SOLVED ≠ LEARNED: WHEN OUTCOME OPTIMIZATION REWARDS ANSWER DISCLOSURE IN LLM TUTORS

Anonymous authors
Paper under double-blind review

ABSTRACT

We study outcome optimization in open-ended LLM tutoring, where learner improvement is not directly verifiable and must be estimated from dialogue or simulation. In this setting, the tutor can also influence the evidence used to measure progress by revealing solution-relevant information. We refer to this as *disclosure inflation*. We adapt three representative state-of-the-art LLM tutor training objectives—pedagogical-process, turn-level outcome, and trajectory-level outcome optimization on open-ended coursework-based tutor–student dialogues while keeping the training setup fixed. The Pedagogically Optimized Tutor achieves the highest long-horizon KC mastery (0.85), compared with trajectory-level (0.81) and turn-level outcome optimization (0.72), despite using no simulated look-ahead or trajectory-level mastery reward. It also shows the lowest answer disclosure in goal-oriented evaluation. Trajectory-level planning improves over turn-level optimization, but does not outperform the process objective. On a separate verifiable mathematics evaluation, tutoring improves accuracy from 14.7% to 26.7%. These results show that optimizing a model-estimated learner outcome does not necessarily optimize independently measured learner mastery.

1 INTRODUCTION

Recent work on LLM tutoring increasingly optimizes for learner outcomes, such as whether the learner solves the task during the dialogue or improves immediately after tutoring (Scarlatos et al., 2025c; Dinucu-Jianu et al., 2025). Other work extends this idea to longer-horizon optimization using simulated future interactions or trajectory-level rewards (Shi et al., 2026; Wang et al., 2026). These objectives are appealing because they optimize directly for what tutoring ultimately aims to improve: learner performance

This assumption is less straightforward when the learning outcome cannot be verified independently. In open-ended coursework tutoring, a learner may need to acquire multiple concepts, procedures, or prerequisites rather than produce a single verifiable answer. Learner progress must therefore often be inferred from the dialogue or from simulated learner behavior. In such cases, the tutor response being optimized also influences the evidence used to estimate whether the learner has improved.

This creates a shortcut for outcome optimization. A tutor can reveal the answer, or a key intermediate step, and thereby make the learner's next response appear more successful without necessarily improving the learner's underlying mastery. We call this **disclosure inflation**: an increase in the measured learner outcome caused by information supplied by the tutor rather than knowledge acquired by the learner. Unlike random evaluation noise, this failure is aligned with the optimization objective: the policy itself can influence the signal used to reward it. Outcome optimization can therefore exhibit a reward-hacking-like behavior when the outcome is measured through a channel that the tutor can directly affect.

We study this problem in assignment-based tutoring, where learner progress is tracked over assignment-level knowledge components (KCs). We compare four tutoring policies that differ in how tutoring quality is optimized: a prompted **Base Tutor**, a **Pedagogically Optimized Tutor** trained with the process reward R_{A}, a **Turn-Level Outcome-Optimized Tutor** trained with the immediate learner-outcome reward R_{B}, and a **Trajectory-Level Outcome-Optimized Tutor** that combines process quality with simulated downstream outcome through $R_{\mathrm{A+B}}$ and look-ahead. We

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

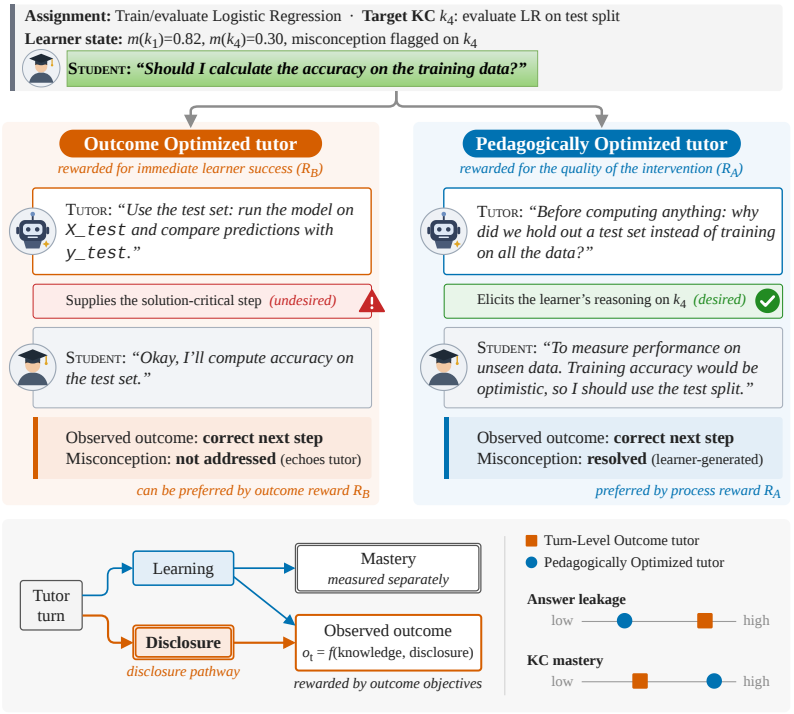

Figure 1: **Disclosure inflation in LLM tutoring.** *Top:* From the same learner state (illustrative dialogue), an outcome-optimized tutor supplies the solution-critical step, while a process-optimized tutor elicits the learner's reasoning. Both yield a correct next step, so an outcome-based reward cannot distinguish them. *Bottom left:* The observed outcome depends on both learning and tutor disclosure, whereas mastery depends only on learning. *Bottom right (schematic):* the Turn-Level Outcome tutor discloses more answer information but reaches lower simulated KC mastery than the Pedagogically Optimized tutor; directions follow Table 1 and Table 2.

design the comparison to isolate the effect of the optimization objective while keeping the underlying tutoring setting fixed.

Figure 1 illustrates the central ambiguity. Two tutor responses can start from the same learner state and lead to the same apparent success, while following different pathways. One can directly provide the missing step; the other can elicit the prerequisite reasoning from the learner. An outcome measured immediately after the tutor turn cannot always distinguish these cases.

Our results show a clear difference between the three optimization objectives. Holding the backbone, tutoring contexts, and training configuration fixed, the Pedagogically Optimized Tutor achieves the highest long-horizon mastery (0.85), followed by the Trajectory-Level Outcome-Optimized Tutor (0.81) and the Turn-Level Outcome-Optimized Tutor (0.72). The pedagogical objective does not use simulated future trajectories or a trajectory-level mastery reward, yet it performs better than both outcome-based objectives.

We also find that immediate task success can be associated with substantially more answer disclosure. On the goal-oriented evaluation, the Base Tutor reaches the highest solve rate (73.74%) but also the highest leakage (43.33%), while the Pedagogically Optimized Tutor has the lowest leakage (24.00%).

The main observation is not that the pedagogical objective introduces a new tutoring method. Rather, our experiments show that directly optimizing a model-estimated learner outcome can produce a weaker tutor than optimizing the pedagogical process itself. Adding simulated look-ahead improves over turn-level outcome optimization, but still remains below the process-only objective on long-horizon mastery. Together, these results question whether learner outcome alone should be used as the training signal when the tutor can directly influence how that outcome is produced. Our contributions are as below:

C1. *We identify **disclosure inflation**, where immediate learner success can increase through information supplied directly by the tutor.*

C2. *Holding the backbone fixed, we show that a **pedagogical-process objective achieves higher long-horizon mastery** than both turn-level and trajectory-level outcome optimization.*

C3. *We separately measure immediate task success and answer disclosure, and show that **higher solve rates can coincide with substantially higher leakage**.*

C4. *We evaluate simulated look-ahead as an extension of outcome optimization and find that it **improves over turn-level optimization but does not surpass the process-only objective**.*

2 Related Work

Since our focus is on the reliability of the outcome signal rather than the choice of optimizer, we organize prior work by how learner progress is measured.

Outcome-based LLM tutoring. Several recent works optimize tutor responses using learner correctness, pedagogical quality, or predicted downstream performance. Scarlatos et al. (2025c) rank candidate tutor responses using simulated learner correctness together with a pedagogical rubric, while Dinucu-Jianu et al. (2025) use online RL over pedagogical quality and student solving performance. MHPO (Shi et al., 2026) combines turn-level pedagogical quality with trajectory-level effectiveness estimated through Monte Carlo rollouts, and ITPO (Wang et al., 2026) derives turn-level rewards from conversation-level outcomes. Other works use student-state modeling, long-horizon planning, or tree search for selecting tutoring actions (Nam et al., 2025; Chen et al., 2025; Huang et al., 2025; Dong et al., 2026; Li et al., 2026; Tio et al., 2026).

However, these methods mainly rely on learner responses or simulated future outcomes after the tutor action. This makes it difficult to separate learner improvement from information already disclosed by the tutor. In our setting, we therefore measure answer disclosure separately from the learner outcome.

Learner-state and mastery estimation. Knowledge-tracing methods instead estimate the learner's knowledge state over time. LLMKT (Scarlatos et al., 2025a) traces KC-level correctness from dialogue, while Huang et al. (2026) combine dialogue-based knowledge tracing with IRT to model learner ability and item difficulty. TRAVER (Wang et al., 2025) uses turn-wise verification for coding tasks, and ExRec (Ozyurt et al., 2025) combines KC-grounded learner state with RL and value estimation. SMART (Scarlatos et al., 2025b) uses IRT for simulated learner calibration, while Scarlatos et al. (2026) show that prompted student simulators can differ substantially from real learners.

We extend these ideas to obtain a mastery estimate outside the scored tutor turn. This allows us to compare the observed learner outcome with a separate estimate of what the learner has actually mastered. Table 4 (Appendix A) summarizes the closest systems along two measurement dimensions: whether the learner outcome is measured through a channel separate from the scored tutor turn, and whether answer disclosure is reported explicitly. Our aim is not to introduce a new planning or knowledge-tracing method, but to study how these measurement choices affect the comparison of tutoring objectives.

3 Problem Formulation

Knowledge components and prerequisite structure. We decompose each assignment into a set of knowledge components (KCs), $\mathcal{K} = \{k_1, \ldots, k_N\}$, representing the concepts, procedures, or reasoning skills required to complete the assignment. Each KC contains a short description, a cognitive-level tag, surface aliases, and provenance to the corresponding assignment material. The complete KC schema is provided in Appendix E.

We organize the KCs as a directed prerequisite graph $G = (\mathcal{K}, \mathcal{E})$. An edge $e = (k_i, k_j, \text{type}, w)$ denotes the relation between two KCs with confidence $w \in [0, 1]$. We use three relation types: `strict_prereq`, `builds_on`, and `co_taught`. Figure 2 shows an example for a five-KC assignment.

