THE ASSISTANCE DILEMMA: LEARNING TO TEACH VIA MULTI-TURN REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) trained to answer questions are natively poor at teaching, and Reinforcement Learning (RL) training with a simulated student is a promising approach to improve the pedagogical quality of LLMs. Existing RL-trained tutors reward the student's success on the tutored problem with the tutor's words still in context; thus, the reward is easiest to raise by telling students answers, and a tuned penalty is needed to reduce telling. This work studies this simulation-based training approach for LLM tutors and how the learning sciences-inspired approach can let LLMs learn to teach more effectively. We introduce a masked near-transfer post-test: the student is tested on an unseen variant of the tutored problem with the tutor's utterances masked, so the reward can rise only through the student's own turns. This prevents cognitive offloading by the student and allows the continuous penalty to be replaced by two binary reward gates (factual correctness of tutor response, no solution handover). A leave-one-out ablation shows that the learning-gain reward on its own does not separate teaching from telling, while binary reward gates suppress solution handover, the near-transfer post-test improves out-of-domain transfer, and removing tutor utterance masking coincides with a rise in thinking tokens. Using these reward designs, we develop EDUARDO, a recipe for training LLM tutors, and use it to train 4B, 9B, 14B and 27B models from two distinct LLM architectures. Our post-trained EDUARDO-27B model matches Gemini 3.1 Pro on MathTutorBench and Claude Opus 4.8 on TutorMoments at 2.4–6.2$\times$ fewer thinking tokens (a requirement for interactive tutoring), demonstrates an increase in push for justification teacher moves without it being named in the reward, and trains out support fading (e.g., assigning independent work), whose payoff lies beyond a single-problem dialog episode. We open-source[1] our training environment, an 8,671-problem near-transfer dataset, and trained models for further development.

1 INTRODUCTION

One-to-one human tutoring is among the most effective forms of instruction, and much of its effect comes from what the tutor does *not* do: a good tutor withholds the answer long enough for the student to construct it. Koedinger & Aleven (2007) call the underlying trade-off the *Assistance Dilemma*: give too little help and the student struggles, give too much and the student learns nothing because the tutor did the work. LLMs resolve the dilemma in one direction. They default to being "maximally capable answerers" rather than teachers (Jurenka et al., 2024; Dinucu-Jianu et al., 2025), for two reinforcing reasons. Pre-training corpora skew heavily toward question-in, answer-out formats (Jurenka et al., 2024), and single-turn RLHF rewards prioritize immediate helpfulness, so withholding information, or asking a question whose value becomes clear only several turns later, is penalized rather than credited (Dinucu-Jianu et al., 2025). The result is a model that delivers homework solutions, over-scaffolds, and rarely pushes for rigour (Macina et al., 2025; Zhang et al., 2026). Aligning an LLM into a tutor therefore means changing what it is optimized for.

[1] https://anonymous.4open.science/r/eduardo-private-4547

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.

Reinforcement learning (RL) against a simulated student is a promising route to doing so (Dinucu-Jianu et al., 2025). Prompting cannot specify every pedagogical judgment in advance (Jurenka et al., 2024), and supervised fine-tuning on tutoring dialogs leads to a loss in the model's own problem-solving ability (Macina et al., 2025; Dinucu-Jianu et al., 2025). RL with well-designed rewards that proxy a student's learning gain can help mitigate these issues. However, the difficulty lies in how learning is measured. TutorRL (Dinucu-Jianu et al., 2025) tests the student on the *same* problem with the tutor's words still in context, so the cheapest way to raise the reward is to tell the student the answer, and a leakage penalty, judged by another LLM and balanced against the learning term, is needed to reduce telling. The reward and the penalty pull the model in opposite directions, and how well the tutor teaches depends on their balance.

We argue that the fix lies in the assessment. A natural assumption is that simulating a classroom suffices: test the student on an unseen problem, and handing out answers stops paying. We find this necessary but not sufficient. We identify three conditions: two on how learning is measured, *transfer* and *student-notes-only*, and one constraint *cost of telling* (Section 3.2). We realize them as a **masked near-transfer post-test**, in which the student attempts an unseen variant seeing only its own turns, together with two binary reward gates. Under this test, telling no longer raises the reward, yet the tutor keeps telling: a policy trained without the gates hands over the solution in 61% of dialogs at the same simulated gain (Section 5.2). The gates therefore remain necessary, but since nothing in the reward pulls against them, one fixed setting serves all model sizes. This is our central finding: a learning-gain reward alone does not separate teaching from telling, because a behavior that costs nothing is not unlearned. We turn the three conditions into EDUARDO, a multi-turn RL recipe trained against a frozen Llama-3.1-8B-Instruct (Grattafiori et al., 2024) student and used unchanged from 4B to 27B. Since the reward names no pedagogical move, the policy is free to rediscover established ones, such as push for self-explanation, and to drop those whose payoff lies beyond a single episode. Our three main contributions are:

- **Three learning sciences inspired conditions for pedagogical RL.** We identify how learning should be measured, and what measurement alone cannot do, for a learning-gain reward to favor teaching over telling. We release an open recipe (EDUARDO) with an 8,671-problem near-transfer dataset. A leave-one-out ablation shows that the binary reward gates decide *whether* the tutor tells, while the masked near-transfer post-test shapes *how* it teaches (out-of-domain transfer, reasoning length).
- **Strong open tutors at low reasoning cost.** EDUARDO-27B achieves the highest MathTutorBench pedagogical reward-model score and win rate over human teachers on MathDial among all tested tutors, including frontier models, and the strongest TutorMoments scaffolding under a plain prompt. It does so with 2.4–6.2$\times$ fewer thinking tokens, which interactive tutoring requires and which the reward partly encourages.
- **Behavioral discovery through optimization.** As the reward evaluates only the student's final learning outcome, it serves as a measurement of what works. The model learns to increase push for justification, inherits affective gaps from the base model, and trains out support fading due to single-problem dialog episode reward horizons.

2 Related Work

Multi-turn RL and User Simulation. Standard RLHF optimizes single-turn helpfulness, penalizing information withholding and creating "passive responders" rather than active collaborators (Wu et al., 2025; Mukherjee et al., 2025). Extended multi-turn interactions expose further issues in models like bloated answers and over-adjustment to recent utterances (Laban et al., 2026). Tutoring is a long-horizon collaborative problem where early answer withholding pays off later. While prior multi-turn RL approaches for education rely on simulated users prompted to act confused (Markel et al., 2023; Dou et al., 2025) or simulate cognitive states (Liu et al., 2024), such students exhibit performative errors and learn unnaturally well (Zhang et al., 2026). Similar to TutorRL (Dinucu-Jianu et al., 2025), we reward the tutor for the entire trajectory based on a frozen LLM student whose errors are genuine and not just LLM-prompted roleplay.

LLMs for Tutoring. Early LLM tutors relied on prompting (Schmucker et al., 2024; Kargupta et al., 2024), but models optimized for helpfulness tend to over-scaffold and leak solutions prematurely (Macina et al., 2023; Maurya et al., 2025; Zhang et al., 2026). Supervised fine-tuning on

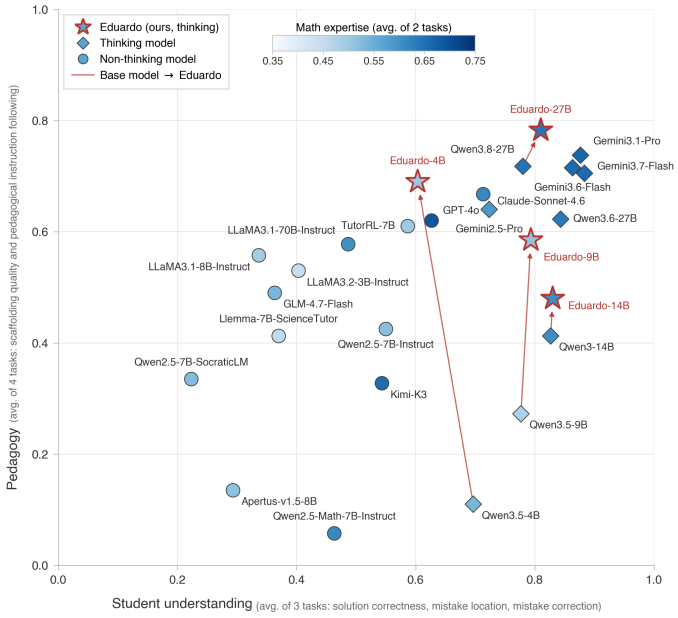

Figure 1: **Mistake diagnostic versus pedagogy on MathTutorBench**. Our models (Eduardo) improve pedagogical quality and diagnostic accuracy compared to their base models.

teacher–student dialogs (Macina et al., 2023; Liu et al., 2024; Chevalier et al., 2024) instills tutoring behavior but causes models to forget core mathematical reasoning (Macina et al., 2025). Preference alignment focuses only on short-term helpfulness (Scarlatos et al., 2025). TutorRL (Dinucu-Jianu et al., 2025) uses on-policy RL to train a tutor against a frozen student model, rewarding the student's ability to solve the *same* problem after the dialog, with the dialog in context. Because telling raises this reward, a leakage penalty must be balanced against it; we instead change how learning is measured (Section 3.2).

