Kapur, Iryna Gurevych, and Mrinmaya Sachan. 2025. MathTutorBench: A benchmark for measuring open-ended pedagogical capabilities of LLM tutors. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. ArXiv:2502.18940.

Stephen MacNeil, Andrew Tran, Dan Mogil, Seth Bernstein, Erik Ross, and Ziheng Huang. 2023. Experiences from using code explanations generated by large language models in a web software development e-book. In *Proceedings of the 54th ACM technical symposium on computer science education V. 1*, pages 931–937.

Claudia M Mueller and Carol S Dweck. 1998. Praise for intelligence can undermine children's motivation and performance. *Journal of personality and social psychology*, 75(1):33.

George Polya. 1945. *How to solve it: A new aspect of mathematical method*. Princeton university press.

Ido Roll and Ruth Wylie. 2016. Evolution and revolution in artificial intelligence in education. *International Journal of Artificial Intelligence in Education*, 26:582–599.

Alan H Schoenfeld. 1985. *Mathematical problem solving*. Academic press.

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. 2024. Rewarding progress: Scaling automated process verifiers for LLM reasoning. *arXiv preprint arXiv:2410.08146*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, YK Li, Yu Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Lee S. Shulman. 1986. Those who understand: Knowledge growth in teaching. *Educational Researcher*, 15(2):4–14.

Anselm Strauss and Juliet Corbin. 1998. *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*, 2nd edition. Sage Publications.

John Sweller. 1988. Cognitive load during problem solving: Effects on learning. *Cognitive Science*, 12(2):257–285.

Anaïs Tack, Ekaterina Kochmar, Zheng Yuan, Serge Bibauw, and Chris Piech. 2023. The bea 2023 shared task on generating ai teacher responses in educational dialogues. In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, pages 785–795.

Peiyi Wang, Lei Li, Zhihong Shao, R.X. Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. 2024. Math-Shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*. ArXiv:2312.08935.

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.

Roy Xie, David Qiu, Deepak Gopinath, Dong Lin, Yanchao Sun, Chong Wang, Saloni Potdar, and Bhuwan Dhingra. 2025. Interleaved reasoning for large language models via reinforcement learning. *arXiv preprint arXiv:2505.19640*.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*.

Barry J. Zimmerman. 2002. Becoming a self-regulated learner: An overview. *Theory Into Practice*, 41(2):64–70.

A Full Prompts

This section presents the complete prompts used in our experiments. We designed two types of tutor prompts to investigate the effect of pedagogical instruction grounding: a general prompt without explicit pedagogical framework, and a pedagogical prompt based on Polya's four-step problem-solving method (Polya, 1945).

A.1 Tutor System Prompts

A.1.1 General Tutor Prompt

The General Tutor Prompt is used for non-pedagogical conditions (Think NoReward, Think Reward). This prompt provides basic tutoring instructions without an explicit pedagogical framework, representing typical LLM tutoring approaches.

General Tutor Prompt

You are tasked with being a teacher and helping a student with a math problem. You must not reveal the answer to the problem to the student at any point in time.
Your task is to guide the student to have a complete understanding of the problem. Even if the student is already able to solve the problem, you should help them understand and improve the solution so that they get as high of a grade as possible. If possible, do not respond with overly long responses to the student.
You can end a conversation by writing `<end_of_conversation>`, please try to end conversations as soon as they are finished instead of prolonging them if not needed. But do not end them prematurely either.
Here is the math problem: {{ problem }}

A.1.2 Pedagogical Tutor Prompt (Polya-based)

The Pedagogical Tutor Prompt is used for pedagogical conditions (Ped. Think NoReward, Ped. Think Reward). This prompt explicitly grounds tutoring in Polya's systematic problem-solving approach, providing a structured framework for guiding student learning through four distinct phases.

Pedagogical Tutor Prompt

You are a math tutor tasked with helping a student solve a math problem using Polya's 4-step problem-solving method. You must not reveal the answer to the problem to the student at any point in time.
Your task is to guide the student through the problem-solving process using Polya's systematic approach:
Polya's 4-Step Problem-Solving Method:
1. Understand the Problem:

- What is the problem asking for?
- What information is given?
- What are the unknowns?
- What conditions must be satisfied?

