What Drives Interactive Improvement from Feedback?

Bartłomiej Cupiał[1,2,*] **Jan Łojek**[1] **Mikołaj Garstecki**[1]
Szymon Pobłocki[1] **Alicja Ziarko**[1,2,3] **Piotr Miłoś**[1,4,5]

Abstract

We study when natural-language feedback produces improvement beyond the gains obtainable from repeated attempts alone. In multi-turn language agent setting, higher final accuracy can reflect useful feedback, but it can also arise from resampling, format correction, or additional test-time computation. To separate these effects, we introduce a controlled student-teacher protocol across Omni-MATH, Codeforces, BBEH Linguini, and ARC-AGI1, evaluating thirteen open-weight models in both student and teacher roles. We compare external feedback, self-feedback, and unguided self-refinement, while varying interaction history, task difficulty, and teacher access to privileged task information. Across settings, we find that multi-turn improvement is often not evidence of feedback use: self-generated feedback adds little beyond unguided self-refinement, whereas the strongest external teachers produce substantially larger feedback-specific gains, suggesting that useful feedback must provide guidance beyond generic retry. Dense student-teacher interaction matrices further show that interactive gains are driven more by the student's ability to use feedback than by the teacher's identity, although teacher choice remains important for a fixed student. These results suggest that feedback-based agents should be evaluated against repeated-attempt baselines, and that ability to act on feedback, not merely feedback availability, is a central bottleneck for interactive improvement. We release our controlled student-teacher evaluation framework at https://janlojek.cc/feedback-generation-is-a-bottleneck/.

1 Introduction

Language models (LMs) are increasingly deployed as agents that act, observe, and revise rather than as systems that produce a single isolated answer. Coding agents react to tests and error traces, computer-use agents observe changing interface states, and assistants revise plans in response to user corrections or clarifications. Across these settings, feedback can do more than mark success or failure: it can explain what went wrong, identify missing constraints, and suggest a path to recovery. Improving from feedback is therefore becoming an important capability in its own right: an agent must recognize which parts of the feedback are actionable, preserve the parts of its attempt that are already correct, and revise the parts that caused failure.

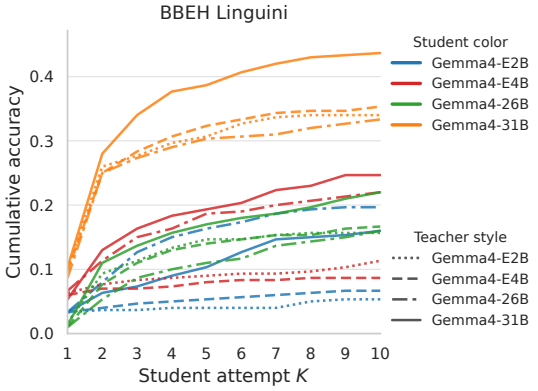

Figure 1: **Multi-turn trajectories separate mostly by student.** While stronger teachers like Gemma4-31B provide noticeable boosts for highly capable students, the dominant vertical separation between the curves is driven by student rather than teacher.

Accepted to "RLxF: Reinforcement Learning from World Feedback".
*Corresponding author: bartlomiej.cupial@gmail.com. [1]University of Warsaw [2]AKCES NCBR [3]Princeton University [4]Mistral AI [5]Institute of Mathematics, Polish Academy of Sciences

Studying improvement from feedback requires separating several mechanisms that are usually entangled. A model may improve over turns because it receives useful corrective information, but it may also improve simply because it samples another attempt with more computation. When feedback does help, the limiting factor may lie on either side of the interaction: the teacher may fail to identify the relevant error, or the student may fail to translate a correct diagnosis into a better solution. We therefore distinguish feedback generation from improving from feedback by evaluating separate student and teacher models under controlled interaction histories, teacher-information conditions, and repeated-attempt baselines. This lets us ask not only whether feedback improves performance, but when it improves performance beyond self-refinement, and which part of the feedback system constrains that improvement.