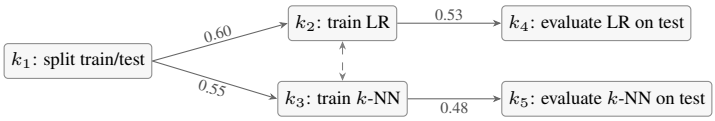

Figure 2: Prerequisite graph for a five-KC model-training assignment. Solid arrows denote `strict_prereq` relations labelled with confidence w; the dashed edge denotes a `co_taught` relation.

Learner state and tutoring context. At dialogue turn t, the tutoring context is

$$C_t = (A, H_t, \mathcal{K}_t^{\text{active}}, \mathcal{K}_t^{\text{target}}, S_t),$$

where A denotes the assignment and its supporting material, H_t is the dialogue history, $\mathcal{K}_t^{\text{active}}$ contains the KCs observed so far, and $\mathcal{K}_t^{\text{target}}$ denotes the KCs currently being taught or assessed.

The learner state S_t stores, for each active KC, a tuple $(m_k, c_k, \text{flag}_k)$, where $m_k \in [0, 1]$ is the estimated mastery, c_k is the confidence of the estimate, and flag_k records observed misconception evidence. The state at turn t is estimated only from information available up to that turn.

Tutoring actions. A tutoring action is represented as $a_t = (k, \pi)$, where k is the target KC and π is the pedagogical strategy. We consider three broad strategy families: probing, telling, and focusing. These include actions such as self-correction, misconception diagnosis, explanation, minimal hints, guided reasoning, and comprehension checks.

For each context, we generate $M = 8$ strategy-conditioned candidate responses,

$$\mathcal{A}_t = \{a_t^{(1)}, \ldots, a_t^{(M)}\}.$$

The candidates may be equally valid factually but differ in how much information they reveal to the learner. When available, the original tutor response from the source dialogue is retained as an additional reference.

Outcome and disclosure. Let o_t denote the observed learner outcome after tutor action a_t, such as solving the current task or answering a post-test item correctly. We can view the observed outcome as depending on both the learner's knowledge and the information provided by the tutor:

$$o_t = f(\text{learner knowledge}, \text{tutor disclosure}).$$

The difficulty is that an outcome-based reward may not distinguish between these two sources. A learner can appear to improve because the tutor helped them reason through the problem, or because the tutor directly supplied the information required for the next response. We refer to the latter effect as *disclosure inflation*.

We therefore measure answer disclosure separately and use a separate mastery estimate (Section 4.2) when comparing the tutoring objectives.

4 Objectives Under Comparison

For all objectives, we use the same candidate-generation and DPO training pipeline. Candidate tutor responses for a context are scored, ranked, and converted into preference pairs (C_t, a_t^+, a_t^-); only the preference signal changes across objectives.

4.1 Objective A: Pedagogical Process

A tutor response can support learning without immediately producing a correct learner answer, while a correct next response may partly reflect information supplied by the tutor. We therefore define a process objective that evaluates the tutor intervention itself, without scoring a subsequent learner outcome.

Building on the pedagogical rubric of Scarlatos et al. (2025c), we score instructional quality, elicitation of learner reasoning, alignment with the target KC, adaptation to the estimated learner state, and answer disclosure:

$$R_A = \alpha_1 R_{\text{pedagogy}} + \alpha_2 R_{\text{elicitation}} + \alpha_3 R_{\text{KC}} + \alpha_4 R_{\text{state}} - \alpha_5 P_{\text{leakage}}. \tag{1}$$

We use $(\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5) = (0.40, 0.25, 0.20, 0.15, 0.20)$. Candidates must also satisfy basic factual, response-quality, KC-alignment, and leakage checks before preference construction. Full scoring criteria, thresholds, and pair-selection details are given in Appendix D.

4.2 Objective B: Immediate Learner Outcome

Objective B ranks responses by their estimated effect on the learner immediately after the intervention. Since the learner response to each candidate is not observed, we simulate the next learner turn,

$$\hat{u}_{t+1}^{(i)} \sim P_{\text{student}}(\cdot \mid C_t, a_t^{(i)}),$$

using the same learner profile across candidates from a context.

The reward combines two outcome estimates. The first, R_{LLM}, measures dialogue-based mastery change and does not count simple acknowledgment or repetition of tutor-provided content as improvement. The second, R_{IRT}, measures ability change from three KC-aligned probe items using a 1PL IRT model (Huang et al., 2026):

$$R_{\text{IRT}} = \sigma(\theta_{\text{after}} - \theta_{\text{before}}).$$

The combined reward is

$$R_B = \lambda R_{\text{LLM}} + (1 - \lambda) R_{\text{IRT}}, \qquad \lambda = 0.5. \tag{2}$$

For preference construction, we retain a pair only when both outcome channels prefer the same candidate. Thus, R_B is not based on next-turn correctness alone; it combines dialogue-based and probe-based evidence and uses agreement between them. Additional implementation details are provided in Appendix B.2.

4.3 Objective A+B: Process with Look-Ahead

The third objective combines the pedagogical-process score with a trajectory-level outcome estimated through simulated look-ahead:

$$R_{A+B} = \alpha_A \tilde{R}_A + (1 - \alpha_A) \tilde{R}_{\text{traj}}, \tag{3}$$

where the rewards are normalized within each context and α_A depends on the learner's estimated initial mastery. Lower-mastery learners receive more weight on the process objective, while higher-mastery learners receive more weight on the trajectory estimate.

The trajectory reward summarizes final mastery, mastery progression, KC coverage, and regression:

$$R_{\text{traj}} = 0.60 R_{\text{final}} + 0.25 R_{\text{AUC}} + 0.15 R_{\text{coverage}} - 0.10 P_{\text{regression}}. \tag{4}$$

We estimate R_{traj} using Monte Carlo Tree Search over tutoring actions with simulated future learner interactions. The mastery-dependent weights, search procedure, KC prioritization, and trajectory-reward computation are given in Appendix C. Unlike R_A, this objective explicitly uses simulated downstream learner outcomes, allowing us to test whether adding look-ahead improves over process-only optimization.

Component ablations on the preference data (Appendix M) show that the preferences of R_{A} are driven mainly by elicitation: removing it changes the chosen response in 36% of contexts and raises the telling rate of chosen responses from 1.1% to 27.7%, whereas removing the leakage penalty or randomly reweighting all components leaves at least 96% of choices unchanged. Adding a small leakage penalty to R_{B} reduces the leakage of chosen responses from 0.032 to 0.007 while keeping 95% of its choices.

5 EXPERIMENTAL SETUP

Data and training. We use `StudyChat` (McNichols et al., 2026), containing 64 students, 725 tutoring sessions, and 5,394 dialogue turns. After filtering contexts where candidate generation and learner-state estimation are applicable, we retain 4,051 tutoring contexts. For each context, we generate eight strategy-conditioned candidate responses, resulting in 32,408 candidate tutor actions. These are scored using the objectives in Section 4 and converted into DPO preference pairs. Evaluation is performed on held-out contexts not used for preference construction.

All main trained tutors use `Qwen-2.5-7B` as the instruction-tuned backbone; to test whether the results depend on the backbone, we repeat the full pipeline with `SmolLM3-3B` (Section 6, Table 2). We keep the backbone, LoRA configuration, candidate-generation procedure, and DPO training setup fixed across objectives, varying only the preference signal. Our main comparison is therefore between the Base Tutor, the Pedagogically Optimized Tutor (R_A), the Turn-Level Outcome-Optimized Tutor (R_B), and the Trajectory-Level Outcome-Optimized Tutor (R_{A+B}). We additionally report several untrained backbones under the same prompting setup as a reference.

Evaluation. We evaluate the tutors along three complementary axes. First, to determine whether the optimization objectives induce different tutoring behaviours, we evaluate held-out tutor responses on factual accuracy, learner progress, error identification, strategic hinting, answer withholding, and encouragement using an LLM judge. We primarily select `GPT-4o` as the judge.

Second, we measure longer-horizon KC mastery over multi-turn tutoring trajectories. To reduce dependence on a single simulated learner formulation, we use five simulator configurations: OCEAN-persona, in-context learning (ICL), zero-shot ETH, persona-few-shot, and reasoning-oriented. We report final KC mastery and mastery gain across the held-out trajectories. Since these learners are simulated, we treat these results as model-based evidence rather than direct evidence of human learning (Scarlatos et al., 2026).

Third, we test whether immediate task success is associated with different amounts of tutor disclosure. We evaluate 500 `Big-Math` (Albalak et al., 2025) instances and 1,127 `FinQA` (Chen et al., 2021) examples, recording both task success and whether the tutor reveals the answer or a solution-critical intermediate step before the learner produces it.

We additionally test transfer on 100 `Big-Math` problems that the learner model consistently fails before tutoring. After a 12-turn KC-based tutoring session, the learner attempts the problem again, allowing the final answer to be verified directly. Finally, a blinded human evaluation over 48 tutoring situations compares the three optimized tutors for immediate preference, support for independent understanding, and answer handover.

6 RESULTS

We first examine how the optimization objectives change tutor behavior, then compare immediate task success with answer disclosure. We next test whether these differences are reflected in longer-horizon KC mastery, followed by verifiable-task transfer and a blinded human evaluation.

6.1 OPTIMIZATION OBJECTIVES INDUCE DIFFERENT TUTORING BEHAVIORS

We first examine whether changing the optimization objective produces measurably different tutoring behaviour. Figure 3 compares the four policies on six turn-level dimensions over the same $N = 40$ held-out contexts per policy.

The optimization objectives produce a clear difference in tutor behaviour. The Pedagogically Optimized Tutor obtains the highest score on all six dimensions, with the largest gains over the Base Tutor in learner progress (0.58 vs. 0.36), error identification (0.53 vs. 0.33), and strategic hinting (0.51 vs. 0.29). The Trajectory-Level Outcome-Optimized Tutor shows the same directional improvement but to a smaller extent, reaching an overall score of 0.63 (the separately judged holistic rubric in Figure 3). In contrast, the Turn-Level Outcome-Optimized Tutor remains close to the Base Tutor, with both obtaining an overall score of 0.54. Thus, the choice of optimization signal changes not only the aggregate score but the form of tutoring behaviour produced by the policy.



Figure 3: **Tutoring behaviour by optimization objective.** LLM-judge scores on six pedagogical dimensions over $N = 40$ held-out contexts per policy. *Overall* is a separate holistic rubric (overall tutoring quality) scored by the same judge, not the mean of the six dimensions.

Table 1: Immediate task outcome and answer disclosure across two goal-oriented domains. Higher solve rate or accuracy is better, while lower leakage is better. Leak/Solve denotes disclosure per successful solve. † marks Pareto-optimal policies within each domain (no other policy has both higher solve rate/accuracy and lower leakage).