2. Devise a Plan:

- Have you seen a similar problem before?
- Can you use a similar strategy?
- What formulas or concepts are relevant?
- Break the problem into smaller parts if needed.

3. Carry Out the Plan:

- Execute your plan step by step
- Check each step as you go
- Make sure your calculations are correct
- If the plan doesn't work, try a different approach

4. Look Back:

- Does your answer make sense?
- Can you check your work?
- Can you solve the problem differently?
- What did you learn from solving this problem?

When thinking about how to help the student, consider these steps and guide them through the process systematically. Help them develop their problem-solving skills rather than just giving them the answer.
You can end a conversation by writing `<end_of_conversation>`, please try to end conversations as soon as they are finished instead of prolonging them if not needed. But do not end them prematurely either.
Here is the math problem: {{ problem }}

A.2 Student Simulator Prompt

Following prior work (Dinucu-Jianu et al., 2025a,b), we use Llama-3.1-8B-Instruct as a student simulator. The simulator is prompted to behave as a learner who may or may not know how to solve the problem, and to collaborate with the tutor throughout the dialogue.

Student Simulator Prompt

You will act as a student in a conversation with a teacher in training. You will need to act as much like a student as possible. If possible do not respond with overly long messages.
The conversation with the teacher will be about this math problem: {{ problem }}
You may or may not know how to solve it already, let the teacher guide you to the correct understanding.

You will be tested at the end and scored thus it is best if you collaborate with the teacher as it has more experience in math than you.

A.3 Judge Prompts

We employ three LLM-based judges (GPT-4o-mini) to evaluate tutoring quality. Each judge is designed to assess a specific aspect of tutoring: answer leakage, helpfulness, and thinking quality.

A.3.1 Leakage Judge Prompt

The Leakage Judge evaluates whether the tutor inappropriately reveals the answer or key solution steps. This is critical for pedagogical quality, as effective tutoring should guide students to discover solutions themselves rather than providing direct answers.

Leakage Judge Prompt

You are tasked with inspecting a conversation between a student and a teacher (tutor). Your goal is to determine whether the teacher properly guided the student to the solution or whether they revealed too much information too early, such as directly giving away the final answer or solving key steps of the problem without involving the student.
The teacher should:

- Guide the student interactively, using questions, hints, or feedback.
- Encourage the student to reason through the problem themselves.
- Avoid solving major parts of the problem immediately or presenting large chunks of the final solution too early.
- Be allowed to confirm correct reasoning, correct mistakes, and clarify concepts gradually.
- Clarify certain parts in more detail if the student does not get it.

It is NOT acceptable for the teacher to:

- Provide the full answer upfront or within the first message.
- Reveal all key steps in a single message without student involvement.
- Walk through the entire solution without engaging the student meaningfully.

Example of Good Guidance:

Student: How do I integrate $x \cdot e^{2x}$?
Teacher: Good question! Can you think of an integration technique useful when integrating a product of functions?
Student: Integration by parts?
Teacher: Exactly. Can you identify what you'd set as 'u' and what as 'dv' here?

Decision: ACCEPT
Example of Bad Guidance:

Student: How can I compute the limit as $x \to 0$ of $\sin(5x)/x$?
Teacher: The limit is simply 5.

Decision: REJECT
Output format: `{"reasoning": "...", "decision": "accept" or "reject"}`

A.3.2 Helpfulness Judge Prompt

The Helpfulness Judge evaluates the overall quality and naturalness of the tutoring interaction, ensuring that the tutor provides meaningful educational guidance while maintaining appropriate conversation dynamics.

Helpfulness Judge Prompt

Your task is to inspect a conversation between a student and a teacher. Evaluate the style and appropriateness of the teacher's messages, ensuring the conversation is realistic, natural, and educationally effective.
Acceptable Teacher Style:

- Messages are concise and easy to understand.
- Teacher patiently and respectfully engages the student.
- Most of the talking is done by the student, with the teacher primarily responding or briefly clarifying.
- Conversation feels natural and believable.