We answer these questions in a student-teacher protocol on verifiable tasks. In each episode, a student model attempts a problem. If the attempt is incorrect, a teacher model provides natural-language feedback, and the student revises its answer. The verifier makes every attempt measurable, while the student-teacher split lets us vary the student, the teacher, the interaction budget, the visible history, and the teacher's access to privileged task information. We instantiate this evaluation across four verifiable reasoning environments: Omni-MATH, Codeforces, BBEH Linguini, and ARC-AGI1, and run student-teacher interaction grids in which each model acts in both roles.

Across these settings, we find that multi-turn improvement is often not evidence of feedback use: repeated attempts explain a large share of the gain, self-feedback improves over generic retry inconsistently, the strongest feedback-specific improvements come from high-quality external teachers, and the student's ability to use feedback explains more variation than the teacher's identity.

In summary, our contributions are as follows:

- We introduce a controlled zero-shot student-teacher evaluation for natural-language feedback across four verifiable reasoning environments.
- We show that multi-turn improvement is not necessarily feedback use: self-feedback adds little beyond self-refinement, while the best external teachers add substantial feedback-specific gains.
- We find that interactive performance is primarily student-dependent: student identity explains substantially more gain variation than teacher identity, although teacher choice can matter for a fixed student.
- We show that longer histories and privileged teacher information are selective interventions rather than default improvements, with benefits depending on task and model capability.

2 Related Work

Self-refinement and repeated sampling. Large language models are increasingly used as interactive agents that solve tasks over multiple turns rather than through a single generation [Wang et al., 2024]. In such settings, models must follow sequential instructions and incorporate new feedback across turns [Laban et al., 2025]. However, multi-turn improvement does not necessarily imply that the model has used feedback: later attempts also receive more test-time computation and additional samples. This distinction is especially important because additional computation is known to improve performance through mechanisms such as chain-of-thought prompting [Wei et al., 2023], and Best-of-N sampling [Wang et al., 2023]. Related agent systems such as Reflexion use verbal feedback or memories of past failures to improve future attempts [Shinn et al., 2023]. Our work builds on this line of work by explicitly comparing feedback-conditioned improvement against unguided self-refinement, allowing us to estimate how much of the observed gain is feedback-specific rather than a consequence of repeated attempts.

Post-training LMs with teacher feedback. Recent post-training methods increasingly frame model improvement as a student-teacher problem, where a student model learns from signals produced by a stronger teacher, verifier, or environment. Standard approaches include supervised fine-tuning, preference optimization, reinforcement learning from AI feedback, and reinforcement learning with verifiable rewards [Ouyang et al., 2022, Lee et al., 2024, Rafailov et al., 2024, Guo et al., 2025, Shao et al., 2024]. These methods are effective, but their feedback signals are often demonstrations, pairwise preferences, scalar rewards, or outcome-level verification. Such signals can improve behavior,

but they provide limited information about which part of a student's attempted solution caused