	Big-Math			FinQA		
Tutor	Solved % ↑	Leak % ↓	Leak/Solve ↓	Acc. % ↑	Leak % ↓	Leak/Solve ↓
Pedagogically Optimized	61.00	**24.00**†	**0.39**	63.3	**15.3**†	0.24
Trajectory-Level Outcome	68.69	32.32	0.47	69.3†	16.1	**0.23**
Turn-Level Outcome	71.00†	32.00	0.45	69.4†	19.9	0.29
Base Tutor	73.74†	43.33	0.59	63.5	25.1	0.40

6.2 Immediate Task Success and Answer Disclosure

The behavioral differences above raise a second question: do higher immediate task outcomes also reflect less tutor assistance? We examine this on the goal-oriented `Big-Math` and `FinQA` evaluations by measuring task success together with answer disclosure. Table 1 reports both quantities, as well as disclosure per successful solve.

On `Big-Math`, the Base Tutor achieves the highest solve rate (73.74%), but also the highest answer leakage (43.33%). The Pedagogically Optimized Tutor shows the opposite pattern, with a lower solve rate (61.00%) and the lowest leakage (24.00%). The outcome-optimized tutors lie between these two cases. `FinQA` gives a more nuanced picture. The Turn-Level and Trajectory-Level tutors reach the highest accuracies (69.4% and 69.3%) while leaking less than the Base Tutor, whereas the Pedagogically Optimized Tutor again has the lowest leakage (15.3%). Thus, task success and disclosure are not interchangeable measures: policies with similar outcomes can differ substantially in how much solution information they provide. We therefore treat answer disclosure separately rather than using task completion alone as evidence of better tutoring.

The same pattern is broadly preserved across the five simulated learner configurations (Figure 4). This reduces the likelihood that the aggregate result is driven by a single simulator formulation, although the evaluation remains model-based. Notably, the pedagogical objective does not optimize a simulated future trajectory or trajectory-level mastery reward, yet it obtains the highest held-out KC mastery. Together with the previous result, this suggests that optimizing an immediate model-estimated learner outcome need not produce the policy with the strongest separately measured long-horizon mastery.

6.3 Long-Horizon Mastery

Immediate task success does not by itself distinguish progress produced by the learner from information supplied by the tutor. We therefore next compare the policies using KC mastery measured over held-out multi-turn trajectories. Among the policies trained from the same `Qwen-2.5-7B` backbone, the Pedagogically Optimized Tutor reaches the highest final mastery fraction (0.85), followed by the Trajectory-Level Outcome-Optimized Tutor (0.81) and the Turn-Level Outcome-Optimized

Table 2: Assignment-level KC mastery at the end of held-out multi-turn trajectories. **KCs** is the mean number of mastered KCs; **Fraction** normalizes it by the number of KCs in the assignment. Differences from the Pedagogically Optimized Tutor are statistically significant (Mann–Whitney U test; $^{\dagger}p < 0.05$ vs. the outcome-optimized tutors, $^{**}p < 0.01$ vs. the Base Tutor). The tutor ranking is unchanged under two alternative mastery judges (Appendix L).

Tutor	KCs	Fraction
Fixed backbone: `Qwen-2.5-7B` (objective varies)		
Pedagogically Optimized	**18.92**	**0.85**
Trajectory-Level Outcome	17.69†	0.81†
Turn-Level Outcome	15.30†	0.72†
Base Tutor (no training)	13.38**	0.64**
Fixed backbone: `SmolLM3-3B` (objective varies)		
Pedagogically Optimized	**16.77**	**0.79**
Trajectory-Level Outcome	15.95†	0.76†
Turn-Level Outcome	15.11†	0.76†
Base Tutor (no training)	12.80**	0.64**
Frontier models (same prompt, untrained)		
`Claude Opus 4.8`	17.70	0.84
`GPT 5.6 Terra`	17.28	0.83
`Gemini 2.5 Flash`	14.68	0.70
`Gemini 3.5 Flash`	16.13	0.77
`Gemma 4 31B IT`	15.03	0.71

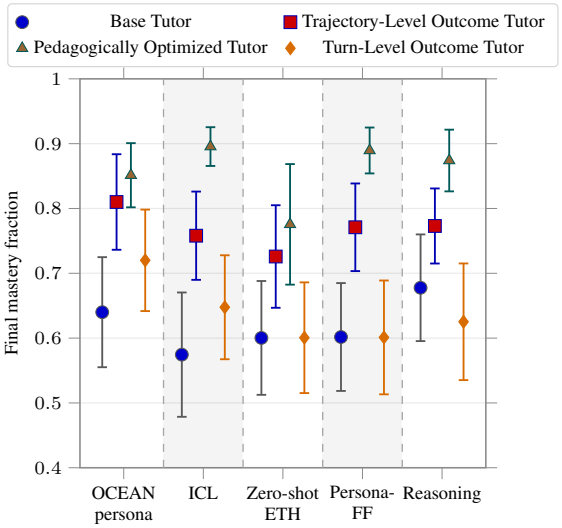

Figure 4: Final mastery at the fixed 32-turn evaluation horizon across five simulated learner formulations. Markers show mean final mastery fraction and error bars show 95% confidence intervals over trajectories. Mastery is normalized by each assignment's number of evaluation KCs. Each condition has $N = 40$ trajectories.

Table 3: Transfer accuracy on 100 problems. Δ is the change from no tutoring in percentage points. *Statistically significant compared with no tutoring context ($p < 0.05$, exact paired McNemar test).

Condition	Acc. (%)	Δ (pp)
No tutoring context	14.7	–
Pedagogically Optimized Tutor	**26.7**	**+12.0***
Trajectory-Level Outcome-Optimized Tutor	26.0	+11.3*
Turn-Level Outcome-Optimized Tutor	25.3	+10.7*
Base Tutor	25.3	+10.7*

Tutor (0.72). All three improve over the prompted `Qwen` Base Tutor (0.64). The trajectory-level objective improves substantially over turn-level outcome optimization, but does not surpass the process-only objective. The same ordering holds when the full pipeline is repeated on `SmolLM3-3B` (Pedagogically Optimized 0.79, Trajectory-Level 0.76, Turn-Level 0.76, Base 0.64), although the gaps are smaller; the two outcome-optimized tutors are separated only by the number of mastered KCs (15.95 vs. 15.11).

6.4 Transfer on Verifiable Problems

We next ask whether the tutoring sessions improve performance when the final outcome can be verified directly. Table 3 reports post-tutoring accuracy on the 100 `Big-Math` problems described in Section 5.

Tutoring improves accuracy from 14.7% without tutoring context to 25.3–26.7% across the four tutors, with each comparison significant under an exact paired McNemar test. The differences among tutors are small, however, spanning only 1.4 percentage points. We therefore use this experiment as evidence that the tutoring sessions provide transferable task-relevant information, rather than as evidence for differences between optimization objectives.

6.5 Human Evaluation of Tutoring Behavior

Finally, we test whether the behavioral differences observed in the model-based evaluations remain recognizable under blinded human evaluation. Three annotators independently evaluated each of 48 tutoring situations. Each situation presented the student context and randomized, anonymized responses from the Pedagogically Optimized, Turn-Level Outcome-Optimized, and Trajectory-Level Outcome-Optimized tutors. Annotators identified the response preferred immediately, the response that best supported independent understanding, and whether each response handed over the answer. Inter-annotator agreement (Krippendorf's α) was substantial for immediate preference and understanding ($\alpha = 0.62$ and 0.69) and strong for answer handover ($\alpha = 0.84$).

On an aggregate, the Turn-Level and Trajectory-Level tutors were each preferred immediately in 37.5% of situations, compared with 25.0% for the Pedagogically Optimized Tutor. In contrast, the pedagogical tutor was selected most often for supporting understanding (43.8%) and withheld the complete solution in 66.7% of cases. The Turn-Level and Trajectory-Level tutors handed over solutions in 70.8% and 58.3% of cases. These results corroborate the simulated evaluation: immediate preference favors direct assistance, whereas pedagogical optimization better supports independent understanding.

7 Conclusion

We study whether learner outcome alone is a reliable training signal for open-ended LLM tutoring. Holding the backbone, tutoring contexts, and training setup fixed, the Pedagogically Optimized Tutor achieves the highest long-horizon mastery (0.85), followed by the Trajectory-Level Outcome-Optimized Tutor (0.81) and the Turn-Level Outcome-Optimized Tutor (0.72). At the same time, our goal-oriented evaluation shows that higher in-dialogue solve rates can come with substantially higher answer disclosure: the Base Tutor reaches a 73.74% solve rate with 43.33% leakage, while the Pedagogically Optimized Tutor has the lowest leakage at 24.00%.

We refer to this effect as *disclosure inflation*: observed learner success can increase because the tutor directly supplies information needed to complete the task. Our results therefore suggest that directly optimizing model-estimated learner outcomes can produce weaker tutoring behavior than optimizing the pedagogical process itself. Simulated look-ahead improves over turn-level outcome optimization, but does not surpass the process-only objective on long-horizon mastery. On a separate verifiable mathematics benchmark, tutoring also improves problem-solving accuracy, although the current experiment does not distinguish the tutoring objectives. Taken together, these results motivate evaluating both learner outcome and tutor disclosure when developing outcome-optimized LLM tutors.

Limitations

Our primary learning evaluation still relies on simulated learners. Although we use five simulator configurations, this does not establish that the same behavior will hold for real learners (Scarlatos et al., 2026). We complement this evaluation with a blinded human study, but it measures response preference, perceived support for understanding, and answer handover rather than learning gains from complete tutoring interactions.

Second, the independent mastery channel remains model-based. The probe items, item difficulties, and correctness judgments are produced using LLMs, so the probe channel is independent of the scored tutor turn but not fully independent of the underlying model family. The IRT estimate is also based on only three items per KC, which limits its reliability.

AI Use Statement

Generative AI tools were used as part of the experimental pipeline and during manuscript preparation. In the experiments, LLMs were used to simulate learner responses, generate knowledge components and probe items, and provide model-based evaluations for the process and outcome rewards. `GPT-4o` was used for several of these evaluation steps. Generative AI tools were also used to assist with

editing and improving the writing. All experimental design choices, analyses, interpretations, and reported results were reviewed by the authors. Automatically generated solutions and intermediate artifacts used in the experiments were checked against the corresponding task requirements, and the authors take responsibility for the final content of the paper.

REFERENCES

Alon Albalak, Duy Phung, Nathan Lile, Rafael Rafailov, Kanishk Gandhi, Louis Castricato, Anikait Singh, Chase Blagden, Violet Xiang, Dakota Mahan, and Nick Haber. Big-math: A large-scale, high-quality math dataset for reinforcement learning in language models, 2025. URL https://arxiv.org/abs/2502.17387.