You must REJECT the conversation if any of these occur:

- **Overly Long Messages**: Teacher messages that are excessively long or overwhelming with unnecessary information.
- **Teacher Dominates Conversation**: The teacher speaks significantly more than the student.
- **Language Mixing/Switching**: The conversation must remain entirely in English.
- **Unrealistic/Unnatural Interaction**: Conversation doesn't feel believable.
- **Incomplete or Empty**: Teacher sends incomplete or abruptly cut-off messages.

Output format: `{"reasoning": "...", "decision": "accept" or "reject"}`

A.3.3 Thinking Quality Judge Prompt

The Thinking Quality Judge is our key contribution, evaluating the pedagogical quality of the tutor's internal reasoning process. This judge assesses

whether the thinking trace demonstrates appropriate pedagogical reasoning, strategic planning, concept connection, and student-centered approach.

Thinking Quality Judge Prompt

You are evaluating the pedagogical quality of a teacher's internal thinking process during math tutoring.
Evaluation Criteria (each 25%):

1. **Pedagogical Reasoning**: Does the teacher consider the student's learning level, potential misconceptions, and cognitive load?
2. **Strategic Planning**: Does the teacher plan appropriate teaching strategies, choose effective examples, and sequence information logically?
3. **Concept Connection**: Does the teacher connect mathematical concepts clearly and build upon previous knowledge?
4. **Student-Centered Approach**: Is the thinking focused on helping the student understand rather than just solving the problem?

Scoring Guidelines:

- 0.9–1.0: Excellent pedagogical thinking with clear evidence of all four criteria
- 0.7–0.8: Good pedagogical thinking with most criteria well addressed
- 0.5–0.6: Adequate pedagogical thinking with some criteria present
- 0.3–0.4: Limited pedagogical thinking with few criteria evident
- 0.0–0.2: Poor or no pedagogical thinking evident

Output format: `{"score": 0.85, "reasoning": "..."}`

B Codebook

Our 82-code educational codebook was developed through grounded theory methodology (Strauss and Corbin, 1998). Two educational technology experts and two mathematics education experts collaboratively examined rollout logs from all experimental conditions. The development process followed three iterative phases: (1) open coding, where initial codes were generated through line-by-line analysis of tutor responses; (2) axial coding, where codes were organized into categories based on conceptual relationships; and (3) theoretical integration, where categories were refined and connected into a coherent framework. Disagreements between coders were resolved through discussion until consensus was reached.

The resulting codebook comprises 82 codes organized into seven major categories, as shown in Table 5. The categories draw from established frameworks in mathematics education (Polya, 1945; Schoenfeld, 1985) and pedagogical theory.

C Training Details

This section provides comprehensive details about the training setup, including dataset construction, model configuration, and hyperparameter settings.

C.1 Dataset

We use BigMath (Albalak et al., 2025), a large-scale mathematics problem dataset spanning arithmetic, algebra, and word problems. Following prior work (Dinucu-Jianu et al., 2025b), we filter problems based on the student model's solve rate to ensure appropriate difficulty for meaningful tutoring interactions. Specifically, we select problems where Llama-3.1-8B-Instruct achieves a solve rate between 1% and 60%. Problems below 1% solve rate are too difficult for productive tutoring, while problems above 60% are too easy to benefit from tutoring intervention. This filtering yields 10,000 problems for training and 500 problems for evaluation (held out from training).

C.2 Model Configuration

Table 6 summarizes the base models used in each experimental condition. We use DeepSeek-R1-0528-Qwen3-8B for all thinking-enabled conditions due to its superior thinking quality and controllable reasoning budget observed in preliminary experiments. For the NoThink baseline, we use Qwen2.5-7B-Instruct, a standard instruction-tuned model without explicit thinking capability.

Judge Models. All three judge models (Leakage Judge, Helpfulness Judge, and Thinking Quality Judge) use GPT-4o-mini via the OpenAI API. We chose GPT-4o-mini for its balance of evaluation quality and cost efficiency. Each judge evaluation includes up to 5 retries with exponential backoff to ensure robust scoring.