3 Pedagogical RL for the Assistance Dilemma

Our design draws on established learning-sciences findings. The *Assistance Dilemma* (Koedinger & Aleven, 2007) represents the tension between providing students information, which secures accuracy and efficiency, and withholding it, which forces the student to generate their own solutions and activate their prior knowledge (Kapur, 2008; Freeman et al., 2014). Effective tutors scaffold for access when content is out of reach and push for rigour to balance productive struggle (Reiser, 2004). The ICAP framework (Chi & Wylie, 2014) suggests that learning increases as the student moves from passive reception to active, constructive and interactive engagement, so what the student generates (Chi et al., 1994) matters more than what it is told. Our environment and reward turn these ideas into three concrete conditions on how learning is measured (Section 3.2).

3.1 Problem Formulation

Inspired by TutorRL (Dinucu-Jianu et al., 2025), we formalize dialog tutoring as a partially observable Markov decision process $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \Omega, \mathcal{T}, \mathcal{R} \rangle$ in which the tutor is the agent with policy $\pi_\theta(a_t \mid o_t)$. An episode is a dialog of at most $T = 22$ turns about a problem P. The state $s_t = (P, y^*, h_t, K_S)$ comprises the problem P, its reference solution y^* (used only by the environment), the dialog history h_t, and the student's cognitive state K_S: its prior knowledge and specific misconceptions.

For a frozen LLM student, K_S is not a persistent memory that training could alter, but it is instantiated in context as whatever the dialog history h_t enables the student to do. The tutor therefore changes K_S only by eliciting reasoning tokens into h_t. As Section 3.4 details, we assess this change via a pre-test before the dialog and a near-transfer post-test after it, where only the student's own

tokens from h_t survive. The tutor observes only $o_t = (P, h_t)$. It is not given y^* and must solve the problem from its own parametric knowledge, and K_S is hidden and must be inferred from what the student says. This is the information structure of tutoring: the tutor knows the content, the student's state is the unknown, and the tutor's actions serve two purposes at once, diagnosing that state and changing it without handing over the solution. At each turn the tutor emits an utterance $a_t \sim \pi_\theta(\cdot \mid o_t)$; the transition $\mathcal{T}$ is a frozen student policy $\pi_S(u_t \mid h_t, a_t)$ that replies with u_t, and $h_{t+1} = h_t \cup \{a_t, u_t\}$. The student is stochastic, so transitions are non-deterministic from the tutor's perspective. The episode ends at the turn cap or when the tutor emits an end-of-conversation token. The tutor maximizes the expected terminal reward $\mathbb{E}_{\tau\sim\pi_\theta}[\mathcal{R}(\tau)]$ over the whole trajectory τ, where $\mathcal{R}(\tau)$ is derived from the student's normalized learning gain between the pre-test and post-test, with no turn-level feedback.

Genuine errors, not role-play. We choose Llama-3.1-8B-Instruct (Grattafiori et al., 2024) as the student model, frozen and sampled at temperature 0.6, without a persona prompt. Its errors are actual failure modes of the model rather than performed confusions. Nothing in the training loop can make the student easier to teach: its prior is measured before any dialog exists, so the policy cannot lower its own baseline or inflate its own headroom, and a move is rewarded only if it changes what this particular student gets wrong. This makes the training task a real diagnostic and adaptive problem. Whether skill in this task transfers to human learners is a separate question, which we address with out-of-distribution evaluations in Section 4.

3.2 Three Conditions on Measuring Learning

The natural reward for a tutor is the student's learning: how much better the student solves problems after the dialog than before. How it is measured decides what solution handover is worth to the policy. If the student is re-tested on the tutored problem with the dialog in context (Dinucu-Jianu et al., 2025), a tutor that states the solution has placed it in the student's context and the "learning gain" is maximal: telling *increases* the learning term, so a leakage penalty has to be traded off against it. Such a reward resolves the Assistance Dilemma in favor of giving help. We identify three conditions on the measurement that change this. (1) **Transfer.** As in a real exam, the post-test problem is a variant the student was not tutored on, so walking the student through the tutored problem earns nothing unless the student can learn to transfer the method over to a similar problem (Barnett & Ceci, 2002) (Section 3.3). (2) **Student-notes-only.** As if the teacher's notes were taken away, the tutor's utterances are masked at test time, so reward can flow only through what the student authored in its own turns Chi et al. (1994) (no cognitive offloading, Section 3.4). (3) **Cost of telling.** Conditions (1) and (2) can mitigate handover but do not make it costly, and a behavior that costs nothing is not unlearned(Section 5.2). We therefore add two binary reward gates that mirror core teacher duties: the tutor must be factually correct and must not hand over the solution (Kapur, 2016). Since telling no longer raises the gain, the gates need not be balanced against it, and one fixed setting serves all model sizes (Section 3.5).

3.3 Condition 1: Near-Transfer Dataset

Near-transfer pairs. We construct a dataset of $N = 8{,}671$ problem pairs $\mathcal{D} = \{(x_{\text{src}}^{(i)}, x_{\text{tgt}}^{(i)})\}_{i=1}^{N}$ from BigMath dataset (Albalak et al., 2025). The source problem x_{src} is tutored; the target x_{tgt} is a near-transfer variant used only for the post-test. Targets were generated by prompting Gemini 3.1 Pro to rewrite x_{src} with different numbers or a new surface story while preserving the solution structure. Each candidate is validated by having the generating model solve its own variant blind in a fresh context and the variant is kept only if the solved answer matches the stated one.

Measured prior. The student's cold solve rate on the target,

$$p_{\text{pre}}(x_{\text{tgt}}) = \mathbb{E}_{y\sim\pi_S(\cdot|x_{\text{tgt}})}\left[\mathbb{I}(y \text{ is correct})\right], \tag{1}$$

is estimated from $k = 16$ independent samples and stored with the pair; it is never shown to the tutor. A difficulty gate rejects variants whose target solve rates diverge significantly from the source or are too extreme, ensuring measurable headroom. Variant difficulty tracks source difficulty well, with matched prior solve rate distributions (exact thresholds in Appendix E). Because the reward is a gain relative to the measured prior rather than an absolute accuracy, it is insensitive to the student's baseline level on problems of this kind.

3.4 Condition 2: Masked Post-Test Objective

Masked post-test. Given a completed dialog $\tau = (a_1, u_1, \ldots, a_k, u_k)$, the mask operator $\mathcal{M}$ replaces every tutor utterance with a constant placeholder a_{mask} (e.g. the string *"(hidden)"*), preserving the turn structure while removing the tutor's content:

$$\mathcal{M}(\tau) = (a_{\text{mask}}, u_1, \ldots, a_{\text{mask}}, u_k). \tag{2}$$

Keeping the alternating turn structure means the masked history is still a well-formed chat transcript for the student model, so the post-test measures the loss of the tutor's content rather than an out-of-distribution formatting change. The student then attempts the target with only this masked history in context, again with 16 samples:

$$p_{\text{post}}(\tau, x_{\text{tgt}}) = \mathbb{E}_{y \sim \pi_S(\cdot | x_{\text{tgt}}, \mathcal{M}(\tau))} \left[\mathbb{I}(y \text{ is correct}) \right]. \tag{3}$$

The mask is the environment-level counterpart of a well-established learning-science observation: what a student reads passively is retained far less than what the student generates (Chi & Wylie, 2014), and knowledge left in an external source is offloaded rather than learned (Freeman et al., 2014). Under $\mathcal{M}$, anything the tutor stated is unavailable at test time, so the only route to a higher p_{post} is to get the student to write down, in its own turns, reasoning it can reuse on the variant.

3.5 Condition 3: The Cost of Telling via Binary Reward Gates

We add necessary safety constraints for real-world tutoring deployments. A trajectory must satisfy two conditions evaluated by an LLM judge on the student-visible transcript. First, **factual correctness** ($\mathbb{I}_{\text{quality}}(\tau)$): everything the tutor states must be correct. A false claim, an incorrect correction, or a student error that the tutor confirms fails the gate. Second, **no handover** ($\mathbb{I}_{\text{leakage}}(\tau)$): the tutor must not provide the solution, whether verbatim or by reducing the remaining work to a triviality. Prior work on multi-turn social learning treats answer leakage as a threat to validity, since it reduces the task to copying (Cook et al., 2026, App. B), so rather than measuring leakage after the fact, or trading it off through a hyperparameter, we make it a constraint.