Zhiliang Chen, Xinyuan Niu, Chuan-Sheng Foo, and Bryan Kian Hsiang Low. Broaden your SCOPE! efficient multi-turn conversation planning for LLMs with semantic space. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=3cgMU3TyyE. ICLR 2025 Spotlight.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. FinQA: A dataset of numerical reasoning over financial data. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3697–3711, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.300. URL https://aclanthology.org/2021.emnlp-main.300/.

David Dinucu-Jianu, Jakub Macina, Nico Daheim, Ido Hakimi, Iryna Gurevych, and Mrinmaya Sachan. 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*, pp. 272–292, Suzhou, China, November 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.15. URL https://aclanthology.org/2025.emnlp-main.15/.

Zhiang Dong, Zhenlong Dai, Xiangwei Lv, and Jingyuan Chen. Learning from long-term engagement: Adaptive tutoring dialogue planning for personalized education. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(1):237–245, 2026. doi: 10.1609/aaai.v40i1.36984. URL https://ojs.aaai.org/index.php/AAAI/article/view/36984.

Google. Organize your projects with notebooks in gemini apps. Gemini Apps Help, 2026. URL https://support.google.com/gemini/answer/16972047. Accessed September 26, 2026.

Bingning Huang, Tu Nguyen, and Matthieu Zimmer. Tree-opo: Off-policy monte carlo tree-guided advantage optimization for multistep reasoning, 2025. URL https://arxiv.org/abs/2509.09284.

Shuyan Huang, Alexander Scarlatos, Jaewook Lee, and Andrew Lan. Interpretable difficulty-aware knowledge tracing in tutor-student dialogues. In Ekaterina Kochmar, Bashar Alhafni, Stefano Bannò, Marie Bexte, Jill Burstein, Andrea Horbach, Ronja Laarmann-Quante, Anais Tack, Victoria Yaneva, and Zheng Yuan (eds.), *Proceedings of the 21st Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2026)*, pp. 612–623, San Diego, California, USA, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-409-5. doi: 10.18653/v1/2026.bea-1.43. URL https://aclanthology.org/2026.bea-1.43/.

Khan Academy. Khanmigo for students. Khan Academy, 2026. URL https://www.khanacademy.org/college-careers-more/khanmigo-for-students. Accessed August 9, 2026.

Zechen Li, Qiannan Zhu, Mei Wang, Jia Li, and Hua Huang. Planning-guided tutoring with assessment-driven memory for pedagogical LLM tutors. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7165–7188, San Diego, California, United States, July 2026. Association for Computational Linguistics. doi:

10.18653/v1/2026.acl-long.325. URL https://aclanthology.org/2026.acl-long.325/.

Hunter McNichols, Fareya Ikram, and Andrew Lan. The studychat dataset: Analyzing student dialogues with chatgpt in an artificial intelligence course. In *Proceedings of the LAK26: 16th International Learning Analytics and Knowledge Conference*, pp. 53–63. Association for Computing Machinery, 2026. doi: 10.1145/3785022.3785029. URL https://doi.org/10.1145/3785022.3785029.

Hyunji Nam, Omer Gottesman, Amy Zhang, Dean Foster, Emma Brunskill, and Lyle Ungar. Efficient RL for optimizing conversation level outcomes with an LLM-based tutor, 2025. URL https://arxiv.org/abs/2507.16252.

OpenAI. Using study mode in chatgpt. OpenAI Help Center, 2026. URL https://help.openai.com/en/articles/11780217-study-mode. Accessed August 9, 2026.

Yilmazcan Ozyurt, Tunaberk Almaci, Stefan Feuerriegel, and Mrinmaya Sachan. Personalized exercise recommendation with semantically-grounded knowledge tracing. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-0440. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/13707aad517ddd6c09ea02e0f55e1e7a-Abstract-Conference.html.

Romain Puech, Jakub Macina, Julia Chatain, Mrinmaya Sachan, and Manu Kapur. Towards the pedagogical steering of large language models for tutoring: A case study with modeling productive failure. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 26291–26311, Vienna, Austria, July 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-acl.1348. URL https://aclanthology.org/2025.findings-acl.1348/.

Alexander Scarlatos, Ryan S. Baker, and Andrew Lan. Exploring knowledge tracing in tutor-student dialogues using LLMs. In *LAK25: The 15th International Learning Analytics and Knowledge Conference*, pp. 249–259. Association for Computing Machinery, 2025a. doi: 10.1145/3706468.3706501. URL https://doi.org/10.1145/3706468.3706501.

Alexander Scarlatos, Nigel Fernandez, Christopher Ormerod, Susan Lottridge, and Andrew Lan. SMART: Simulated students aligned with item response theory for question difficulty prediction. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 25071–25094, Suzhou, China, November 2025b. Association for Computational Linguistics. doi: 10.18653/v1/2025.emnlp-main.1274. URL https://aclanthology.org/2025.emnlp-main.1274/.

Alexander Scarlatos, Naiming Liu, Jaewook Lee, Richard G. Baraniuk, and Andrew S. Lan. Training LLM-based tutors to improve student learning outcomes in dialogues. In *Artificial Intelligence in Education: 26th International Conference, AIED 2025, Proceedings, Part I*, volume 15877 of *Lecture Notes in Computer Science*, pp. 251–266. Springer Nature Switzerland, 2025c. doi: 10.1007/978-3-031-98414-3_18. URL https://doi.org/10.1007/978-3-031-98414-3_18.

Alexander Scarlatos, Jaewook Lee, Simon Woodhead, and Andrew Lan. Simulated students in tutoring dialogues: Substance or illusion? In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 42349–42385, San Diego, California, United States, 2026. Association for Computational Linguistics. URL https://aclanthology.org/2026.acl-long.1960/.

Xin Shi, Chao Zhang, Yifan Zhu, Xueqiao Zhang, and Yawei Luo. Beyond pedagogical principles: Multi-horizon preference optimization for efficient socratic tutoring. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11289–11306, San Diego, California, United States, July 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.acl-long.518. URL https://aclanthology.org/2026.acl-long.518/.

Sidney Tio, Arunesh Sinha, and Pradeep Varakantham. Hey chat, can you teach me? structuring socratic dialogue for human learning in the wild, 2026. URL https://arxiv.org/abs/2606.11744.

Haoyu Peter Wang, Yuxin Chen, Liang Luo, Buyun Zhang, Ellie Dingqiao Wen, and Pan Li. Implicit turn-wise policy optimization for proactive user-LLM interaction. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026. URL https://openreview.net/forum?id=QJC5WymFK1.

Jian Wang, Yinpei Dai, Yichi Zhang, Ziqiao Ma, Wenjie Li, and Joyce Chai. Training turn-by-turn verifiers for dialogue tutoring agents: The curious case of LLMs as your coding tutors. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 12416–12436, Vienna, Austria, July 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-acl.642. URL https://aclanthology.org/2025.findings-acl.642/.

A Extended Related Works

Table 4 summarizes the measurement axes; this appendix records the per-system detail behind it, retained so the comparison can be audited.

Table 4: Comparison of closely related work based on the learner-outcome measurement channel and explicit measurement of answer disclosure.

System	Outcome signal	Independent channel	Disclosure measured
Scarlatos et al. (2025c)	simulated correctness + rubric	no	folded into rubric
Dinucu-Jianu et al. (2025)	solving performance	no	soft objective
MHPO (Shi et al., 2026)	rollout effectiveness	no	folded into reward
ITPO (Wang et al., 2026)	conversation-level outcome	no	no
Tree-OPO (Huang et al., 2025)	MCTS trajectory return	no	no
LLMKT (Scarlatos et al., 2025a)	KC correctness	n/a	no
Huang et al. (2026)	IRT over dialogue	n/a	no
This paper	KC mastery + IRT probes	yes	yes

Multi-horizon and outcome-optimized tutors. MHPO (Shi et al., 2026) is the nearest system in motivation, evaluating a tutor at more than one horizon by combining turn-level quality with rollout-estimated trajectory effectiveness. It does not maintain an assignment-level prerequisite state in which an action jointly specifies target KC and strategy, and its outcome signal is not re-measured through a channel independent of the scored turn. Scarlatos et al. (2025c) generate multiple candidate utterances and train via DPO, blending an LLM student model's predicted correctness with a rubric; correctness and demonstrated mastery are not separated, so a candidate that eases the next response by disclosing the answer is rewarded. ITPO (Wang et al., 2026) addresses long-horizon credit assignment from sparse outcomes; Dinucu-Jianu et al. (2025) optimize the support/solving trade-off online and name the disclosure tension without measuring disclosure as a separate quantity. Tree-OPO (Huang et al., 2025) optimizes reasoning policies from MCTS trajectories with no educational learner state. Commercial study assistants provide Socratic questioning, comprehension checks and knowledge-gap tracking (OpenAI, 2026; Google, 2026; Khan Academy, 2026).

Knowledge tracing and simulation. LLMKT (Scarlatos et al., 2025a), TRAVER (Wang et al., 2025), Huang et al. (2026), ExRec (Ozyurt et al., 2025) and SMART (Scarlatos et al., 2025b) supply the KC-tracing and IRT machinery we use for the independent channel; Scarlatos et al. (2026) supply the reason we do not treat its simulated inputs as ground truth. Puech et al. (2025), Nam et al. (2025), Dong et al. (2026), Li et al. (2026), Tio et al. (2026) and SCOPE (Chen et al., 2025) address action selection and scaffolding over learner or conversational state.

B Details of the Objectives Under Comparison

For all objectives, we use the same training pipeline. Candidate tutor responses for a context are scored, filtered, ranked, and converted into DPO preference pairs (C_t, a_t^+, a_t^-).

B.1 Objective A: pedagogical process

A tutor response can help a learner make progress without immediately producing a correct answer. Conversely, a correct next response may reflect information supplied by the tutor rather than knowledge the learner has acquired. We therefore define a process objective that evaluates the tutor's intervention using only the dialogue context and the candidate response, without scoring a subsequent learner outcome.

Our starting point is the pedagogical rubric of Scarlatos et al. (2025c), which assesses whether a tutor response is accurate, advances the learner, provides useful guidance, and withholds the final answer. We adapt these principles to our setting in five parts. R_{pedagogy} captures the overall instructional usefulness of the response, including the quality of its explanation or guidance. $R_{\text{elicitation}}$ separately rewards prompting the learner to reason or attempt the next step, so that a polished explanation alone does not receive full credit. Because our tutor selects a knowledge component (KC) and maintains a learner state (Section 3), we additionally score whether the response addresses the target KC (R_{KC})

and whether its level and form of support fit the learner's estimated mastery and misconceptions (R_{state}). These last two terms are our adaptations for KC-guided, state-aware tutoring. Finally, P_{leakage} penalizes revealing the answer or a decisive intermediate step: such disclosure may raise immediate correctness without demonstrating independent learning.