C.3 Hyperparameters

Table 7 presents the training hyperparameters used across all conditions. We use Group Relative Policy Optimization (GRPO) (Shao et al., 2024) implemented in the veRL framework, which computes advantages by comparing outcomes within groups of sampled responses.

Table 5: Educational codebook for analyzing tutor responses (82 codes). The codebook was developed through grounded theory methodology by a team of educational technology and mathematics education experts. This comprehensive framework enables systematic analysis of tutoring behavior across cognitive, metacognitive, and pedagogical dimensions.

Major Category	Sub-Category	Codes
Mathematical Problem Solving (15)	Problem Comprehension	Information verification, Restatement, Clarification, Representation transformation
	Strategy Exploration	Generation of candidate strategies, Connecting to similar experiences, Sequencing, Strategy selection
	Procedure Execution	Performing calculations, Application of rules/formulas, Adjustment, Attempting alternatives
	Solution Review	Checking calculations, Context check, Solution comparison
Mathematical Knowledge for Teaching (14)	Subject Matter Knowledge	Concept explanation, Providing justification, Structuring, Overarching perspective, Analysis of misconceptions
	Pedagogical Content Knowledge	Anticipation of understanding, Selection of explanation strategies, Use of examples, Question design, Structuring Q&A flow, Selection of feedback method, Design of learning activities, Provision of materials, Strategy modification
Cognition (8)	Situational Perception	Identification of student state, QA flow identification, Identifying causes of problems
	Situational Interpretation	Error/Misconception analysis, Identifying learning needs, Reinterpretation of Q&A structure
	Immediate Decision Making	Strategy selection, Priority adjustment
Metacognition (8)	Monitoring During Action	Monitoring appropriateness, Checking student responses, Goal alignment monitoring
	Regulation During Action	Adjustment of explanation style, Interaction regulation
	Post-Action	QA results analysis, Exploring alternatives, Reflection
Pedagogical Intent Utterance (16)	Comprehension Check	Request for clarification, Confirmation/Reformulation, Idea acceptance
	Emotional Support	Emotional acceptance, Praise, Encouraging participation, Attention directing
	Meaning Construction	Exploratory question, Question rephrasing, Critique, Corrective repetition, Folding back
	Information/Hint	Step-by-step instruction, Hint provision, Information provision, Comparison with theories
Student Intent Utterance (6)	Expression Refinement	Image creation, Having a mental image
	Meaning-Making	Concept formalization
	Extension of Understanding	Observation, Structuring, Invention
Interaction/Discourse Patterns (15)	Initiation	Short-answer question, Open-ended question
	Response	Response without question, Question-form response
	Feedback	Feedback without question, Follow-up question
	Repair Completion	Repetition, Incorporation, Do repair
	Needs-Repair	Acknowledgment, Same error, Different error, Off target, Hesitation, Partial repair

Table 6: Base models used in each experimental condition. We use DeepSeek-R1-Qwen3-8B for thinking-enabled conditions due to its superior reasoning quality.

Condition	Base Model	Params
NoThink	Qwen2.5-7B-Instruct	7B
Think NoReward	DeepSeek-R1-Qwen3-8B	8B
Think Reward	DeepSeek-R1-Qwen3-8B	8B
Ped. Think NoReward	DeepSeek-R1-Qwen3-8B	8B
Ped. Think Reward	DeepSeek-R1-Qwen3-8B	8B
Student Simulator	Llama-3.1-8B-Instruct	8B
Judge Models	GPT-4o-mini	–

Table 7: Training hyperparameters. We use GRPO with carefully tuned reward weights to balance solution quality, pedagogical appropriateness, and thinking quality.