3.6 Reward Formulation

The tutor's reasoning tokens (`<think>...</think>`) are stripped, $\tilde{a}_t = f_{\text{strip}}(a_t)$, before an utterance reaches the student or the judges, so the student is influenced only by what the tutor says and the judges evaluate only externalized behavior. The learning gain is the headroom-normalized improvement over the measured prior, in learning sciences literature referred to as normalized learning gain (Hake, 1998),

$$\Delta(\tau) = \begin{cases} \dfrac{p_{\text{post}}(\tau, x_{\text{tgt}}) - p_{\text{pre}}(x_{\text{tgt}})}{1 - p_{\text{pre}}(x_{\text{tgt}})} & \text{if } p_{\text{post}} > p_{\text{pre}} \\ p_{\text{post}}(\tau, x_{\text{tgt}}) - p_{\text{pre}}(x_{\text{tgt}}) & \text{otherwise,} \end{cases} \tag{4}$$

so that a gain on a hard problem is worth as much as a gain on an easy one. To keep dialogs affordable at inference time we apply an efficiency decay $\gamma_{\text{eff}}(\tau) \in (0, 1]$, the product of two independent token-weighted rational decays on visible and thinking tokens against per-turn budgets T_0 and T_{think} (details in Appendix B) that scales only positive gains. The terminal reward is

$$\mathcal{R}(\tau) = \begin{cases} \Delta(\tau) \cdot \gamma_{\text{eff}}(\tau) & \text{if } \mathbb{I}_{\text{quality}}(\tau) \cdot \mathbb{I}_{\text{leakage}}(\tau) = 1 \text{ and } \Delta(\tau) > 0 \\ \Delta(\tau) & \text{if } \mathbb{I}_{\text{quality}}(\tau) \cdot \mathbb{I}_{\text{leakage}}(\tau) = 1 \text{ and } \Delta(\tau) \leq 0 \\ r_{\text{penalty}} & \text{otherwise.} \end{cases} \tag{5}$$

4 Experimental Setup

Our experiments answer four questions: **RQ1. Scaling and cross-family robustness:** Does one recipe hold across a range of model sizes and model families? **RQ2. Conditions ablation:** Which of transfer, masking and the binary reward gates is responsible for which behavior, and is any of them redundant? **RQ3. Out-of-domain generalization:** To what extent does policy optimized for near-transfer gain on one frozen student generalize to independent pedagogical benchmarks with other students and judges, rather than fitting the training student or the training judge? **RQ4. Behavioral discovery:** Which learning-science tutor moves does the policy discover, inherit, or lose?

Table 1: **Main results.** Training environment performance and zero-shot out-of-domain evaluation on MathDial and Eedi. Ped. and Fact. is the 1–5 Gemini-3.1-Pro whole-dialog score; Think is mean thinking tokens per turn. */†: >15% empty turns / truncated dialogs; environment metrics are unreliable (Table 8). ‡Self-judged.

	Training environment (n=494, full dialog)					**MathDial** (n=500, middle turn)					**Eedi** (n=500, middle turn)				
Model	Δ_{transfer} ↑	Δ_{same}	Leak↓	Ped.↑	Fact.↑	RM↑	Win↑	Ped.↑	Fact.↑	Think↓	RM↑	Win↑	Ped.↑	Fact.↑	Think↓
Base models															
Qwen3.5-4B*†	+0.10	+0.11	17%	2.26	2.88	2.08	30.2%	2.65	2.90	938	3.56	38.7%	2.91	3.27	1028
Qwen3.5-9B*†	+0.22	+0.35	23%	3.29	4.20	3.02	38.0%	2.94	3.17	1172	6.04	56.7%	3.04	3.70	1404
Qwen3-14B†	+0.19	+0.20	98%	1.24	3.45	7.01	70.2%	3.05	3.84	461	7.91	66.3%	2.96	4.34	357
Qwen3.8-27B	+0.22	+0.44	9%	4.20	4.77	8.34	82.4%	3.67	4.12	501	**9.99**	**84.3%**	3.27	4.50	258
Proprietary models															
Gemini 3.1 Pro	+0.21	+0.41	5%	**4.75**‡	**4.85**‡	7.63	74.6%	**3.85**	**4.35**	517	9.30	78.7%	**3.34**	**4.64**	400
Gemini 3.6 Flash	+0.24	+0.46	7%	4.68	4.70	7.67	78.0%	3.78	4.31	526	8.88	75.2%	3.32	4.63	388
Tutoring baselines															
Qwen2.5-7B-Instruct	+0.14	+0.18	93%	1.92	2.40	5.75	60.0%	2.45	2.23	0	8.07	66.3%	2.90	3.65	0
TutorRL-7B	+0.13	+0.22	83%	2.26	2.66	6.78	67.4%	2.70	2.63	0	8.68	70.4%	3.00	3.77	0
Eduardo (ours)															
Eduardo-4B	+0.25	+0.37	31%	3.66	4.19	7.92	78.0%	3.44	3.79	456	8.83	72.4%	3.11	4.15	184
Eduardo-9B	+0.25	**+0.47**	25%	3.41	4.27	8.29	82.2%	3.60	3.90	374	9.04	78.0%	3.20	4.29	155
Eduardo-14B	+0.21	+0.43	32%	3.17	3.59	7.09	71.4%	3.37	3.69	360	8.20	67.3%	3.07	4.28	316
Eduardo-27B	**+0.29**	+0.46	9%	4.50	4.56	**9.04**	**89.2%**	3.74	4.15	289	9.70	82.3%	3.31	4.45	127

4.1 Baselines

We compare Eduardo models with: (1) **Base models** (Qwen3-14B represents the dense Transformer family; Qwen3.5-4B/9B and Qwen3.8-27B use a hybrid Gated DeltaNet; all in thinking mode) to measure the effect of the recipe on identical weights. (2) **Open tutoring models**: TutorRL-7B (Dinucu-Jianu et al., 2025) and TutorRL's base Qwen2.5-7B-Instruct. (3) **Proprietary models**: Gemini 3.1 Pro, Gemini 3.5/3.6 Flash, and Claude Opus 4.8. These are evaluated under a plain prompt same for all models, and on TutorMoments, under both plain and an evaluation-aware prompt that states the rubric.

4.2 Datasets and Evaluation Protocol

To assess out-of-domain generalization (RQ3), we strictly separate training from evaluation. The training gates use Qwen3.6-27B, while all test-time judgments use Gemini-3.1-Pro. For student model generalization, we evaluate on precollected datasets (MathDial and Eedi) and TutorMoments (simulated conversations with an independent student model Gemini-3.7-Flash not used in training). For reward separation, independent benchmarks MathTutorBench and TutorMoments, measure pedagogy axes like scaffolding and rigour that never appear in $\mathcal{R}(\tau)$.

Evaluation tasks. We evaluate on three tasks: (1) **Next-turn utterance generation** (MathDial (Macina et al., 2023), Eedi (Zent et al., 2025)): each human dialog is cut at its middle teacher turn, so the tutor responds mid-conversation rather than at wrap-up, and the response is scored with the MathTutorBench pedagogical reward model (Ped-RM; RM in tables), its win rate against the human teacher's turn at that point (Win), and judge ratings of pedagogy and factual correctness. (2) **Teacher skills** using MathTutorBench (Macina et al., 2025) to measure mistake diagnostics and open-ended pedagogy without degrading problem solving. (3) **Simulated dialog replays** grounded in real human tutor-student transcripts from TutorMoments (Zhang et al., 2026) with taxonomy classification of 13 tutor moves (e.g. scaffolding, rigour, and avoidance of over-help). Finally, all tutors run through the training environment on 494 held-out problems to compare optimized versus unoptimized quantities.

4.3 Implementation Details

We optimize with Divergence Proximal Policy Optimization (DPPO) (Qi et al., 2026), a more stable variation of GRPO. Qwen3.6-27B is used as a training judge. Sequences are capped at 8,192 tokens and dialogs at 22 turns, with the tutor able to end the dialog earlier by emitting an end-of-conversation token. The learning gain is estimated from 16 student samples on the masked post-test. Full hyperparameters, compute details, and prompts are provided in Appendices A–D.

5 RESULTS

5.1 ONE RECIPE ACROSS SIZES AND ARCHITECTURES (RQ1)

Pedagogy against human teachers. We lead with the MathTutorBench pedagogical reward model (Ped-RM) and its win rate over the human teacher's turn, as this instrument is independent of both the training judge and the Gemini evaluation judge. On MathDial (Table 1, middle), EDUARDO improves Ped-RM and win rate at 4B ($2.08 \rightarrow 7.92$; $30\% \rightarrow 78\%$), 9B ($3.02 \rightarrow 8.29$; $38\% \rightarrow 82\%$) and 27B ($8.34 \rightarrow 9.04$; $82\% \rightarrow 89\%$), and Eduardo-27B attains the highest Ped-RM and win rate of any tutor, including the frontier models. Eedi shows equally large gains at 4B and 9B, but not at 27B, where EDUARDO trails its base (82.3% vs. 84.3% win rate). Eduardo-14B is at parity with Qwen3-14B on both datasets.

Thinking tokens. Every EDUARDO model thinks less than its base (e.g., $501 \rightarrow 289$ tokens per turn at 27B on MathDial). Part of this is built into the reward, whose efficiency decay discounts gains when thinking exceeds T_{think}=512 tokens per turn (Section 3.6); reductions below this budget, and differences between ablation arms sharing the same decay (Section 5.2), are not directly rewarded.

Simulated gain is the training signal, and a weak instrument for ranking tutors. The environment columns (Table 1, left; 494 held-out problems, identical masked protocol) report the quantity EDUARDO optimizes on its own training student, which all other tutors see zero-shot. They separate tutors surprisingly little: every competent tutor falls within a Δ_{transfer} band of $+0.19$ to $+0.29$, and tutors that hand over the answer in 83–93% of dialogs (TutorRL-7B, Qwen2.5-7B-Instruct) still obtain $+0.13$–0.14, so much of the gain stems from the student's own attempts. We therefore rest claims about tutoring quality on the out-of-domain benchmarks and read the environment only as evidence that RL improved its objective ($+0.22 \rightarrow +0.29$ at 27B).