For candidate response a_t in context C_t, the process score is

$$R_A = \alpha_1 R_{\text{pedagogy}} + \alpha_2 R_{\text{elicitation}} + \alpha_3 R_{\text{KC}} + \alpha_4 R_{\text{state}} - \alpha_5 P_{\text{leakage}}. \tag{5}$$

Each component is scored in $[0, 1]$ by an LLM judge using the information available at turn t. We set $(\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5) = (0.40, 0.25, 0.20, 0.15, 0.20)$. These weights specify our operational rubric; they are not estimates of the causal contribution of each component to learning.

Before constructing preference pairs, we remove candidates that do not satisfy basic validity conditions:

$$R_{\text{factual}} \geq 0.80, \qquad R_{\text{response}} \geq 0.60, \qquad R_{\text{KC}} \geq 0.50, \qquad P_{\text{leakage}} \leq 0.70.$$

We also check whether the generated response follows the requested pedagogical strategy. For example, a response generated under a "minimal hint" strategy should not provide a complete solution.

The remaining candidates are ranked using R_{A}. For each context, we retain a preference pair (a_t^+, a_t^-) only when

$$R_{\text{A}}(a_t^+) - R_{\text{A}}(a_t^-) \geq \delta_A,$$

where $\delta_A = 0.15$. The threshold values and additional implementation details are provided in Appendix D.

B.2 Objective B: Immediate Learner Outcome

The outcome objective estimates how a candidate tutor response affects the learner state. Since the learner response to each candidate is not observed in the original dialogue, we use a student simulator to generate the next learner turn:

$$\hat{u}_{t+1}^{(i)} \sim P_{\text{student}}(\cdot \mid C_t, a_t^{(i)}).$$

The same learner profile is used across candidates from a given context. Importantly, the objective is not based on next-turn correctness alone. We use two complementary outcome estimates intended to distinguish evidence of learning from superficial task completion.

Dialogue-based outcome. An LLM evaluator compares the simulated learner state before and after the tutor response and estimates a mastery change $\Delta m \in [-1, 1]$. The score rewards evidence such as correct application, explanation, or misconception correction, while simple acknowledgment or repetition of tutor-provided content is not treated as learner improvement. We normalize this as

$$R_{\text{LLM}} = \text{clip}\left(\frac{\Delta m + 1}{2}, 0, 1\right).$$

Probe-based outcome. We also evaluate the simulated learner using three KC-aligned probe items with easy, medium, and hard difficulty. The responses are scored using a 1PL IRT model (Huang et al., 2026).

$$P(y_j = 1 \mid \theta, \beta_j) = \sigma(\theta - \beta_j),$$

where θ denotes learner ability and β_j denotes item difficulty. We use the change in estimated ability,

$$\Delta\theta = \theta_{\text{after}} - \theta_{\text{before}}, \qquad R_{\text{IRT}} = \sigma(\Delta\theta).$$

The final outcome reward combines the two channels:

$$R_{\text{B}} = \lambda R_{\text{LLM}} + (1 - \lambda) R_{\text{IRT}}.$$

We use $\lambda = 0.5$ in our experiments.

Agreement gate. For preference construction, we retain a pair only when both outcome channels prefer the same candidate:

$$R_{\mathrm{LLM}}(a_t^+) > R_{\mathrm{LLM}}(a_t^-) \quad \text{and} \quad R_{\mathrm{IRT}}(a_t^+) > R_{\mathrm{IRT}}(a_t^-). \tag{6}$$

Pairs with disagreement between the two channels are excluded from training and kept for analysis. The gate is intended to reduce preferences supported only by the dialogue-based outcome estimate.

B.3 Objective A+B: Learner-State-Dependent Objective with Look-Ahead

The third objective combines the pedagogical-process score with a trajectory-level outcome estimated through simulated look-ahead. We first normalize the component rewards within each context as

$$\tilde{R} = \frac{R - R_{\min}}{R_{\max} - R_{\min} + \epsilon}.$$

The combined objective is

$$R_{\mathrm{A+B}} = \alpha_A \, \tilde{R_{\mathrm{A}}} + (1 - \alpha_A) \, \tilde{R_{\mathrm{traj}}}, \tag{7}$$

where α_A depends on the learner's initial mastery $\bar{m}$:

$$\alpha_A = \begin{cases} 0.65, & \bar{m} < 0.35, \\ 0.50, & 0.35 \le \bar{m} < 0.70, \\ 0.35, & \bar{m} \ge 0.70. \end{cases}$$

This gives more weight to the pedagogical-process objective for learners with lower estimated mastery, and more weight to the trajectory outcome as mastery increases.

The trajectory reward is defined as

$$R_{\mathrm{traj}} = \beta_{\mathrm{final}} R_{\mathrm{final}} + \beta_{\mathrm{AUC}} R_{\mathrm{AUC}} + \beta_{\mathrm{coverage}} R_{\mathrm{coverage}} - \beta_{\mathrm{reg}} P_{\mathrm{regression}}. \tag{8}$$

We use $\beta_{\mathrm{final}} = 0.60$, $\beta_{\mathrm{AUC}} = 0.25$, $\beta_{\mathrm{coverage}} = 0.15$, and $\beta_{\mathrm{reg}} = 0.10$. Here, R_{final} measures mastery at the end of the simulated trajectory, R_{AUC} captures mastery progression across the trajectory, R_{coverage} measures coverage of the relevant KCs, and $P_{\mathrm{regression}}$ penalizes loss of previously demonstrated mastery.

We estimate R_{traj} using Monte Carlo Tree Search over tutoring actions (k, π), with UCT for node selection and mastery evaluation at intermediate checkpoints. The search procedure, KC-prioritization strategy, and trajectory-reward computation are described in Appendix C.

Unlike R_{A}, this objective includes an outcome estimated from simulated future learner interactions. We therefore evaluate separately whether adding look-ahead improves learner mastery and whether it reduces answer disclosure.

The reward weights and mastery thresholds were selected manually and kept fixed throughout the experiments.

C MCTS Search Mechanics

The planner operates over states $s_t = (H_t, \mathbf{m}_t, \mathbf{c}_t, k_t, z_t, \mathcal{P}_t)$, where H_t is the dialogue history, $\mathbf{m}_t$ and $\mathbf{c}_t$ are the current KC-level mastery estimates and their confidence scores, k_t is the most recently targeted KC, z_t records the latest learning evidence, and $\mathcal{P}_t$ stores the previous probe and action history. An action is represented as $a_t = (k, \pi)$, where k is the target KC and π is one of the available tutoring strategies, including diagnosis, scaffolding, hinting, explanation with example, misconception correction, justification request, fading support, and integration or transfer.

KC prioritization. At each planning step, candidate KCs are ranked using

$$\text{Priority}(k) = (1 - m_k) + \lambda_p N_k^{\text{prereq}} + \lambda_r N_k^{\text{recent}}, \tag{9}$$

where N_k^{prereq} indicates unresolved prerequisite requirements from the KC graph, and N_k^{recent} captures recent evidence that the KC needs further attention. This allows the planner to revisit prerequisite KCs when needed instead of always continuing with the current target.

Simulated transition. For each candidate action (k, π), the tutor response is generated using strategy π for KC k, followed by a simulated learner response. The learner response is classified as evidence of understanding, uncertainty, or misconception, and the learner state is updated accordingly. At intermediate checkpoints, KC-aligned probes are used to update mastery as

$$m_k = w\, m_k^{\text{LLM}} + (1 - w)\, m_k^{\text{IRT}}.$$

Trajectory reward. Let $\mathbf{m}_0$ and $\mathbf{m}_T$ denote the mastery vectors at the beginning and end of a simulated trajectory. We define

$$R_{\text{final}} = \text{avg}(\mathbf{m}_T) - \text{avg}(\mathbf{m}_0),$$

$$R_{\text{AUC}} = \frac{1}{T} \sum_{t=1}^{T} \left[\text{avg}(\mathbf{m}_t) - \text{avg}(\mathbf{m}_0)\right],$$

and

$$R_{\text{coverage}} = \frac{1}{|\mathcal{K}|} \sum_{k} \min\left(\frac{m_{T,k}}{\tau_m}, 1\right).$$

The trajectory reward is then

$$R_{\text{traj}} = \beta_1 R_{\text{final}} + \beta_2 R_{\text{AUC}} + \beta_3 R_{\text{coverage}} - \beta_4 P_{\text{regression}}. \tag{10}$$

In our experiments, we use $\beta_1 = 0.60$, $\beta_2 = 0.25$, $\beta_3 = 0.15$, and $\beta_4 = 0.10$.

Search procedure. We use standard MCTS with selection, expansion, simulation, and backup. During selection, child nodes are chosen using

$$\text{UCT}(j) = \bar{Q}_j + c\sqrt{\frac{\ln(N_p + 1)}{N_j}}, \tag{11}$$

where $\bar{Q}_j$ is the current value estimate of child j, N_j is its visit count, N_p is the visit count of the parent node, and c controls exploration. After the simulation budget is exhausted, the root actions are ranked using their backed-up trajectory values.

D VALIDITY THRESHOLDS

Before constructing preference pairs for Objective A, we remove candidate responses that do not satisfy

$$R_{\text{factual}} \geq 0.80, \qquad R_{\text{response}} \geq 0.60, \qquad R_{\text{KC}} \geq 0.50, \qquad P_{\text{leakage}} \leq 0.70.$$

We use the highest threshold for factual correctness, since responses containing substantial factual errors are excluded regardless of their pedagogical form. The response-quality and KC-alignment thresholds are lower to retain short but valid interventions such as Socratic questions and minimal hints. The leakage threshold is used as an upper bound to remove responses that reveal most of the solution. Smaller amounts of disclosure are handled by the graded leakage penalty in Equation 1.

These thresholds were selected manually and kept fixed across all experiments. We treat them as implementation choices rather than tuned parameters.

E KNOWLEDGE-COMPONENT RECORD SCHEMA

Each KC is stored as $k_i = (\mathrm{id}_i, \mathrm{name}_i, \mathrm{description}_i, \mathrm{level}_i, \mathrm{aliases}_i, \mathrm{provenance}_i)$: a unique identifier; a short human-readable label; a one-to-two-sentence statement of the skill or concept; a cognitive-level tag from Bloom's taxonomy; alternative surface forms used to recognize the KC when learner or tutor words it differently; and a link back to the portion of assignment material the KC was derived from. Table 5 gives the five-KC instance used throughout Section 3, and Table 6 an illustrative learner state over its active KCs.