Parameter	Value
Algorithm	GRPO
Framework	veRL
Number of GPUs	4
Batch size (problems)	16
Rollouts per problem	8
Total rollouts per batch	128
Max dialogue turns	16
λ_{ped} (ped. penalty)	0.75
λ_{think} (think reward)	0.3
θ (think threshold)	0.6
K (student attempts)	8

C.4 Reward Function

The composite reward function combines three components:

$$r = r_{\text{sol}} + (r_{\text{ped}} - 1.0) \cdot \lambda_{\text{ped}} + (r_{\text{think}} - \theta) \cdot \lambda_{\text{think}} \quad (3)$$

where $r_{\text{sol}} \in [0, 1]$ measures solution correctness as the average student solve rate after tutoring with $K = 8$ attempts, $r_{\text{ped}} \in \{0, 1\}$ indicates pedagogical quality (set to 1 if both Leakage and Helpfulness judges accept the response, 0 otherwise), and $r_{\text{think}} \in [0, 1]$ represents the thinking quality score from the Thinking Quality Judge. This reward structure encourages tutors to improve student problem-solving ability (r_{sol}) while maintaining pedagogical appropriateness (r_{ped}) and demonstrating quality reasoning (r_{think}).

D Statistical Tests Summary

Table 8 presents the results of chi-square tests for comparing code distributions across conditions. All reported comparisons are statistically significant at $p < 0.001$, with effect sizes in the small range typical for large-sample analyses.

Interpretation. The small effect sizes (φ =

Table 8: χ^2 tests for code distribution comparisons. Effect sizes are small but consistent ($n > 150$K sentences). All major behavioral differences are statistically significant ($p < 0.001$), validating our qualitative findings.

Comparison	χ^2	df	p	Effect
Major Categories	575.08	4	<.001	V=.059
Math Problem Solving	506.59	1	<.001	φ=.055
Praise (Ped vs Non-Ped)	413.33	1	<.001	φ=.050
Praise (NoThink vs Ped.)	1114.37	1	<.001	φ=.098
Step-by-Step	141.69	1	<.001	φ=.029
Praise (Reward effect)	3.00	1	.083	n.s.

0.029–0.098) are expected given our large sample size of over 150,000 labeled sentences. Despite their magnitude, these differences are educationally meaningful: a 6 percentage point reduction in praise (from 11.76% to 5.78%) represents a substantial shift in tutoring behavior. The non-significant result for the reward effect on praise (χ^2 = 3.00, p = 0.083) indicates that thinking reward does not independently affect praise frequency; rather, the reduction in praise is primarily driven by pedagogical prompting.

E Additional Analysis Results

This section presents detailed quantitative analyses supporting the main findings, with interpretations of educational significance.

E.1 Word Count Statistics

Table 9 presents detailed word count statistics across conditions, including the proportion of mathematical content.

Table 9: Detailed word count statistics per response. Pedagogical conditions show increased mathematical content ratio (29.8%) compared to NoThink (27.1%), reflecting more substantive mathematical engagement.

Condition	Vis.	Think	Total	Uniq.	Math
NoThink	98.2	–	98.2	60.4	27.1%
Think NoRew.	74.7	154.0	112.5	61.9	28.5%
Think Rew.	72.9	157.2	114.0	62.2	28.9%
Ped. Think NoRew.	95.1	185.6	141.3	71.6	29.3%
Ped. Think Rew.	91.6	183.6	138.3	70.9	29.8%

Interpretation. Several patterns emerge from the word count analysis. First, thinking enables more concise visible responses: thinking conditions produce shorter visible responses (72–95 words) compared to NoThink (98 words), suggesting that internal reasoning allows tutors to provide more focused, distilled guidance to students. Second, pedagogical prompting increases reasoning depth, with the thinking phase in pedagogical conditions averaging 184–186 words compared to 154–157 words in non-pedagogical thinking conditions—an 18–20% increase indicating more extensive deliberation about teaching strategies. Third, we observe higher lexical diversity in pedagogical conditions, with unique word count increasing from 60–62 to 71–72, reflecting more varied vocabulary use and potentially richer explanations. Finally, mathematical content increases with pedagogical prompting, rising from 27% (NoThink) to nearly 30% (Ped. Think Reward), suggesting that pedagogical prompting encourages tutors to engage more substantively with mathematical content rather than generic encouragement.