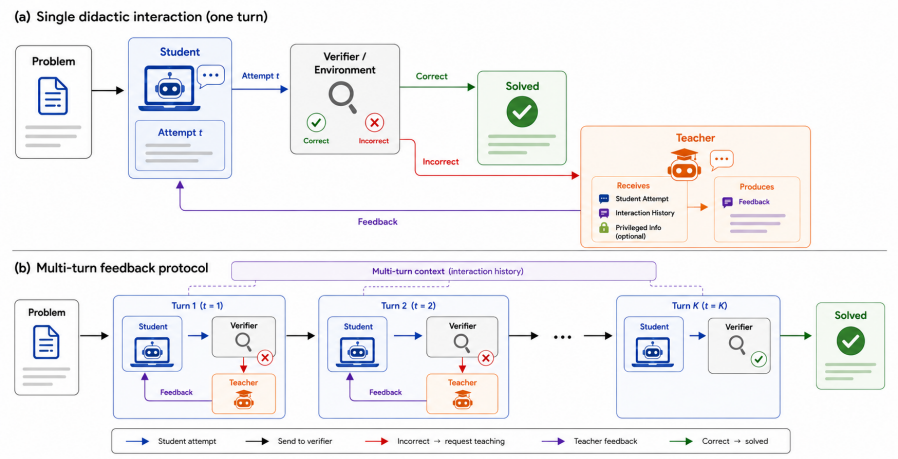

Figure 2: **(Top) Single Didactic Interaction.** We transform single-turn problems into multi-turn didactic interactions. A teacher model, conditioned on optionally privileged information (e.g., the ground-truth solution), provides natural language feedback to a student model without revealing the final answer, guiding it to correct its errors. **(Bottom) Multi-turn feedback protocol.** Each failed student attempt is converted into teacher feedback for the next attempt. This cycle repeats until the predefined maximum number of turns is reached or the student correctly solves the problem.

failure. This limitation has motivated recent work on denser teacher supervision, including on-policy distillation, where trajectories are sampled from the student and a teacher provides supervision on the student-generated states [Lu and Lab, 2025]. It has also motivated work on synthetic and interactive data generation, especially as high-quality human-generated training data becomes increasingly limited [Villalobos et al., 2024, Kessler et al., 2026]. In verifiable domains, natural-language feedback can be used to construct multi-turn training trajectories or improve interactive learning from feedback [Klissarov et al., 2026, Cook et al., 2026]. Related work has also used natural-language feedback as a fine-tuning signal or as part of reinforcement-learning-style updates [Choudhury and Sodhi, 2024, Feng et al., 2024]. Our work studies the test-time analogue of this post-training shift: a teacher observes a student's failed attempt and provides diagnostic feedback, and we ask when this feedback improves performance beyond repeated attempts, whether stronger teachers provide more useful corrections, and whether the main bottleneck lies in feedback generation or student uptake.

Feedback generation and tutoring. Several works study the quality of feedback itself. [Xu et al., 2025] identify conditions under which language feedback can support no-regret learning, including unbiased feedback, the agent's ability to translate feedback into evaluations of candidate hypotheses, and sufficient representational coverage of the true environment. In mathematical reasoning, [Li et al., 2024] evaluate whether models can detect and correct errors in flawed reasoning traces, while [Gupta et al., 2025] analyze the tutoring behavior of GPT-family models beyond final-answer correctness. These works focus on whether feedback is valid, helpful, or pedagogically appropriate. Our work complements them by evaluating the full student-teacher interaction: a teacher may generate useful feedback, but improvement also depends on whether the student can act on it.

3 Experimental Setup

In this section we describe our experimental setup.

Environments. We restrict our evaluation to environments, where for each problem and solution pair, it is possible to verify whether the given solution is correct. This assumption is often made in the literature . To make our results robust, we consider four challenging environments, that focus on different domains and test for different abilities: **Omni-MATH** [Gao et al., 2024] Olympiad-level mathematical reasoning benchmark. **Codeforces** [juvi21, 2024], a competitive programming benchmark, the **Linguini** task from **BIG-Bench Extra Hard (BBEH)** [Kazemi et al., 2025], which

specifically challenges the model's ability to learn new rules and **ARC-AGI** [Chollet, 2019], which presents grid transformation puzzles solvable only by inferring abstract rules from a few examples.

Interaction protocol. For investigating the influence of feedback, we consider a realistic scenario of a back-and-forth conversation between a model and an entity capable of giving it feedback (for simplicity, we call them student and teacher). Each episode is a trajectory $\tau = (\text{problem formulation}, s_1, a_1, t_1, \ldots, s_n, a_n)$, where the initial model input contains the problem formulation. In turn i, s_i is the student's output, a_i is the student's answer. In the case where the task-specific verifier recognized a_i as the wrong answer, t_i is the teacher's feedback on the student's attempt. Then, the next turn $i+1$ starts, where the student model attempts to solve the problem again, with the added information of the teacher's feedback. The conversation continues until a maximal n is reached or the student's answer is correct. Our interaction protocol is visualized in Figure 2.