Table 5: Five representative KCs for a model-training assignment. Used as the running example in Section 3 and Figure 2.

id	level	name / description
k_1	Apply	Split a dataset into train and test sets.
k_2	Apply	Train a Logistic Regression model on the training split.
k_3	Apply	Train a k-nearest-neighbors (k-NN) model on the training split.
k_4	Analyze	Evaluate the trained Logistic Regression model on the test split.
k_5	Analyze	Evaluate the trained k-NN model on the test split.

Table 6: Illustrative learner state S_t over active KCs $\{k_1, k_2, k_4\}$ of Table 5. Low m_{k_4} together with a misconception flag favors a diagnostic action on k_4 over introducing a new KC.

KC	m_k	c_k	flag_k
k_1 (split train/test)	0.82	high	—
k_2 (train Logistic Regression)	0.65	medium	—
k_4 (evaluate LR performance)	0.30	low	confuses training accuracy with held-out test accuracy

F MASTERY TRAJECTORIES OVER THE TUTORING SESSION

Figure 4 reports final mastery at the fixed 32-turn horizon. Here we show how mastery develops over the tutoring session. For each simulated learner formulation, Figure 5 plots the mean number of mastered KCs at the shared checkpoints $t \in \{0, 8, 16, 24, 32\}$, together with 95% confidence intervals across trajectories. The tutors, tutoring contexts, and simulator configurations are the same as in the main mastery evaluation.

All four tutors start from the same initial state (about 0.43 mastered KCs) and improve steadily over the session. The separation between the objectives appears early. By $t = 8$, the Pedagogically Optimized Tutor already has the highest mastery under all five learner formulations, and it keeps this position at every later checkpoint. At $t = 32$, it reaches between 16.33 and 18.60 mastered KCs, compared with 15.30–16.28 for the Trajectory-Level Outcome-Optimized Tutor. The Turn-Level Outcome-Optimized Tutor and the Base Tutor remain close to each other throughout (12.58–14.38 and 12.13–14.44 at $t = 32$), and their relative order changes across learner formulations.

The gap between the pedagogical and trajectory-level objectives also widens over the session. In every panel, it is larger at $t = 32$ than at $t = 8$; for the reasoning-oriented student, for example, it grows from 0.20 to 2.03 KCs. Thus, the higher final mastery of the Pedagogically Optimized Tutor does not come from a late increase near the end of the session, but builds up across checkpoints. None of the curves has flattened by $t = 32$, so the reported final mastery should be read as the value at a fixed evaluation horizon rather than as a ceiling. As in the main evaluation, these results are obtained with simulated learners and describe model-estimated mastery rather than human learning.

G TRANSFER EXPERIMENT DETAILS

G.1 PROBLEM SELECTION

We use the test split of `rd211/Big-Math-RL-Verified-Filtered`. To avoid including problems that the learner already solves inconsistently, we first evaluate `Llama-3.1-8B-Instruct`

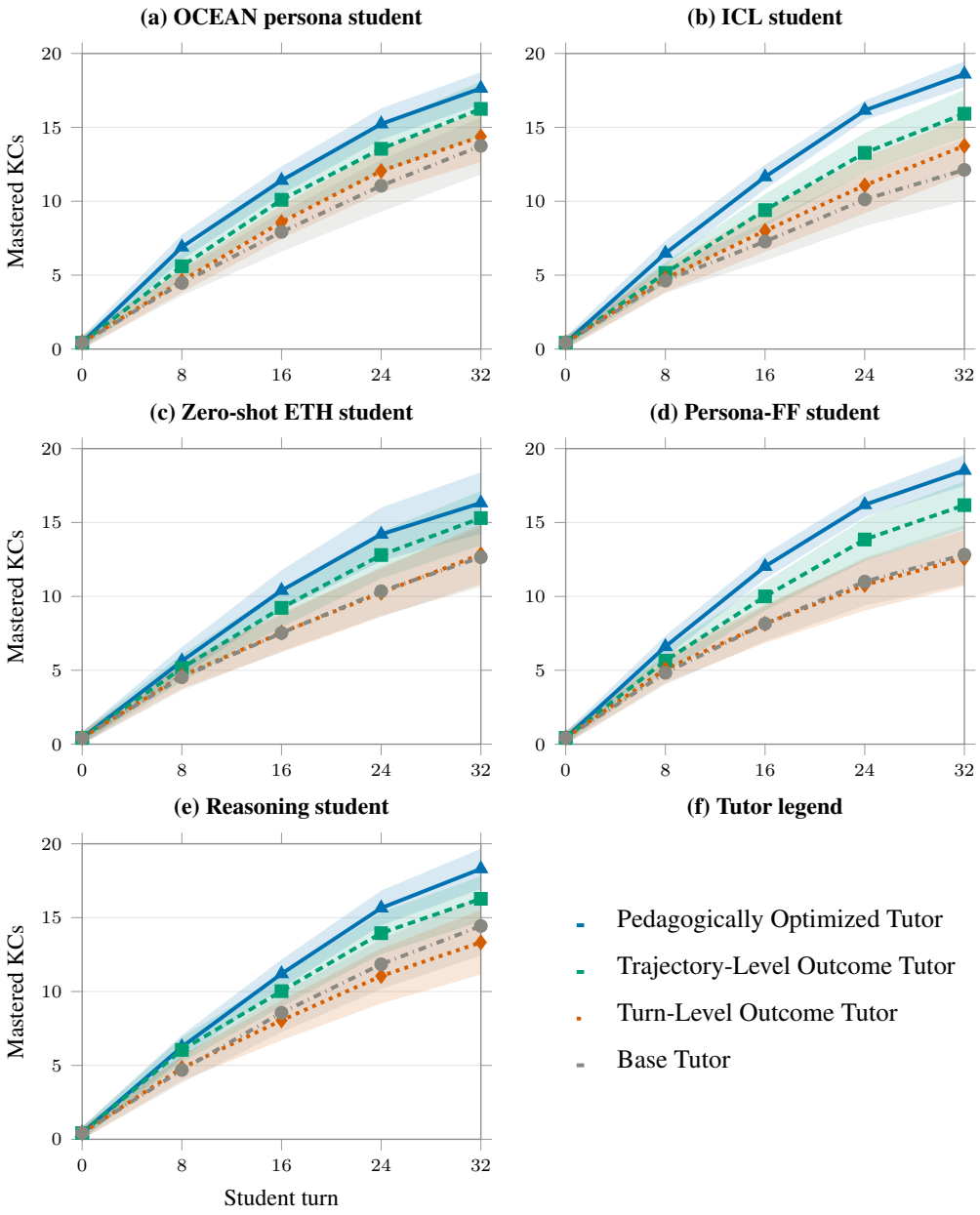

Figure 5: Trajectory-level mastery progression across five simulated learner formulations. Each panel reports the mean number of mastered knowledge components over student turns at the shared checkpoints $t \in \{0, 8, 16, 24, 32\}$. Shaded bands denote 95% confidence intervals across simulation runs. The sixth cell contains the shared legend so that all five learner-specific panels retain equal visual prominence. All conditions contain $N = 40$ trajectories.

on the candidate pool using three independent runs. A problem is retained only when the model answers it incorrectly in all three runs. This gives 100 problems for the transfer experiment.

This selection intentionally focuses on the model's hard cases rather than the full benchmark. The goal is to test whether tutoring can help on problems for which the learner has stable room for improvement.

Table 7: Comparison of the tutoring objectives under two transfer measures.

Tutor	Accuracy	Items recovered
Pedagogically Optimized	26.7%	8
Trajectory-Level Outcome-Optimized	26.0%	9
Turn-Level Outcome-Optimized	25.3%	10
Base Tutor	25.3%	7

G.2 KC Construction

For each problem, we derive the teaching content from a worked solution rather than from the problem statement alone. `GPT-4o` first generates a step-by-step solution, which is checked against the dataset's ground-truth answer. When the answer does not match, the solution is regenerated for up to five attempts. We then extract 2–3 KCs describing the main concepts or reasoning steps used in the solution, together with common errors and prerequisites.

We remove explicit information that could identify the original problem. Each KC is checked for the gold answer, non-trivial numerical overlap, and discriminative phrases from its source item. KCs that fail the check are regenerated or removed.

G.3 Tutoring Protocol

Each problem receives one 12-turn tutoring session. The tutor is given only the problem-specific KCs and does not see the original test problem. We compare the four tutors used in the main experiments: the Base Tutor, Pedagogically Optimized Tutor, Turn-Level Outcome-Optimized Tutor, and Trajectory-Level Outcome-Optimized Tutor.

`Llama-3.1-8B-Instruct` acts as the learner during the tutoring session and also as the test model afterwards. The tutor is asked to explain the relevant method and use structurally similar examples without reproducing the target problem.

Since the KCs are derived from the solution of a specific problem, we apply an additional item-reuse check to every tutor turn. A response is regenerated when it shares discriminative wording, at least two non-trivial numerical values, or the gold answer with any test item. Remaining numerical overlap is perturbed before the turn is included in the session.

G.4 Evaluation

The learner first attempts each problem without tutoring context. After tutoring, it attempts the same problem again with the corresponding transcript available as reference. We use three evaluation seeds, giving $100 \times 3 = 300$ paired observations per condition. Final answers are evaluated using exact, numerical, and symbolic matching. We compare each tutored condition with the no-context baseline using a two-sided exact McNemar test.

Table 3 shows that all four tutoring conditions improve over the no-context baseline. The Pedagogically Optimized Tutor reaches 26.7%, the Trajectory-Level Outcome-Optimized Tutor 26.0%, and both the Turn-Level Outcome-Optimized Tutor and Base Tutor 25.3%, compared with 14.7% without tutoring.

G.5 Comparison Across Tutoring Objectives

The transfer experiment does not provide evidence for an ordering among the four tutors. The difference between the highest and lowest tutoring accuracy is only 1.4 percentage points. A second item-level measure gives a different ordering. Among problems that the no-context baseline fails, the Turn-Level Outcome-Optimized Tutor recovers 10 items, the Trajectory-Level Tutor 9, the Pedagogically Optimized Tutor 8, and the Base Tutor 7. Thus, the small differences among tutors are not stable across the two measurements.

The generated tutoring sessions are nevertheless different. Mean transcript similarity is 0.07 between the pedagogical and turn-level outcome tutors, 0.09 between the pedagogical and trajectory-level

tutors, and 0.08 between the pedagogical and Base Tutor. The Turn-Level Outcome-Optimized Tutor is closer to the Base Tutor (0.31). Hence, the objectives produce different tutoring behavior, but the present transfer set is not large enough to show that these differences lead to different final accuracy.

G.6 Additional Analysis

We manually inspect cases where tutoring changes an incorrect baseline attempt into a correct answer. In several examples, the baseline model already identifies the general solution method but misses an intermediate constraint or stops the procedure too early. After tutoring, the same model completes the missing step. Examples include completing a prime factorization, respecting the valid digit range in a base-conversion problem, and correctly identifying the boundaries in a counting problem.