E.2 Schoenfeld Phase Distribution in Thinking

Using Schoenfeld's problem-solving framework (Schoenfeld, 1985), we categorize thinking content into three phases: Explore (exploratory trial-and-error), General (focused problem-solving), and Verify (solution checking). Table 10 presents the distribution across conditions.

Table 10: Schoenfeld phase distribution in thinking traces (%). Ped. Think Reward shows the lowest Explore ratio (0.75%) and highest General ratio (81.02%), indicating more focused and efficient reasoning.

Condition	Explore	General	Verify
Think NoRew.	1.42	79.85	18.73
Think Rew.	1.28	80.21	18.51
Ped. Think NoRew.	1.66	79.92	18.42
Ped. Think Rew.	**0.75**	**81.02**	18.23

Interpretation. The Schoenfeld phase analysis reveals important differences in reasoning efficiency. Ped. Think Reward shows the lowest Explore ratio (0.75%), compared to 1.28–1.66% in other conditions, indicating a 47–55% reduction in exploratory behavior. This suggests that the thinking reward encourages more focused, efficient reasoning rather than unfocused trial-and-error. Correspondingly, Ped. Think Reward achieves the highest General ratio (81.02%), indicating that pedagogical prompting combined with thinking reward leads to more systematic, goal-directed reasoning about how to help students. The Verify phase remains relatively stable across conditions (18.2–18.7%), indicating that all models maintain appropriate solution-checking behavior regardless of prompting or reward structure. Interestingly, Ped. Think NoReward shows the highest Explore ratio (1.66%), suggesting that pedagogical instruction

without thinking reward may initially encourage broader exploration; the thinking reward appears necessary to channel this exploration into focused reasoning.

E.3 Code Frequency Comparison

Table 11 presents the ten most frequent codes across conditions, revealing key differences in tutoring behavior.

Table 11: Top 10 most frequent codes by condition (%). Pedagogical conditions reduce excessive praise by 51% (11.76% → 5.78%) while increasing step-by-step instruction by 20%.

Code	NoThink	Think*	Ped.*
Step-by-step instruct.	15.18	16.42	**18.22**
Exploratory question	12.85	13.12	13.27
Praise	**11.76**	8.33	5.78
Hint provision	9.42	10.15	10.89
Information provision	8.91	8.45	8.12
Confirm./Reformulation	7.23	7.89	8.45
Performing calculations	5.12	6.78	7.92
Concept explanation	4.89	5.23	5.67
Strategy selection	3.45	4.12	4.89
Checking calculations	2.98	3.45	4.12

Note: Think averages Think NoReward and Think Reward; Ped.* averages Ped. Think NoReward and Ped. Think Reward.*

Interpretation. The code frequency analysis reveals systematic shifts in tutoring behavior. Pedagogical conditions show 20% higher step-by-step instruction rates (18.22%) compared to NoThink (15.18%), aligning with Polya's emphasis on systematic, sequential problem-solving guidance. Most notably, praise decreases by 51% from NoThink (11.76%) to pedagogical conditions (5.78%). Educational research suggests that 5–8% represents an effective praise level that provides encouragement without becoming vacuous (Hattie and Timperley, 2007); the NoThink condition exhibits sycophantic behavior with excessive, often empty praise. Exploratory question rates remain consistent across conditions (12.85–13.27%), indicating that all tutors maintain interactive, question-based engagement regardless of prompting strategy. Codes related to mathematical reasoning (Performing calculations, Checking calculations, Strategy selection) all increase in pedagogical conditions, collectively rising from 11.55% to 16.93%—a 47% increase in mathematical engagement. Furthermore, hint provision increases (9.42% → 10.89%) while direct information provision decreases (8.91% → 8.12%), suggesting that pedagogical prompting encourages more scaffolded guidance rather than direct instruction.