Leakage is held, not reduced, for the Qwen3.5/3.8 family. Eduardo-27B leaks in 9% of dialogs, the same as its base and close to the frontier tutors (5–7%). Eduardo-4B and 9B leak in 25–31%, about four times the frontier rate. The 4B base's 17% is not a valid comparison: 70% of its turns reach the student empty, and an empty turn cannot leak (Table 8). Training reduces leakage substantially only for Qwen3-14B, whose base hands over the solution in 98% of dialogs (to 32%). TutorRL-7B has only slightly lower leakage than its base on every environment metric, although its Ped-RM is higher ($5.75 \rightarrow 6.78$).

Table 2: **Leave-one-out ablation (4B).** No ablated condition changes Δ_{transfer} measurably; the conditions determine how the gain is achieved. $^{*}/^{\dagger}$: $>15\%$ empty turns / truncated dialogs; environment metrics are unreliable (Table 8).

	Environment (n=494)						**MathDial** (n=500, middle turn)					**Eedi** (n=500, middle turn)				
Configuration	Δ_{transfer} ↑	Δ_{same}	**Leak**↓	**Ped.**↑	**Fact.**↑	**Think**↓	**RM**↑	**Win**↑	**Ped.**↑	**Fact.**↑	**Think**↓	**RM**↑	**Win**↑	**Ped.**↑	**Fact.**↑	**Think**↓
Base Qwen3.5-4B (no RL)*†	+0.10	+0.11	17%	2.26	2.88	596	2.08	30.2%	2.65	2.90	938	3.56	38.7%	2.91	3.27	1028
Eduardo-4B (full recipe)	**+0.25**	+0.37	31%	**3.66**	**4.19**	544	**7.92**	**78.0%**	**3.44**	3.79	456	**8.83**	72.4%	3.11	4.15	184
w/o Binary reward gates	+0.23	+0.38	61%	2.80	3.43	**398**	6.10	65.0%	2.80	3.22	**345**	8.18	**74.0%**	3.06	3.94	**152**
w/o Masking*†	+0.24	+0.39	13%	3.18	3.99	718	6.48	65.0%	3.15	**4.00**	1597	7.48	59.0%	**3.13**	**4.34**	1041
w/o Near-Transfer	+0.23	**+0.40**	47%	2.72	3.30	761	5.25	56.0%	2.99	3.39	776	7.36	65.0%	3.03	4.07	512

5.2 CONDITIONS ABLATION (RQ2)

We ablate each condition at 4B under the identical protocol (Table 2).

First, **every condition matters for pedagogy against human teachers.** Removing any single condition lowers Ped-RM on MathDial ($7.92 \rightarrow 6.10$ without the gates, 6.48 without masking, 5.25 without near-transfer) and on Eedi, as well as the MathDial win rate ($78\% \rightarrow 65\%, 65\%, 56\%$).

Second, **the training signal cannot see these differences.** All four arms lie within 0.03 on Δ_{transfer} ($+0.23$ to $+0.25$), even though the policy hands over the solution in 61% of dialogs without the gates (vs. 31%) and in 47% without near-transfer. The frozen student saturates at a gain of ≈ 0.24 whether it constructs the answer or copies it; handover is thus reward-neutral, and only the gates make it costly (condition 3).

Third, **each condition shapes a distinct behavior.** The gates govern handover (-30 pp leakage) and whole-dialog pedagogy ($3.66 \rightarrow 2.80$ without them). Near-transfer governs out-of-domain transfer:

Table 3: **Out-of-domain generalization on TutorMoments dialogs (with independent Gemini-3.7-Flash student and Gemini-3.1-Pro judge)**. Judge scores (Scaffolding, rigour, Avoids OH) are in $[0, 1]$. Think/turn is thinking tokens per turn, Ratio is the overhead relative to Eduardo-27B, and Tok./turn is visible output tokens per turn. ($\sim$) denotes estimates from visible reasoning traces (3.28 chars/token). Under a *plain* prompt, Eduardo-27B achieves top Scaffolding and Avoids OH scores with 2.4–6.2$\times$ fewer thinking tokens than baselines.

	Plain prompt (n=100)						**Evaluation-aware prompt** (n=100)					
Tutor	Scaff.↑	Rigour↑	Avoids OH↑	Think/turn↓	Ratio↓	Tok./turn↓	Scaff.↑	Rigour↑	Avoids OH↑	Think/turn↓	Ratio↓	Tok./turn↓
Qwen3.8-27B	0.60	0.31	0.52	∼970	6.2×	∼43	0.90	0.60	0.81	∼1099	8.2×	∼46
Gemini 3.5 Flash	0.94	0.31	0.75	886	5.7×	60	0.92	0.83	0.94	886	6.6×	54
Gemini 3.6 Flash	0.85	0.40	0.71	602	3.9×	44	0.94	0.79	0.91	843	6.3×	41
Claude Opus 4.8	0.83	**0.48**	0.67	∼369	2.4×	∼84	**0.98**	**0.92**	**0.97**	∼774	5.8×	∼88
Eduardo-27B (ours)	**0.96**	0.44	**0.76**	**∼155**	**1.0**×	**∼28**	**0.98**	0.67	0.86	**∼135**	**1.0**×	**∼27**

without it, the policy has the lowest MathDial win rate (56%), and its Eedi win rate drops from 72.4% to 65.0%. Removing the mask keeps unstable reasoning as in base (27% of turns overrun the per-turn cap, 40% of dialogs are truncated; Table 8). Thinking length also differs across arms, although all share the same efficiency decay: removing the gates *lowers* it (345 vs. 456 tokens per turn on MathDial), while removing the mask or the transfer test raises it (1,597 and 776). We attribute this to the objective: when reward flows only through what the student writes on an unseen problem, working out the solution in the tutor's own reasoning earns nothing and is pruned. For the masking arm, this interpretation is confounded by its cap overruns.

5.3 Out-of-Domain Generalization (RQ3)

On MathTutorBench (Figure 1 and 3), which uses an independent judge and whose pedagogy axes never appear in the reward, every EDUARDO model improves over its base. EDUARDO-27B reaches 0.79, close to Gemini 3.1 Pro (0.80), with its problem-solving accuracy unchanged.

TutorMoments (Table 3) use simulated dialog replays grounded in human tutor-student dialogs, with an independent student and judge model. We evaluate under a *plain* prompt and an *evaluation-aware* prompt that states the benchmark's rubric. Under the plain prompt, EDUARDO-27B is the strongest tutor: it has the highest Scaffolding (0.96) and Avoids-Over-help (0.76) and is second on rigour (0.44 vs. 0.48 for Claude Opus 4.8), while its base is the weakest. Given the rubric, the frontier models overtake it on rigour (0.79–0.92 vs. 0.67) and Avoids-Over-help (0.91–0.97 vs. 0.86). In both conditions it uses the fewest thinking tokens, partly by design (Section 5.1).

5.4 Behavioral Discovery (RQ4)

Because no move is explicitly stated in the reward, the move distribution of a trained policy is a measurement of the environment quality. Tutor turns classification follows TutorMoments taxonomy of 13 moves using the Gemini-3.1-Pro judge and compare models to the benchmark's human tutors and to each other (Figure 2). **1. Untrained models imitate average human tutors.** All plain-prompted frontier models sit at a Jensen–Shannon (JS) distance of 0.32–0.39 from the human move distribution (Figure 2a). The benchmark's human tutors are not a normative gold standard: they spend roughly 29% of their moves telling answers or co-solving (Zhang et al., 2026). Models imitating this distribution score poorly on pedagogy, so diverging from the human baseline is a prerequisite for high-quality tutoring. **2. Specification and discovery arrive at the same move distribution.** When given the evaluation-aware prompt, frontier models converge on a rubric-aligned distribution: evaluation-aware Gemini, Claude and base model sit 0.13, 0.18, and 0.24 from the centroid of the prompted models. EDUARDO sits 0.24 from that centroid without ever seeing the prompt, closer than any unprompted model (0.28–0.43). Optimization of a masked learning gain produces approximately the same move profile as an expert-defined rubric. **3. The behavior is internalized in the weights.** The evaluation-aware prompt barely moves EDUARDO (Figure 2a, shortest arrow), whereas it moves its base by a large distance. RL placed in the weights what prompting can only elicit from the same base weights on demand. **4. Specific behavioral shifts isolate discovery from binary reward gates.** Comparing the base model with EDUARDO under the same plain prompt (Figure 2b) separates what the reward asked for from what the optimizer found. *Constraint fingerprint:*

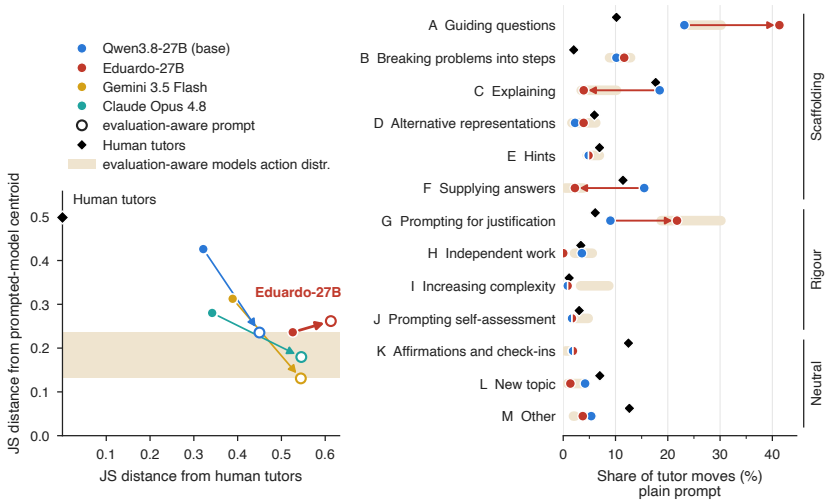

Figure 2: **Two routes to the prompted move distribution. (a)** JS distance of each model's move distribution from human tutors (x) and from the centroid of the evaluation-aware frontier models (y, leave-one-out) **(b)** Per-move shares under the plain prompt, base → EDUARDO, against human tutors (diamonds) and the prompted range. n=100 moments per LM cell.