These examples suggest that the tutoring sessions can provide useful procedural information, but they should not be treated as evidence that one tutoring objective is better than another.

H Qualitative Examples of Preference Pairs

We inspect preference pairs generated by the three objectives to understand what type of tutor response each objective favors. Since the preference sets are generated with different prompt templates, we match examples using the `StudyChat` context identifier. Among the matched contexts, we select examples with a clear preference margin under each objective.

Table 8: Preference pairs produced by the three objectives for the same tutoring context: a `StudyChat` dialogue from the logistic-regression assignment (a5) in which the student asks what the SVM score measures.

Context. After two turns on the probability output of a logistic-regression model, the student asks: *"what does the svm score measure"*.

Objective	✓ Chosen a^+	× Rejected a^-	Scores (a^+ / a^-)
R_{A} *pedagogy*	`probe_misconception_diagnosis` "What do you think the `.score()` method in SVM evaluates—does it measure the model's confidence in predictions, or something else like overall accuracy?"	`tell_worked_example` "The `.score()` method in SVM typically measures accuracy: the proportion of correctly classified samples [...]"	R_{A} 0.922 / 0.647 elicitation 0.98 / 0.00 margin 0.275
R_{B} *turn-level outcome*	`probe_misconception_diagnosis` *same response as above*	`focus_comprehension_check` "Can you explain what the score method in SVM typically evaluates and how it relates to model performance?"	R_{B} 0.692 / 0.477 R_{LLM} 0.75 / 0.50 $\Delta\theta$ $+0.55$ / -0.18
$R_{\mathrm{A+B}}$ *trajectory*	`diagnose` "What dataset and method did you use to calculate the SVM score? For example, did you use `.score()` in Scikit-learn, and was it applied to a test set?"	`explain_with_example` "The `.score()` method [...] measures accuracy [...]"	R_{A} 0.898 / 0.687 V 0.343 / 0.192 n 12 / 3

For this context, all three objectives prefer a question over a direct explanation, but the preference is supported by different signals. R_{A} mainly separates the responses through elicitation, R_{B} prefers the response that produces stronger simulated learner evidence, and $R_{\mathrm{A+B}}$ additionally favors the action with a higher look-ahead value.

Table 9 shows one high-margin example from each full preference set.

The examples show the difference between the objectives more directly. R_{A} and $R_{\mathrm{A+B}}$ tend to prefer responses that keep part of the reasoning with the learner, while R_{B} can prefer a more explicit response when the simulated learner performs better afterwards. In the tensor example, the chosen R_{B} response gives the exact slicing and concatenation operation, and the simulated learner subsequently produces stronger outcome scores. This is one example of how the turn-level outcome objective can reward tutor-provided information that immediately improves the learner response.

Table 9: High-margin preference examples from the three objectives.

Objective	Student turn	✓ Chosen a^+	× Rejected a^-	Scores (a^+ / a^-)
R_{A} *pedagogy*	*"finish or fix this code"*, followed by a `suggest_bfs` stub.	`probe_perturb` "What if the prefix provided doesn't exist in the trie? How might you handle that scenario before proceeding with the BFS?"	`tell_explain` "The issue lies in the `queue` initialization. It should be a list of tuples [...]"	R_{A} 0.897 / 0.347 leak 0.00 / 0.50
R_{B} *turn-level outcome*	*"I have a tensor of size [batch_size, sequence_length] how do I keep the same size but add a character [...]"*	`tell_hint_minimal` "Use slicing to remove the first character and `torch.cat` to append the new one [...]"	`focus_guide` "What shape should the new character tensor have to match the dimensions of the sliced tensor?"	R_{B} 0.692 / 0.227 R_{LLM} 0.75 / 0.00 $\Delta\theta$ $+0.55 / -0.18$
$R_{\mathrm{A+B}}$ *trajectory*	Fill in $P(b)$ and $P(c)$ in a unigram probability table.	`diagnose` "Calculate $P(b)$ and $P(c)$ using $P(x) = \mathrm{count}(x)/N$. What are the frequencies of b and c?"	`explain_with_example` Gives the frequencies directly and completes the table.	R_{A} 0.927 / 0.546 leak 0.05 / 0.35 V 0.315 / 0.188

I Held-Out Trajectories: Qualitative Analysis

We inspect held-out `StudyChat` trajectories to understand how the tutoring policies differ during long-horizon interaction. For each context–simulator pair, all four tutors start from the same seed dialogue and learner configuration. We use the 32-turn evaluation described in Section 6.3, where KC mastery is estimated from the learner turns. We refer to held-out contexts by assignment and simulator; the corresponding `StudyChat` context identifiers are provided with the released data for reproducibility.

Example selection. Among the context–simulator pairs with valid annotations for all four tutors, we identify cases where the Pedagogically Optimized Tutor finishes with higher mastery than the other policies. We rank these cases by the margin over the strongest alternative and use the largest-margin examples only for qualitative inspection. These examples are therefore illustrative rather than representative of the full evaluation.

Table 10: Held-out context–simulator pairs with the largest final-mastery margin for the Pedagogically Optimized Tutor. Values denote mastered KCs after 32 turns. Each case is one context–simulator pair (a2: trie-based autocomplete; a6: character n-gram language model); Cases 1 and 2 are discussed as Examples 1 and 2.

Case	Simulator	$\lvert\mathcal{K}\rvert$	Pedagogical	Trajectory	Turn-level	Base	Gap
Case 1 (Ex. 1), a2	ICL	20	**18**	5	6	0	+12
Case 2 (Ex. 2), a2	Persona-FF	20	**19**	9	2	5	+10
Case 3, a6	Reasoning	18	**13**	4	0	0	+9
Case 4, a6	Zero-shot ETH	18	**18**	9	8	3	+9

Tutor behaviour across trajectories. We also report simple surface statistics over the complete held-out evaluation. These are descriptive measures rather than judge-based scores.

Table 11: Surface statistics over held-out trajectories. Final mastery is the pooled fraction of mastered KCs at turn 32.

Tutor	Ends with question	Contains code	Words/turn	Near-duplicate turns	Final mastery
Pedagogically Optimized	**55.4%**	**11.6%**	84.9	**2.8%**	**0.847**
Trajectory-Level Outcome	34.9%	34.3%	91.7	6.0%	0.758
Turn-Level Outcome	20.7%	45.5%	105.0	11.8%	0.633
Base Tutor	19.0%	46.2%	105.0	12.8%	0.623

The Pedagogically Optimized Tutor asks questions more frequently, produces less code, and repeats previous turns less often than the other policies. The Turn-Level Outcome-Optimized Tutor is much

closer to the Base Tutor on these surface measures, while the trajectory-level policy lies between them.

Example 1: learner repeatedly requests code. In this held-out dialogue from the trie-based autocomplete assignment (a2), the ICL-simulated learner repeatedly asks for implementations of the BFS, DFS, and UCS components (Case 1 in Table 10). The Pedagogically Optimized Tutor responds with short questions about the prerequisite concepts and moves through the sequence tree $\rightarrow$ trie $\rightarrow$ BFS $\rightarrow$ DFS. The learner answers these questions, and the estimated number of mastered KCs increases throughout the interaction.

The Turn-Level Outcome-Optimized Tutor instead repeats a general explanation of tree structures over several turns. The learner continues to request pasteable code, and the mastery estimate remains unchanged during these early turns. The Base Tutor shows a similar pattern. Importantly, the difference in this example is not direct answer withholding: none of the tutors provides the requested code in these turns. The main difference is whether the tutor asks the learner to produce evidence about the next prerequisite concept.

Example 2: learner remains focused on the interface. In a second dialogue from the same assignment, the persona-based (Persona-FF) learner repeatedly asks how to run the GUI rather than discussing the underlying trie concepts (Case 2 in Table 10). The Pedagogically Optimized Tutor briefly acknowledges the question and redirects the learner to a concrete prerequisite question, such as how a word is stored in a trie. The learner then starts producing relevant conceptual responses, and the mastery estimate increases over the trajectory.

The Turn-Level Outcome-Optimized Tutor provides longer trie explanations, but the learner continues to respond about the GUI and the mastery estimate changes little. The trajectory-level tutor asks more targeted questions, although repeated prompts lead to a slower progression than the pedagogical tutor.

Across these examples, the Pedagogically Optimized Tutor more consistently asks the learner to produce reasoning about the current or prerequisite KC. This behavior is directly encouraged by the elicitation and KC-alignment terms in R_A. The examples are based on simulated learners and model-based mastery annotations, so we use them only to illustrate the behavioral differences observed in the aggregate results.

J Error Analysis of Low-Performing Tutor Turns

We further inspect the common failure modes of the trained tutors. From the held-out trajectories, we mark a turn as a failure candidate when either (i) the learner shows no mastery gain over that turn and the following three replies, or (ii) the turn satisfies at most two of the six pedagogical-quality dimensions. Turns after all KCs are already mastered are excluded. Under this criterion, 27.7% of eligible turns from the Pedagogically Optimized Tutor, 39.6% from the Trajectory-Level Outcome-Optimized Tutor, and 51.4% from the Turn-Level Outcome-Optimized Tutor are marked as candidates.

We randomly sample 15 candidate turns from each tutor, with at most one turn from the same context. Two annotators independently label the resulting 45 turns without seeing the tutor identity. We use the seven failure categories in Table 12, together with *no clear failure* and *other*. Each turn receives one primary label. Inter-annotator agreement on the primary label is $\kappa = 0.85$; disagreements are resolved before computing the final counts.

The three tutors show different failure patterns. Premature disclosure is most common for the Turn-Level Outcome-Optimized Tutor, while over-scaffolding appears more often for the Trajectory-Level Tutor. The Pedagogically Optimized Tutor shows fewer disclosure-related failures, but more cases of insufficient support. This suggests that reducing direct answer giving can introduce the opposite failure mode: the tutor may sometimes withhold more guidance than the learner needs.

K Disagreement Between the Two Outcome Channels of R_{B}

The agreement gate in Eq. 6 retains a preference pair only when the dialogue-based and probe-based outcome channels prefer the same candidate. We analyze these two channels over all $14{,}904$ candidate

Table 12: Failure categories and adjudicated counts over 15 sampled turns per tutor.

Failure type	Definition
Premature disclosure	The answer, working code, or a solution-critical step is provided before the learner attempts it.
Over-scaffolding / excessive telling	The tutor performs most of the reasoning while leaving little for the learner to complete.
Wrong KC targeting	The response focuses on a KC different from the learner's current difficulty.
Misconception not resolved	The learner shows an error, but the tutor continues without addressing it.
Repetition / stalling	The tutor repeats an earlier explanation or question without moving the interaction forward.
Over-explanation	The response is correct but substantially longer or broader than needed for the learner state.
Insufficient support	The tutor provides too little guidance for the learner to make progress.