Student's input. Lead by the intuition that the recent feedback is more important than old feedback, we restrict the student's input to only h historical turns, where h is a hyperparameter. Therefore, the student's input is $(\text{problem formulation}, s_k, a_k, t_k, \ldots, s_n, a_n)$, where $k = \max(n - h, 0)$.

Teacher's input. In previous works [Klissarov et al., 2026, Cook et al., 2026], the teacher model was granted access to specific task knowledge, denoted as k_t. Due to the varying nature of the evaluation environments we have chosen, this additional information in this work have taken different forms depending on dataset used: an in-depth step-by-step solution, a plain final answer, or an error stack trace derived from the student's generated code. Because smaller models are prone to inadvertently leaking the direct solution when given such context, we establish a strict baseline where no additional information is provided to the teacher ($k_t = \text{none}$). However, to better understand the impact of privileged context, we also include ablations that selectively provide this knowledge based on the specific environment. The configurations are as follows: for the Math environment, the teacher receives either no information, the final answer, or the full solution; for the BBEH environment, the knowledge is restricted to either no information or the final answer; and for the Code environment, the teacher receives either no additional information or the complete solution.

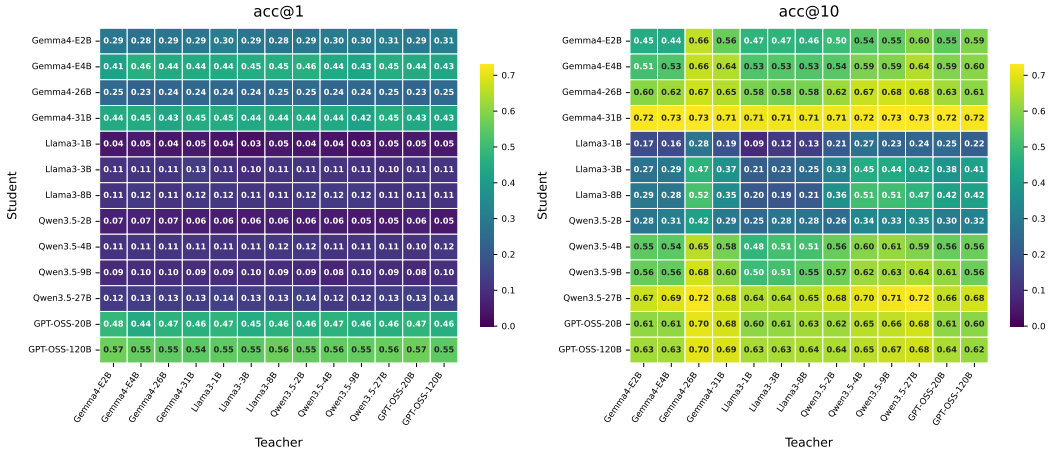

Figure 3: **Dense matrices separate zero-shot ability from feedback-mediated performance.** Rows are students and columns are teachers for Omni-MATH. The left panel reports acc@1 before feedback, so values are constant across each student row; the right panel reports acc@10 after up to ten attempts with teacher feedback. Comparing the panels shows that feedback changes final performance substantially, some brighter columns highlight strong teachers, with the strongest structure remaining row-wise, indicating that the student model is a major driver of interactive outcomes.

Models considered. To evaluate the performance of both teacher and student, we choose from thirteen models from four open-weight families, spanning a parameter range of 1B to 120B, both dense and Mixture-of-Experts (MoE) architectures, and models with and without dedicated reasoning modes. These are series of Gemma4-{E2B, E4B, 26B-A4B, 31B} [Gemma Team, 2026]; the collection of Llama-3.x-{1B, 3B, 8B} [Dubey et al., 2024]; the family of Qwen3.5-{2B, 4B, 9B, 27B} [Team, 2026]; and the GPT-OSS models: GPTOSS-{20B, 120B} [OpenAI, 2025].