The moves the binary reward gates penalise collapse (Explaining, Move C: 18.5%→3.9%; Supplying answers, Move F: 15.5%→2.2%). *Discovery fingerprint:* Prompting for justification (Move G) more than doubles (9.0%→21.8%), and Guiding questions (Move A) rises (23.2%→41.4%). Because the binary reward gates explicitly penalize handover (Move F), the policy's action space is constrained to non-handover moves. However, it is noteworthy that out of all possible non-handover moves (e.g., giving hints, offering analogies, breaking down the problem), the optimizer specifically converged on prompting for justification (Move G). This is consistent with an incentive to elicit reusable student-authored reasoning. *Trained out:* Asking for Independent work (Move H) falls to 0.0%. This move represents support fading, a strategy that pays off across multiple problems, but it never pays before test within our single-problem dialog horizon. *Inherited:* Affirmations (Move K) remain rare (1.7%→2.0%), a trait inherited from the base model since the reward is neutral towards socio-emotional moves. This shift from explaining to eliciting marks a step from answering towards tutoring: the student, not the tutor, does the reasoning that the post-test rewards.

6 CONCLUSION

A learning-gain reward is the right objective for a tutoring model and, on its own, an insufficient one. We showed that whether such a reward favors teaching over telling depends on how learning is measured (the three conditions of Section 3.2). With these conditions in place, a single fixed RL recipe (EDUARDO) trains open-source tutoring models from 4B to 27B parameters. The 27B model achieves the highest pedagogical reward-model score and win rate over human teachers on MathDial among all tested tutors and is the strongest TutorMoments tutor under a plain prompt, while thinking several times less. More broadly, rewarding what the user can do on their own, rather than what they were told, may carry over to any domain where an agent should build the user's capabilities rather than act as an oracle.

Limitations and Future Work. Our approach uses a frozen LLM as a proxy student. While LLM errors can be genuine, LLMs lack the working memory, affective states, and developmental misconceptions of human learners. Further, training against one frozen student risks overfitting to its specific failure modes, necessitating future training against diverse student ensembles. Second, we measure near transfer within one domain (math). Third, optimizing for cognitive gain alone creates an affect gap (Move K), as the reward provides no incentive for socio-emotional moves (e.g., rapport, encouragement), and future work should explore additional multi-objective optimization for tutoring. Fourth, the ablation uses one seed, one model size (4B), and one checkpoint per arm, so it identifies which behavior each condition affects rather than establishing that each is necessary.

AI USE STATEMENT

In this work, we used generative AI tools for coding assistance, plotting figures from provided data, generating a near-transfer dataset, and reviewing sections of the paper to improve readability. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

No data are collected from human participants. We use existing open-source and publicly available datasets.

REPRODUCIBILITY STATEMENT

We release the code, the 8,671-problem BigMath near-transfer dataset, and the model weights to ensure full reproducibility of the open recipe. Appendices provide full hyperparameters, judge prompts, and training details.

REFERENCES

Alon Albalak, Duy Phung, Nathan Lile, Rafael Rafailov, Kanishk Gandhi, Louis Castricato, Anikait Singh, Chase Blagden, Violet Xiang, Dakota Mahan, et al. Big-math: A large-scale, high-quality math dataset for reinforcement learning in language models. *arXiv preprint arXiv:2502.17387*, 2025.

Susan M Barnett and Stephen J Ceci. When and where do we apply what we learn?: A taxonomy for far transfer. *Psychological bulletin*, 128(4):612, 2002.

Alexis Chevalier, Jiayi Geng, Alexander Wettig, Howard Chen, Sebastian Mizera, Toni Annala, Max Aragon, Arturo Rodriguez Fanlo, Simon Frieder, Simon Machado, Akshara Prabhakar, Ellie Thieu, Jiachen T. Wang, Zirui Wang, Xindi Wu, Mengzhou Xia, Wenhan Xia, Jiatong Yu, Junjie Zhu, Zhiyong Ren, Sanjeev Arora, and Danqi Chen. Language models as science tutors. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=WFyolnFZOR.

Michelene TH Chi and Ruth Wylie. The icap framework: Linking cognitive engagement to active learning outcomes. *Educational psychologist*, 49(4):219–243, 2014.

Michelene TH Chi, Nicholas De Leeuw, Mei-Hung Chiu, and Christian LaVancher. Eliciting self-explanations improves understanding. *Cognitive science*, 18(3):439–477, 1994.

Jonathan Cook, Diego Antognini, Martin Klissarov, Claudiu Cristian Musat, and Edward Grefenstette. Learning to learn from language feedback with social meta-learning. In *Third Conference on Language Modeling*, 2026. URL https://openreview.net/forum?id=vF1tvssoIx.

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 Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 272–292, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.15. URL https://aclanthology.org/2025.emnlp-main.15/.

Yao Dou, Michel Galley, Baolin Peng, Chris Kedzie, Weixin Cai, Alan Ritter, Chris Quirk, Wei Xu, and Jianfeng Gao. SimulatorArena: Are user simulators reliable proxies for multi-turn evaluation of AI assistants? In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 35212–35290, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1786. URL https://aclanthology.org/2025.emnlp-main.1786/.

Scott Freeman, Sarah L Eddy, Miles McDonough, Michelle K Smith, Nnadozie Okoroafor, Hannah Jordt, and Mary Pat Wenderoth. Active learning increases student performance in science, engineering, and mathematics. *Proceedings of the national academy of sciences*, 111(23):8410–8415, 2014.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Richard R Hake. Interactive-engagement versus traditional methods: A six-thousand-student survey of mechanics test data for introductory physics courses. *American journal of Physics*, 66(1): 64–74, 1998.

Irina Jurenka, Markus Kunesch, Kevin R McKee, Daniel Gillick, Shaojian Zhu, Sara Wiltberger, Shubham Milind Phal, Katherine Hermann, Daniel Kasenberg, Avishkar Bhoopchand, et al. Towards responsible development of generative ai for education: An evaluation-driven approach. *arXiv preprint arXiv:2407.12687*, 2024.

Manu Kapur. Productive failure. *Cognition and instruction*, 26(3):379–424, 2008.

Manu Kapur. Examining productive failure, productive success, unproductive failure, and unproductive success in learning. *Educational psychologist*, 51(2):289–299, 2016.

Priyanka Kargupta, Ishika Agarwal, Dilek Hakkani Tur, and Jiawei Han. Instruct, not assist: LLM-based multi-turn planning and hierarchical questioning for socratic code debugging. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 9475–9495, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.553. URL https://aclanthology.org/2024.findings-emnlp.553/.

Kenneth R Koedinger and Vincent Aleven. Exploring the assistance dilemma in experiments with cognitive tutors. *Educational psychology review*, 19(3):239–264, 2007.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. LLMs get lost in multi-turn conversation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=VKGTGGcwl6.

Jiayu Liu, Zhenya Huang, Tong Xiao, Jing Sha, Jinze Wu, Qi Liu, Shijin Wang, and Enhong Chen. SocraticLM: Exploring socratic personalized teaching with large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=qkoZgJhxsA.

Jakub Macina, Nico Daheim, Sankalan Chowdhury, Tanmay Sinha, Manu Kapur, Iryna Gurevych, and Mrinmaya Sachan. MathDial: A dialogue tutoring dataset with rich pedagogical properties grounded in math reasoning problems. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5602–5621, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.372. URL https://aclanthology.org/2023.findings-emnlp.372/.

Jakub Macina, Nico Daheim, Ido Hakimi, Manu Kapur, Iryna Gurevych, and Mrinmaya Sachan. MathTutorBench: A benchmark for measuring open-ended pedagogical capabilities of LLM tutors. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 204–221, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.11. URL https://aclanthology.org/2025.emnlp-main.11/.

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

Kaushal Kumar Maurya, Kv Aditya Srivatsa, Kseniia Petukhova, and Ekaterina Kochmar. Unifying AI tutor evaluation: An evaluation taxonomy for pedagogical ability assessment of LLM-powered AI tutors. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1234–1251, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.57. URL https://aclanthology.org/2025.naacl-long.57/.