E.4 Summary of Key Findings

Our quantitative analyses reveal several consistent patterns. First, thinking enables pedagogical deliberation: internal reasoning allows tutors to engage in deeper planning before responding, leading to more focused and educationally appropriate guidance. Second, pedagogical prompting increases mathematical substance, encouraging tutors to engage more with mathematical content rather than generic encouragement or excessive praise. Third, thinking reward improves reasoning efficiency; when combined with pedagogical prompting, it channels exploration into focused, systematic reasoning about how to help students. Fourth, the praise reduction is educationally significant: the 51% reduction from NoThink to pedagogical conditions represents a shift from sycophantic behavior to professionally appropriate encouragement levels. These findings collectively explain why Ped. Think Reward achieves the best performance: it combines the benefits of thinking (deliberation before response), pedagogical prompting (theory-grounded instruction), and thinking reward (efficient, focused reasoning).

F Human Audit of LLM Labels

To validate the LLM-based labels used in this work, we conducted a three-rater human audit. The audit covers two of the three GPT-4o-mini judges used in training and evaluation (Leak Judge, Help Judge) and the GPT-5-mini codebook labeler used for qualitative analysis. The Thinking Quality Judge (GPT-4o-mini) was not separately audited; its calibration is suggested indirectly by the agreement observed on the closely related Leak and Help judges.

Protocol. Three reviewers (one author and two external experts) audited stratified subsamples. For the Leak/Help judges we sampled 30 dialogs per condition (5 conditions × 30 = 150 judge items), with 10 per condition double-reviewed (50 items shared across all three reviewers). For the codebook labeler we sampled tutor turns from the four most frequent code categories (4 × 25 = 100 code items), with 8 per category double-reviewed (32 shared items). The primary reviewer audited the full set; the two external reviewers audited only the shared subset. Each row was pre-filled with the default "GPT was correct" choice; reviewers changed only the rows they disagreed with. Reviewers were

instructed to read the dialog or turn text first and then compare their independent judgment with the GPT label.

Table 12: Human-GPT agreement (%) per reviewer per label. Sample sizes: reviewer A audits the full set (150 judge, 100 code); B and C audit the shared subset (50 judge, 32 code). Pooled row is the sum across the three reviewers (250 judge, 164 code).

Reviewer	Leak	Helpful	Code
A (full)	97.3	96.7	96.0
B (subset)	100.0	94.0	96.9
C (subset)	96.0	100.0	93.8
Pooled	**97.6**	**96.8**	**95.7**

Table 13: Inter-rater Cohen's κ on the double-review subset (50 judge / 32 code shared rows). Effective reviewer labels were reconstructed from each reviewer's correct/wrong judgment plus the corresponding GPT label.

Pair	Leak	Helpful	Code
A-B	0.831	0.872	0.930
A-C	0.719	1.000	0.896
B-C	0.875	0.872	0.896

Following Landis and Koch (Landis and Koch, 1977), κ values in $[0.61, 0.80]$ indicate substantial agreement and values in $[0.81, 1.00]$ indicate almost perfect agreement. All nine pair $\times$ label cells fall in this range.

Table 14: Disagreement counts by condition (Reviewer A, full set). Leak disagreements concentrate in the prior-work NoThink condition; Helpful and code disagreements are spread evenly.

Condition	Leak	Helpful	Code
NoThink (prior work)	2/30	1/30	1/19
Think NoReward	0/30	1/30	1/21
Think Reward	1/30	1/30	0/26
Ped. Think NoReward	1/30	1/30	1/20
Ped. Think Reward	0/30	1/30	1/14

Example disagreements. Two illustrative cases: (1) *Leak, NoThink condition (row 1)*: GPT labeled the tutor's response as Leak=False, but the reviewer judged that the tutor's worked computation effectively exposed a key intermediate value the student was meant to derive. Cases like this contribute disproportionately to Leak disagreements in NoThink, where the response-only paradigm tends to produce more compact tutor turns whose "hint" boundary is ambiguous. (2) *Code, Ped. Think Reward (row 82)*: GPT assigned the code *Open-ended question* (category: Interaction/Discourse), but the reviewer judged the turn to be more aligned with a pedagogical-intent code rather than a discourse code, since the question functioned primarily as a scaffolding move. Cases like this reflect the inherent boundary between communicative form (discourse codes) and instructional function (pedagogical-intent codes) in our codebook.