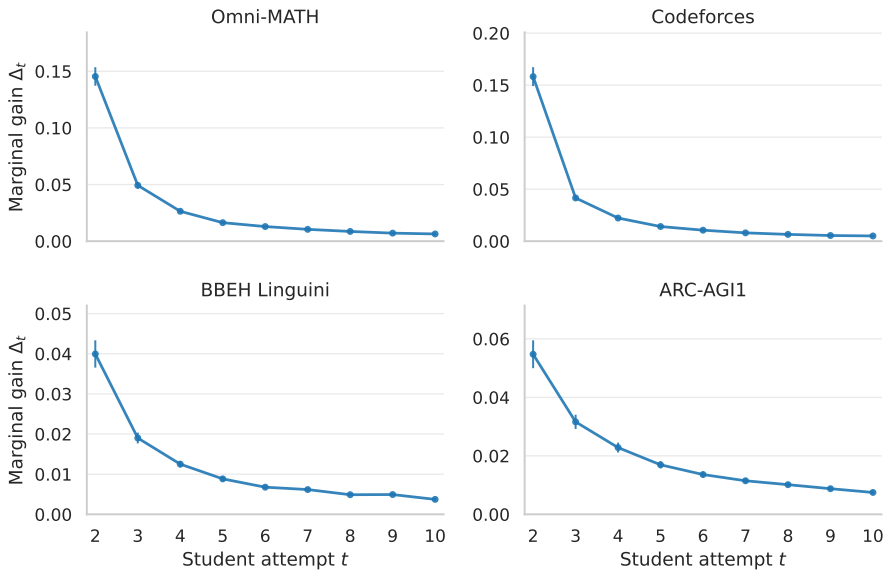

Figure 4: **Most feedback gains arrive in the first few turns.** Each panel reports the marginal improvement from one attempt to the next, averaged over dense-matrix student-teacher pairs. The first feedback-conditioned attempt produces the largest gain, and later turns contribute progressively smaller improvements. This front-loaded pattern suggests that short feedback trajectories capture most of the benefit, while longer rollouts are mainly useful for harder cases with slower recovery.

Ablation tier. To further deepen our research, we conducted ablation studies to isolate the effect of different parameters on multi-turn performance. Due to the large search space and computational limitations, we focused solely on the Gemma-4 family, resulting in a 4×4 student-teacher matrix for each ablation. This provides us with a relatively exhaustive set of interactions between smaller and larger models. Because of the conversation setup, we test whether access to information from previous turns yields any benefits. For this purpose, we vary the maximum visible history length in $\{1, 2, 3, 4, 5\}$. Additionally, we test how the number of turns, also called the interaction budget, affects the final accuracy. We evaluate it for $K \in \{1, 5, 10\}$. Since the conversation progresses sequentially, we can generate the full sequence for $K = 10$ and subsequently extract the states for $K = 5$ and $K = 1$ during post-processing. Finally, to decompose the gains from additional test-time compute, we added a self-refinement baseline for each model. This approach maintains the student's history but does not receive feedback from the teacher.

Metrics. To evaluate the different configurations and teacher-student pairs, we measure the accuracy at various steps of the interaction trajectory, denoted as acc@K. For a more comprehensive analysis, we introduce additional metrics: raw gain (acc@K − acc@1), normalized gain ($\frac{\text{acc@}K - \text{acc@1}}{1 - \text{acc@1}}$), and the Area Under the Curve computed as the mean cumulative accuracy over all turns ($\frac{1}{T}\sum_{k=1}^{T} \text{acc@}k$). Both acc@$K$ and AUC serve as primary measures of the system's overall problem-solving capability, additionally reflecting how quickly the student improves across turns. Raw and normalized gain instead isolate how much of that improvement is attributable to the teacher's feedback and iterative reasoning, independent of the student's starting level. Notably, normalized gain provides a saturation-adjusted comparison across different model scales by accounting for benchmark saturation, ensuring that larger models are not penalized for possessing a higher initial baseline accuracy (acc@1).