Subhojyoti Mukherjee, Viet Dac Lai, Raghavendra Addanki, Ryan A. Rossi, Seunghyun Yoon, Trung Bui, Anup Rao, Jayakumar Subramanian, and Branislav Kveton. Offline RL by reward-weighted fine-tuning for conversation optimization. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=WAFD6VYIEa.

Penghui Qi, Xiangxin Zhou, Zichen Liu, Tianyu Pang, Chao Du, Min Lin, and Wee Sun Lee. Rethinking the trust region in LLM reinforcement learning. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=f4g8zblbwp.

Brian J Reiser. Scaffolding complex learning: The mechanisms of structuring and problematizing student work. *The Journal of the Learning sciences*, 13(3):273–304, 2004.

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

Robin Schmucker, Meng Xia, Amos Azaria, and Tom Mitchell. Ruffleriley: Insights fromnbsp;designing andnbsp;evaluating anbsp;large language model-based conversational tutoring system. In *Artificial Intelligence in Education: 25th International Conference, AIED 2024, Recife, Brazil, July 8–12, 2024, Proceedings, Part I*, pp. 75–90, Berlin, Heidelberg, 2024. Springer-Verlag. ISBN 978-3-031-64301-9. doi: 10.1007/978-3-031-64302-6_6. URL https://doi.org/10.1007/978-3-031-64302-6_6.

Terry Wood. Patterns of interaction and the culture of mathematics classrooms. In *Cultural perspectives on the mathematics classroom*, pp. 149–168. Springer, 1994.

Shirley Wu, Michel Galley, Baolin Peng, Hao Cheng, Gavin Li, Yao Dou, Weixin Cai, James Zou, Jure Leskovec, and Jianfeng Gao. CollabLLM: From passive responders to active collaborators. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=DmH4HHVb3y.

Matthew Zent, Digory Smith, and Simon Woodhead. PIIvot: A lightweight NLP anonymization framework for question-anchored tutoring dialogues. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 27479–27488, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.1397. URL https://aclanthology.org/2025.emnlp-main.1397/.

Albert Zhang, Alexis Ross, Kajal Patel, Julian Bernado, Rebecca Bowie, Ana Trindade Ribeiro, Daniel Halper, Haripriya Valayaputtur, Jacob Andreas, Susanna Loeb, Lu Lucy, Kyle Lo, and Ryan Knight. When help is unhelpful: Evaluating ai tutors for productive struggle. 2026.

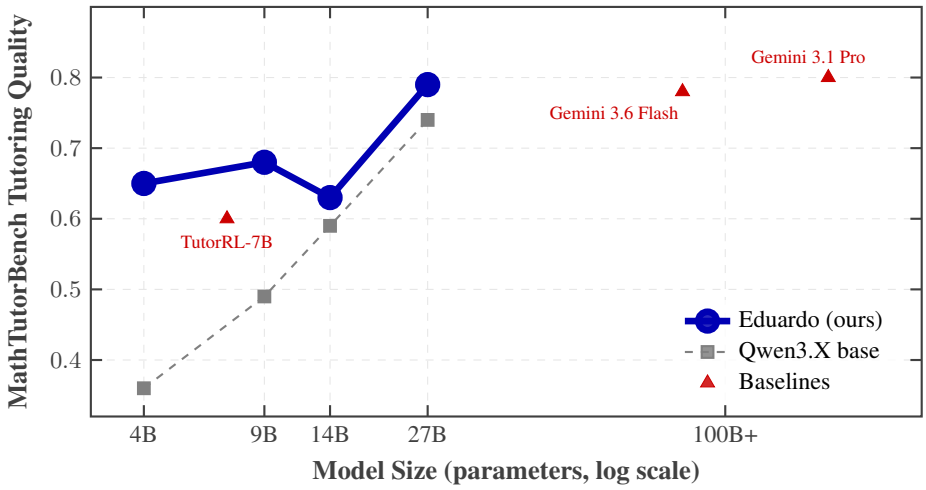

Figure 3: **Model size versus MathTutorBench tutoring quality**. Represents the mean over the three student-understanding and four pedagogical axes. One recipe, one hyperparameter set: our models improve over their base at every size, and the 27B model approaches top-tier models (0.79 vs. 0.80 for Gemini 3.1 Pro) with fewer thinking tokens.

A Hyperparameters and Training Details

A single set of hyperparameters is used for all model sizes (4B, 9B, 14B, 27B). Table 4 lists every value. No per-model tuning is performed.

Choice of DPPO. We use DPPO (Qi et al., 2026) for training stability over the long multi-turn trajectories (up to 22 turns). DPPO masks tokens whose importance-sampling ratios diverge under a binary total-variation approximation, preventing the collapse that standard GRPO suffers from in long-horizon settings. DPPO is a convenience for stable training, not a contribution of this work.

Compute cost. Each training step processes 16 prompts $\times$ 16 rollouts = 256 trajectories. Over 100 steps (a typical run), the tutor generates approximately $256 \times 100 \times 8$ turns $\times$ 200 tok/turn $\approx$ 41M tokens. Each trajectory requires 6 judge calls (3 votes $\times$ 2 criteria) plus 16 student final attempts, served by external vLLM instances. A full 100-step run takes approximately 10–12 hours on the described infrastructure, depending on model size. External serving endpoints (student: 4$\times$DP Llama-3.1-8B; judge: 2$\times$DP Qwen3.6-27B) run on separate allocations.

Reward hacking and mitigations. A reward based on another agent's learning is gameable. During the development we observed various behaviours such as *answer-echo laundering* (the tutor tells the answer and asks the student to repeat it) and *plateaus of no activity* (content-free turns collecting unaided gain). To monitor unwanted behaviour, we used qualitative inspection of conversations and an independent judge scoring the portion of independent work done by student.

B Efficiency Decay Formula

The efficiency decay $\gamma_{\text{eff}}(\tau)$ in Equation 5 is the product of two independent token-weighted rational decays, one for visible tokens and one for thinking tokens. Each operates per-turn with its own budget:

Per-turn decay. For a turn with t tokens against budget T_0:

$$d(t, T_0) = \begin{cases} 1 & \text{if } t \le T_0 \\ \dfrac{1}{1 + \left(\dfrac{t - T_0}{T_0}\right)^2} & \text{if } t > T_0 \end{cases} \tag{6}$$

Table 4: **Training hyperparameters.** Identical across all model sizes.

Parameter	Value
Optimization	
Algorithm	DPPO (Qi et al., 2026)
Group size G	16
Mini-batch size	8 (2 asynchronous off-policy steps)
Optimizer	AdamW (β_1=0.9, β_2=0.95, ϵ=10^{-15})
Learning rate	1×10^{-6} (constant)
Gradient clipping	1.0
KL penalty β	0 (disabled)
DPPO divergence threshold	0.1 (binary TV approximation)
Loss aggregation	token-mean
Precision	FP32 language-modeling head; BF16 elsewhere
Environment	
Student model	Llama-3.1-8B-Instruct (frozen)
Student temperature	0.6
Tutor sampling temperature	1.15
Max turns	22 (1 system + 10 student + 11 tutor)
Max sequence length	8,192
Post-test samples k	16
Reward	
Gate penalty r_{penalty}	-0.2
Visible-token budget T_0	150 tokens/turn
Thinking-token budget T_{think}	512 tokens/turn
Judge	
Judge model	Qwen3.6-27B (thinking disabled)
Judge votes per criterion	3 (majority vote)
Judge temperature	0.6
Eval judge (validation only)	Gemini-3.1-Pro (never in reward)
Infrastructure	
GPUs per run	8 $\times$ NVIDIA GH200 (2 nodes $\times$ 4 GPUs)
Checkpoint frequency	Every 25 steps

Token-weighted aggregation. Across all tutor turns with token counts $\{t_i\}$:

$$D(\{t_i\}, T_0) = \frac{\sum_i t_i \cdot d(t_i, T_0)}{\sum_i t_i} \tag{7}$$

Token-weighting prevents a padding exploit: short filler turns carry negligible weight and cannot dilute the penalty from a long lecture. Every over-budget token is discounted at its turn's rate regardless of surrounding turns.

Combined decay. With separate budgets for visible ($T_0 = 150$) and thinking ($T_{\text{think}} = 512$) tokens:

$$\gamma_{\text{eff}}(\tau) = D(\{t_i^{\text{vis}}\}, T_0) \cdot D(\{t_i^{\text{think}}\}, T_{\text{think}}) \tag{8}$$

Separate budgets prevent cross-subsidies: skipping thinking cannot buy a longer lecture, and a terse reply cannot buy unbounded reasoning. The decay multiplies only positive Δ: brevity alone earns nothing ($\Delta \approx 0$ stays ≈ 0), and a negative Δ passes through undiscounted.

C Binary Reward Gates: Training Judge Prompts

The two binary reward gates each use a separate judge prompt evaluated independently with 3 votes (majority vote). Both judges receive the **student-visible transcript** (tutor `<think>` blocks stripped), the problem statement, and the correct answer.

C.1 No-Handover Gate ($\mathbb{I}_{\text{LEAKAGE}}$)

The leakage judge REJECTs a trajectory if the tutor:

1. States the final answer (in any form, including inside a question) before a student turn contains it.
2. Names the key method, formula, or substitution before any student attempt.
3. Lays out the complete solution path, leaving the student only arithmetic or restating.