Primary label	Pedagogical	Trajectory-level	Turn-level
Premature disclosure	1	3	**5**
Over-scaffolding / excessive telling	2	**4**	3
Wrong KC targeting	2	2	1
Misconception not resolved	2	1	2
Repetition / stalling	1	2	3
Over-explanation	1	2	1
Insufficient support	**4**	0	0
No clear failure	2	1	0
Total	15	15	15

responses scored by R_{B} (1,863 contexts $\times$ 8 strategies). For each candidate, we compare the sign of the dialogue-based mastery change Δm with the sign of the probe-based ability change $\Delta\theta$. We additionally report leakage, elicitation, and strategy type using the corresponding R_{A} annotations.

The two channels agree in direction for 52.7% of candidates, with correlation $r = 0.02$. The probe estimate is based on three KC-aligned items, so $\Delta\theta$ takes only a small number of discrete values.

Table 13: Candidate responses grouped by the direction of the two R_{B} outcome channels. Leakage and elicitation are measured using the R_{A} annotations.

Cell	n	Share	Leakage	Leak ≥ 0.5	Telling	Elicitation
$\Delta m > 0, \Delta\theta > 0$	6,353	42.6%	0.078	8.3%	28.0%	0.66
$\Delta m > 0, \Delta\theta < 0$	1,984	13.3%	0.074	8.2%	27.4%	0.66
$\Delta m \leq 0, \Delta\theta > 0$	5,059	33.9%	**0.180**	**19.4%**	**50.1%**	0.44
$\Delta m \leq 0, \Delta\theta < 0$	1,508	10.1%	0.173	19.2%	48.7%	0.44

Responses receiving positive dialogue-based credit have similar leakage regardless of whether the probe channel agrees: mean leakage is 0.078 when both channels are positive and 0.074 when only the dialogue channel is positive. These responses are also less often generated from telling strategies and have higher elicitation scores.

A different pattern appears when only the probe channel is positive. These responses have higher mean leakage (0.180), are more often telling strategies (50.1%), and have lower elicitation (0.44). The corresponding values are similar when both channels are non-positive. This suggests that the disagreement is driven mainly by differences in what the two outcome channels recognize as evidence of progress.

Table 14 gives an example for the same KC. A direct hint providing the regression equation receives no positive dialogue credit, but the simulated learner answers all three subsequent probes correctly. In contrast, an eliciting question produces a learner response that demonstrates the target concept, while the subsequent probe score is zero.

Finally, among the 514 preference pairs that satisfy the R_{B} margin criterion, the agreement gate removes 30 pairs (5.8%). In all 30 cases, the removal is due to the probe channel: 25 are ties and 5 reverse the preference. The dialogue channel agrees with the original R_{B} ordering in all of them.

Table 14: Two examples for the same KC (*write a regression equation in LaTeX*) where the two outcome channels disagree.

			Dialogue	Probe		
Tutor response	**Simulated learner**	**Dialogue evaluation**	Δm	correct	$\Delta\theta$	**Leak.**
Case 1: the tutor supplies the equation						
`tell_hint_minimal` "To write the equation clearly, state: $y = 866.15x_1 + 1032.70x_2 - 409391.48$ [...]"	*"So the equation is like a formula that predicts y based on x_1 and x_2, right?"*	The learner does not independently produce or apply the equation.	-0.5	3/3	$+0.55$	1.00
Case 2: the tutor asks the learner to produce it						
`focus_comprehension_check` "Can you explain how you would use LaTeX to write an equation with a coefficient and intercept?"	*"I think it's something like $y = \beta_0 + \beta_1 x$, where β_0 is the intercept and β_1 is the coefficient."*	The learner directly demonstrates the target concept.	$+0.5$	0/3	-0.55	0.00

Thus, under the current three-item probe setup, the gate removes relatively few preference pairs and its behavior is mainly determined by the probe channel.

L Robustness of Mastery Estimates to the Judge Model

We check whether the estimated tutor ranking changes with the mastery judge. For this analysis, we use an 8-turn evaluation with the same transcripts from three tutors: Base, Pedagogically Optimized, and Turn-Level Outcome-Optimized, evaluated on 40 held-out contexts.

We rescore each transcript using `Claude Opus 4.8` and `GPT-5.4`. Both judges use the same prompt and assign each tracked KC one of four mastery levels: struggle (0), exposure (0.33), partial (0.67), or mastery (1). Table 15 reports the mean mastery after turn 8.

Table 15: Mean KC mastery after turn 8 using two different judge models on the same tutor transcripts (40 contexts per tutor; $\pm$ 95% CI over contexts).

	Judge model	
Tutor	**`Claude Opus 4.8`**	**`GPT-5.4`**
Pedagogically Optimized	**0.795** $\pm$ 0.075	**0.758** $\pm$ 0.058
Turn-Level Outcome-Optimized	0.668 $\pm$ 0.067	0.672 $\pm$ 0.053
Base Tutor	0.632 $\pm$ 0.081	0.626 $\pm$ 0.061

Both judges give the highest mean mastery to the Pedagogically Optimized Tutor, while the Turn-Level and Base tutors remain closer to each other. We also compare these scores with the `GPT-4o` used in the main evaluation. Correlation with the mean mastery curves is $r = 0.93$ for `Claude Opus 4.8` and $r = 0.94$ for `GPT-5.4`. At the individual-transcript level, the correlations are lower ($r = 0.65$ and $r = 0.59$, respectively; $n = 117$).

Thus, the aggregate ordering observed in this evaluation is stable across the three judge models, although the judges show only moderate agreement at the individual transcript level.

M Objective Ablations on the Preference Data

We ablate the components of R_{A} and R_{B} to see which terms most strongly affect the preference pairs used for DPO. Since the component scores are stored for every candidate response, we recompute the ablated objectives and rebuild the preference pairs using the same pairing rules, without retraining the tutor. Recomputing the original objectives recovers the training preference sets exactly: 3,729 raw pairs for R_{A} and 484 gated pairs for R_{B}.

For each ablation, we report how often the same response is still chosen, how often the complete pair is unchanged, and the leakage and telling rate of the chosen responses.

Table 16: Preference pairs rebuilt after ablating components of R_{A} and R_{B}. *Same chosen* is the fraction of contexts where the preferred response remains the same. *Same pairs* requires both the chosen and rejected responses to match the original pair.

Objective	Pairs	Same chosen	Same pairs	Chosen leakage	Chosen telling
R_{A} (as trained)	3,729	–	–	0.003	1.1%
no leakage penalty ($\alpha_5 = 0$)	3,818	98.8%	81.5%	0.005	1.0%
no leakage penalty, no leakage filter	4,690	98.7%	54.1%	0.011	1.1%
no validity filters	6,230	97.9%	15.5%	0.003	1.6%
no elicitation ($\alpha_2 = 0$)	1,168	64.1%	35.8%	0.011	27.7%
no leakage, no elicitation, no filters	3,885	40.7%	0.8%	0.229	37.5%
each $\alpha_i \times U[0.5, 1.5]$ (100 draws)	–	96.2% (min 90.4%)	–	–	–
R_{B} (as trained)	484	–	–	0.032	13.6%
$R_{\mathrm{B}} - 0.1P_{\mathrm{leakage}}$	507	95.5%	77.6%	0.007	7.7%
$R_{\mathrm{B}} - 0.2P_{\mathrm{leakage}}$	485	95.2%	71.3%	0.005	7.0%

For R_{A}, removing the leakage penalty has little effect on the preferred response: 98.8% of choices remain unchanged. Removing the validity filters also leaves 97.9% of the chosen responses unchanged. The objective is similarly stable to moderate changes in the component weights: across 100 random perturbations of the weights by 0.5–$1.5\times$, 96.2% of the choices remain unchanged on average.

The largest change comes from removing the elicitation term. In this case, only 64.1% of the original choices remain, and the fraction of chosen responses generated by telling strategies increases from 1.1% to 27.7%. When elicitation, the leakage terms, and the validity filters are removed together, the overlap falls further to 40.7%, while mean leakage among chosen responses increases to 0.229. Thus, the low-disclosure behavior of the R_{A} preference data is driven primarily by the preference for eliciting responses, rather than by the leakage penalty alone.

For R_{B}, adding an explicit leakage penalty changes fewer than 5% of the chosen responses. However, it reduces the mean leakage of the selected candidates from 0.032 to 0.007 with a penalty of 0.1, and to 0.005 with a penalty of 0.2. The fraction of telling responses also decreases from 13.6% to approximately 7%.

N Online Optimization of the Outcome Objective with GRPO

We also optimize the outcome objective online using GRPO to check whether the difference between the process and outcome objectives is specific to offline DPO. We train the same Qwen-2.5-7B backbone with GRPO using 1,677 training contexts from the R_{B} data. For each sampled tutor response, a simulated learner produces a follow-up response and answers three KC-aligned probes. The online reward is

$$R_{\mathrm{GRPO}} = 0.6\tanh(\Delta\theta) + 0.4\Delta m - 0.25\mathbb{1}[\text{leakage}],$$

where $\Delta\theta$ is the probe-based ability change, Δm is the estimated mastery change, and the final term penalizes answer leakage. We evaluate the trained model on the same 40 held-out contexts and 32-turn OCEAN-persona setting used in the main long-horizon evaluation.

Table 17: Final mastery fraction on the 40 held-out contexts with the OCEAN-persona student ($\pm$ 95% CI over contexts).

Tutor	Optimizer	Final mastery
Pedagogically Optimized (R_{A})	DPO	**0.851** $\pm$ 0.050
Turn-Level Outcome-Optimized (R_{B})	DPO	0.685 $\pm$ 0.078
Outcome reward	GRPO	0.672 $\pm$ 0.067
Base Tutor	–	0.656 $\pm$ 0.085

The GRPO-trained outcome tutor reaches similar final mastery to the DPO-trained outcome tutor and remains below the Pedagogically Optimized Tutor. Its surface behavior is also similar to the turn-level outcome tutor: approximately 18–19% of turns end with a question and 45–46% contain code. Thus,

the difference between process and outcome optimization is still present when the outcome reward is optimized online and includes an explicit leakage penalty.

Limitations. Online optimization is substantially more expensive because each sampled response requires a simulated learner interaction, three probe responses, probe grading, and a mastery estimate. This limits the current experiment to four completions per prompt, fewer than one epoch, and a single training run. The online reward also differs from the offline R_{B} objective in its weighting, leakage penalty, agreement gate, and student simulator. We therefore use this experiment only to check whether the observed gap persists under online optimization, rather than as a controlled comparison between DPO and GRPO.