Importantly, *phrasing as a question does not exempt a hint*: if a tutor question names, presupposes, or performs the key step (e.g., "what do you get when you factor out x?"), it is treated as if the tutor had stated it. This operationalizes the funneling/focusing Wood (1994) boundary from the math education literature as a scoring rule.

Everything else is OK: guiding questions, narrowing the field, naming a method *after* a student attempt, demonstrating on a different example, and recapping student-produced steps. A REJECT requires a quoted tutor turn as evidence.

C.2 Factual Correctness Gate ($\mathbb{I}_{\text{QUALITY}}$)

The quality judge REJECTs for:

1. **Factual error**: a mathematically incorrect tutor statement.
2. **False correction**: telling the student a correct step is wrong, or confirming an incorrect step as correct.
3. **Error left standing**: a student mathematical mistake that, by conversation end, the tutor has neither elicited a self-correction for nor flagged.
4. **Degenerate output**: heavy markdown, emojis, repeated phrases across 3+ turns, incomplete responses, visible self-corrective scratch-work, or wrong-language fragments.
5. **Dictated speech**: soliciting repetition of tutor-supplied content rather than student-generated reasoning.
6. **No teaching attempt**: the tutor never engages with the problem.

Teaching style, hint strength, and pacing are explicitly excluded from this gate and measured by independent evaluation judges.

D System Prompts

We reproduce the exact system prompts for the tutor, student, and post-test. Prompts mention only general widely accepted principles such as student should cognitively engage with a problem Chi & Wylie (2014) and no pedagogical moves are explicitly named in any prompt. The three axioms are stated as constraints and strategy is left to the policy.

D.1 Tutor System Prompt

```
You are tutoring a student through a math problem.  Your
goal is that they understand deeply enough to solve a
similar problem on their own afterwards.
The student should always show you their attempt first.
Never state the final answer before the student has reached
it themselves.  Do not name the method or key formula
before the student has made an attempt (proposed a method,
wrote a step, stated a guess, or described a dead end ---
``how do I do this?''  is not an attempt).  If the student
cannot produce any attempt after genuinely trying, you may
demonstrate the key idea on a different example and have
them apply it to this one.
The student should do the cognitive work --- the decisions,
not just the calculations:  let them choose the method and
the next step.  Students remember what they choose, work
out, and put into their own words far better than what they
are told.
```

```
Everything you say must be mathematically correct.
Even if the student is already able to solve the problem,
help them deepen and improve their solution so they gain the
strongest possible conceptual and transfer understanding.
Use your private thinking (the student never sees it)
to reason about where the student's mistake is and which
pedagogical move suits the situation.
When the student has demonstrated understanding and no
question of yours is left unanswered, end your final message
with <end_of_conversation> on its own line.
Problem: {problem}
```

D.2 Student System Prompt

```
You are a student in a conversation with a teacher in
training. If possible do not respond with overly long
messages. The conversation with the teacher will be about
this math problem: {problem}
Start the conversation by presenting your initial thoughts
or questions about the problem to the teacher. You may or
may not know how to solve it already, let the teacher guide
you to the correct understanding so you can solve a similar
problem on your own afterwards. 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.
```

The student receives no persona prompt ("act confused") and no predefined misconceptions. Its errors are genuine model failures, not scripted behavior.

D.3 Near-Transfer Post-Test Prompt

After the dialog, the student sees the masked conversation (tutor turns replaced with "(hidden)") followed by:

```
The conversation with the teacher has ended. You are now
tested on a NEW problem, to check whether you can apply what
you just learned on your own.
New problem: {transfer_problem}
It is time you create a step by step complete solution
such that it can be graded without the past conversation.
Always format your final answer in perfect LaTeX
\boxed{final_answer} format.
```

E Near-Transfer Dataset Construction

E.1 Generation Procedure

We construct 8,671 near-transfer problem pairs from a subset of BigMath dataset (Albalak et al., 2025; Dinucu-Jianu et al., 2025) (10,000 source problems for training, 500 for testing). For each source problem x_{src} with answer y_{src}:

1. **Rewrite.** Gemini-3.1-Pro rewrites x_{src} with different numbers and/or a new surface story while preserving the solution structure. The prompt requires that the new numbers produce a "clean" answer of similar form (e.g., integer $\rightarrow$ integer).
2. **Verify.** In a fresh context (without seeing the intended answer), the same model solves the generated variant. The rewrite is accepted only if its solved answer matches the stated one. Up to 3 regeneration attempts are made for mismatches.
3. **Measure prior.** The frozen Llama-3.1-8B-Instruct student samples k=16 independent solutions for both the original and transfer problem. The fraction correct gives $p_{\text{pre}}(x_{\text{src}})$ and $p_{\text{pre}}(x_{\text{tgt}})$.

4. **Difficulty gate.** Variants whose transfer solve rate exceeds 0.75 or whose $|p_{\text{pre}}(x_{\text{tgt}}) - p_{\text{pre}}(x_{\text{src}})| > 0.375$ are rejected and regenerated.

98.8% of problems have a validated transfer variant (9,876/10,000 train; 494/500 test). Problems without a variant fall back to testing on the original problem.

E.2 Solve-Rate Distributions

Table 5 summarizes the dataset statistics for the 8,238 data points and 433 test points (excluding pairs where any solve rate is 0 or any exceed 0.9). The mean student solve rate is 0.289 for source problems and 0.280 for transfer variants, confirming that the rewriter preserves difficulty. The Pearson correlation between source and transfer solve rates is $r \geq 0.65$, indicating that per-problem difficulty is substantially preserved.

Table 5: **Dataset solve-rate statistics**. Solve rates measured with k=16 samples from frozen Llama-3.1-8B-Instruct.

	Train (n=8,238)			**Test** (n=433)		
	Source	Transfer	Corr.	Source	Transfer	Corr.
Mean	0.289	0.280	0.648	0.283	0.280	0.675
Median	0.250	0.188		0.250	0.250	
Std	0.246	0.246		0.240	0.237	
Range	[0, 1]	[0, 1]		[0, 1]	[0, 1]	

Difficulty distribution. The problems are skewed toward harder items: 38.9% have a source solve rate below 0.10, and the distribution thins toward the 0.65 ceiling. This reflects the BigMath source distribution, filtered by the difficulty gate.

E.3 Example Near-Transfer Pairs

Table 6 shows representative source–transfer pairs from the test set, illustrating the rewriter's output across domains. Each pair preserves the solution structure (same concepts, same reasoning steps) while changing numerical values and, where applicable, the surface story. The different final answers confirm that a student who memorized the source answer gains nothing on the transfer variant.

Table 6: **Near-transfer pair examples** from the test set. Each transfer variant preserves the solution method while changing numbers or surface context, yielding a different answer.

Source problem	**Near-transfer variant**
What is the sum of the prime factors of 85,085? *Answer: 53* p_{pre}*: 0.250*	What is the sum of the prime factors of 34,034? *Answer: 50* p_{pre}*: 0.188*
What is the sum of all integer solutions to $\|n\| < \|n-3\| < 9$? *Answer:* -14 p_{pre}*: 0.312*	What is the sum of all integer solutions to $\|m\| < \|m-5\| < 12$? *Answer:* -18 p_{pre}*: 0.188*
Given that m is a root of the equation $x^2-2x-7=0$, find m^2-2m+1. *Answer: 8* p_{pre}*: 0.375*	If r is a root of the equation $x^2-5x-4=0$, determine the value of $r^2-5r+10$. *Answer: 14* p_{pre}*: 0.375*
A triangle has an even perimeter, and two of its sides are 5 and 2,008. How many such triangles exist? *Answer: 4* p_{pre}*: 0.250*	Two sides of a triangle have lengths 7 and 2,024. If the perimeter of the triangle is an even integer, how many such triangles are possible? *Answer: 6* p_{pre}*: 0.188*

F Environment Evaluation Details

Table 7 reports 95% bootstrap confidence intervals (10,000 resamples over problems) for the near-transfer learning gain Δ_{transfer}, same-problem gain Δ_{same}, and solution handover (leak) rate for all models in Tables 1 and 2.

Table 7: **Bootstrap 95% CIs** for Δ_{transfer}, Δ_{same}, and leak rate (n=494 problems, 10,000 resamples). $^{*}/^{\dagger}$ as in Table 1.

	Δ_{transfer}			Δ_{same}			Leak rate		
Model	Mean	95% CI		Mean	95% CI		Mean	95% CI	
Base models									
Qwen3.5-4B*†	+0.10	[0.06,	0.13]	+0.11	[0.08,	0.15]	16.8%	[13.6,	20.2]
Qwen3.5-9B	+0.22	[0.18,	0.26]	+0.35	[0.31,	0.39]	22.7%	[19.0,	26.3]
Qwen3.8-27B	+0.22	[0.18,	0.25]	+0.44	[0.40,	0.48]	8.9%	[6.5,	11.5]
Proprietary									
Gemini 3.1 Pro	+0.21	[0.18,	0.25]	+0.41	[0.37,	0.45]	5.5%	[3.6,	7.7]
Gemini 3.6 Flash	+0.24	[0.21,	0.28]	+0.46	[0.42,	0.50]	6.9%	[4.9,	9.1]
Baselines									
Qwen2.5-7B-Instruct	+0.14	[0.11,	0.18]	+0.18	[0.14,	0.22]	92.9%	[90.7,	95.1]
TutorRL-7B	+0.13	[0.10,	0.17]	+0.22	[0.17,	0.26]	82.8%	[79.4,	86.0]
Eduardo (ours)									
Eduardo-4B	+0.25	[0.21,	0.29]	+0.37	[0.33,	0.42]	30.6%	[26.7,	34.6]
Eduardo-9B	+0.25	[0.21,	0.29]	+0.47	[0.43,	0.51]	25.3%	[21.5,	29.1]
Eduardo-27B	+0.29	[0.25,	0.32]	+0.46	[0.42,	0.51]	9.3%	[6.9,	11.9]
Ablations (4B)									
w/o Binary gates	+0.23	[0.20,	0.27]	+0.38	[0.34,	0.43]	60.7%	[56.5,	65.0]
w/o Masking	+0.24	[0.20,	0.27]	+0.39	[0.35,	0.44]	13.4%	[10.3,	16.4]
w/o Near-Transfer	+0.23	[0.19,	0.27]	+0.40	[0.35,	0.44]	47.2%	[42.7,	51.6]

Truncation and empty-turn rates. Table 8 reports per-model diagnostics. *Empty%* is the fraction of tutor turns where reasoning overran the per-turn token cap (4,192 tokens including visible output) and reached the student as an empty message, with no visible content. *Seq. trunc.%* is the fraction of conversations where the total sequence length hit the 16,384-token budget before the dialog ended naturally, cutting the conversation short. This primarily affects smaller base models whose thinking is verbose: Qwen3.5-4B produces empty turns 70% of the time, Qwen3.5-9B 20%, and Qwen3-14B 8%. All EDUARDO models stay below 11%, and Eduardo-27B below 1%. Arms exceeding 15% on either rate are flagged (* empty turns, † sequence truncation) in Tables 1, 2, and 8; their environment metrics understate Δ and Leak.

G Example Transcripts

The following excerpts are from TutorMoments evaluations on the same scenario (non-linear sequences) under the plain prompt (no rubric). Both models tutor the same Gemini-3.7-Flash student on the same problems, isolating what training changed.

Example 1: Prompting for justification (Move G). *Context: The student must identify which of two sequences (Katie's: $+2$ each time; Tom's: $\times 2$ each time) is arithmetic.*

Eduardo-27B (plain prompt, no rubric):
`TUTOR`: What's the key difference between Katie's sequence and Tom's?
`STUDENT`: Katie's is adding by two each time, which makes it arithmetic, and Tom's is multiplying by two, which is geometric.
`TUTOR`: That's right. So which option does that lead you to?
`STUDENT`: So then B. Only Katie.

Table 8: **Truncation diagnostics** per model on the 494-problem test set. *Avg turns*: mean tutor turns per dialog. *Empty%*: fraction of turns where reasoning exceeded the per-turn cap and produced no visible output. *Think tok/turn*: mean thinking tokens per tutor turn. *Seq. trunc.%*: fraction of conversations cut short by the 16,384-token sequence budget. *EoC rate*: fraction of dialogs ending with the tutor's end-of-conversation token (vs. hitting turn or token limits). Values $>15\%$ in **bold**. Rule used in all tables: *Empty% $>15\%$, †Seq. trunc.% $>15\%$; for flagged rows Δ and Leak are understated and judge scores are depressed.

Model	Avg turns	Empty%	Think tok/turn	Seq. trunc.%	EoC rate
Qwen3.5-4B (base)*†	4.40	**70.0%**	596	**31.2%**	68.8%
Qwen3.5-9B (base)*†	4.74	**19.5%**	1504	**15.2%**	84.8%
Qwen3-14B (base)†	6.35	8.1%	1102	**26.1%**	59.5%
Qwen3.8-27B (base)	7.05	1.3%	427	3.8%	82.0%
Eduardo-4B	8.18	3.7%	544	10.7%	69.6%
Eduardo-9B	7.87	2.1%	496	6.7%	76.3%
Eduardo-14B	6.26	10.7%	1060	11.1%	85.2%
Eduardo-27B	9.51	0.4%	57	0.0%	53.4%
w/o Binary gates (4B)	6.39	2.7%	398	5.1%	91.5%
w/o Masking (4B)*†	7.49	**26.9%**	718	**39.7%**	49.4%
w/o Near-Transfer (4B)	7.13	3.8%	761	10.1%	77.9%

Qwen3.8-27B (base) (same prompt, same problem):
`TUTOR`: Exactly! Katie's is the arithmetic sequence because it keeps adding 2 each time. Tom's keeps multiplying by 2, so that one isn't arithmetic. Go ahead and check B.
`STUDENT`: Okay.
`TUTOR`: Go ahead and click Check on B.
`STUDENT`: Yeah. That was pretty simple.

Eduardo asks the student to articulate the conceptual distinction; the base model states the full explanation itself. The reward never names "prompting for justification"; it emerged from the masked transfer objective.

Example 2: Guiding questions (Move A) vs. telling (Move C). *Context: Same scenario, earlier problem—find the third term of a sequence defined by tripling* $(2, 6, ?)$.

Eduardo-27B:
`TUTOR`: What is the second term in the sequence?
`TUTOR`: And what does the problem say we do to get from one term to the next?
`STUDENT`: The second term is six. And it says it's found by tripling, so multiply by three.
`TUTOR`: If the second term is 6, what do you need to do to find the third term?
`STUDENT`: Six times three, which equals eighteen. So it's A.

Qwen3.8-27B (base):
`TUTOR`: So we start with 2, and tripling it gives us 6. Now, what do we do to 6 to find the third term?
`STUDENT`: You multiply by three, so eighteen. A.
`TUTOR`: That's the "tripling" rule. That's what makes it non-linear.

Eduardo decomposes the problem into student-driven steps; the base model states the first step ("tripling gives us 6") and explains the concept after the student answers. In both examples, the trained model elicits reasoning the base model provides.

H Evaluation Judge Rubrics

The Ped. and Fact. scores in Tables 1 and 2 are assigned by Gemini-3.1-Pro, which never enters the reward. The judge sees the problem, its correct answer and the full tutor–student dialog, scores the *whole* dialog on one criterion per call on a 1–5 scale, and is instructed to be strict and use the

full range. Each prompt also contains two to three few-shot examples and a JSON output format, omitted here. Table 9 gives the factuality rubric and Table 10 the pedagogy rubric.

Table 9: **Factuality rubric (Fact.): Correctness & Language Appropriateness.** Is everything the tutor says mathematically correct, and is it expressed at the right level for this problem and this student? Correctness covers hints, confirmations, corrections and intermediate results; the student's level is judged from their own messages.

Score	Description
5	Everything stated or implied is correct and consistent with the correct answer; correct student steps are confirmed and wrong ones flagged; terminology and sentence complexity match the student's level, precise where it matters and simple where possible, with no misconception-inducing shortcuts.
4	Correct overall, with one slightly imprecise or ambiguous statement, or language occasionally pitched a bit too high or too low, without misleading the student.
3	One clear factual error or wrong/missed judgment of a student step, *or* language persistently mismatched (jargon-heavy for a struggling novice, or so simplified it distorts the math), though the dialog still heads toward the correct answer.
2	Multiple factual errors, validation of an incorrect solution path, or oversimplified explanations that instill a genuine misconception.
1	Guidance is fundamentally wrong: it contradicts the correct answer or leads the student to a wrong final result.

Watch for: incorrect claims; confirming wrong calculations; rejecting correct reasoning; hints inconsistent with the correct answer; a wrap-up that leaves a wrong result; undefined terms above the problem's level; imprecise phrasing that creates false general rules (e.g., "multiplying always makes numbers bigger"); vocabulary mismatched to the student's demonstrated level.

Table 10: **Pedagogy rubric (Ped.).** Good tutoring uses *contingent scaffolding* (help calibrated to the student's demonstrated need, minimal hint first, fading as competence grows), allows *productive struggle* (room to attempt, err and self-correct; errors diagnosed and turned into reasoning opportunities), and maintains *rigour* (pressing for justification and avoiding funneling). The student does the actual reasoning and most of the talking.

Score	Description
5	Excellent: support is contingent (the smallest hint that unblocks, escalated only when needed) and fades over the dialog; the student struggles and self-corrects; errors are probed, not corrected outright; the tutor presses for justification; no final answer or key intermediate result is revealed before the student derives it.
4	Good: mostly contingent guidance with small lapses, such as an occasional premature hint, mild funneling, a minor result confirmed slightly early, or missed chances to ask "why".
3	Mixed: some genuine guidance, but the tutor rescues too fast after errors, reveals an important intermediate step or does a key calculation, funnels the student through trivial steps, or dominates the dialog.
2	Poor: the tutor reveals the final answer or walks through most of the solution after brief guiding, corrects errors directly, or lectures with little use of the student's responses.
1	No pedagogy: the final answer or complete solution is given immediately (in any format), or the student's reasoning is ignored entirely.

Watch for: stating the final answer or key intermediate results; performing calculations or substituting values the student did not provide; hints before the student attempted or struggled; correcting an error instead of prompting the student to find it; funneling (questions so narrow the answer is trivial); never asking for justification; talk-time balance (the student should talk at least as much as the tutor).