4 What matters for improvement from feedback?

In this section we present our experimental results. We organize them around the following six empirical questions, which we answer through research findings (RFs):

Table 1: **Disentangling interaction, feedback, and feedback quality.** Self-refinement measures improvement from additional attempts without external feedback; self-feedback adds feedback generated by the same model; and best feedback selects the strongest available teacher for each student. Cells report mean acc@10 with mean gain@10 in parentheses, in percentage points, averaged over students with all three conditions in the same environment. Full per-student results are reported in table 7.

Environment	Self-refinement	Self-feedback	Best feedback
Omni-MATH	43.9 (+20.7)	48.6 (+24.7)	60.5 (+36.9)
Codeforces	52.8 (+17.8)	57.0 (+23.8)	68.7 (+35.1)
BBEH Linguini	12.0 (+7.6)	12.6 (+9.1)	21.6 (+17.6)
ARC-AGI1	18.2 (+11.6)	26.9 (+17.1)	33.2 (+23.1)

1. How much of multi-turn improvement comes from feedback rather than repeated attempts? (RF1)
2. Can a student that is weaker on the first attempt become stronger after receiving feedback? (RF2)
3. Is interactive performance driven primarily by the student receiving feedback or by the teacher generating it? (RF3)
4. Is a teacher's single-turn task performance a good predictor of its competence as an interactive teacher? (RF4)
5. How much improvement do longer interaction histories buy, given their cost and context-length tradeoffs? (RF5)
6. When does teacher access to privileged information improve interactive outcomes? (RF6)

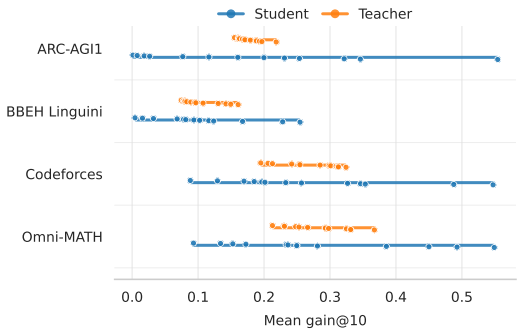

Figure 5: **Student identity explains more gain variation than teacher identity.** Points show role means at K = 10, averaging over the opposite role; segments show the min-to-max spread. Across environments, student means span a much wider range than teacher means.

4.1 Feedback-specific gains and interaction budget

RF1: Feedback-specific gains are limited unless the teacher provides substantially better guidance than generic retry or self-feedback. The feedback-control comparison in table 1 shows that multi-turn improvement is not, by itself, evidence that a student benefits from feedback. Averaged over matched students, self-feedback is inconsistent relative to the self-refinement baseline: it changes acc@10 by -1.2 to $+8.7$ points and gain@10 by only $+0.1$ to $+6.0$ points across environments. The largest self-feedback lift appears on ARC-AGI1, while BBEH Linguini slightly favors unguided self-refinement in final accuracy. Thus, simply adding a feedback message, even one generated by the same model, does not guarantee feedback-specific improvement. In contrast, selecting the best available teacher for each student adds 9.2 to 16.6 points in acc@10 and 9.9 to 17.3 points in gain@10 over self-refinement. This suggests that gains depend not just on the presence of a feedback message, but on whether the feedback provides corrective information that the student can use.

Interaction budget also matters. The marginal-gain curves in Figure 4 show that the benefits of additional turns are front-loaded. Attempt 2, the first with feedback, accounts for 30.5%–58.1% of total gain@10, and attempts through $K = 5$ recover 70.8%–86.8%. Later turns can still matter for hard long-horizon cases, especially ARC-AGI1, but their average marginal return is small. For data collection, this favors shorter trajectories with informative feedback over long rollouts whose gains mostly come from repeated attempts.

4.2 Dense model matrix: separating student and teacher effects

We next analyze the dense model matrix described in section 3: each model is evaluated both as a student and as a teacher across the four environments, with episodes run for up to $K = 10$ student

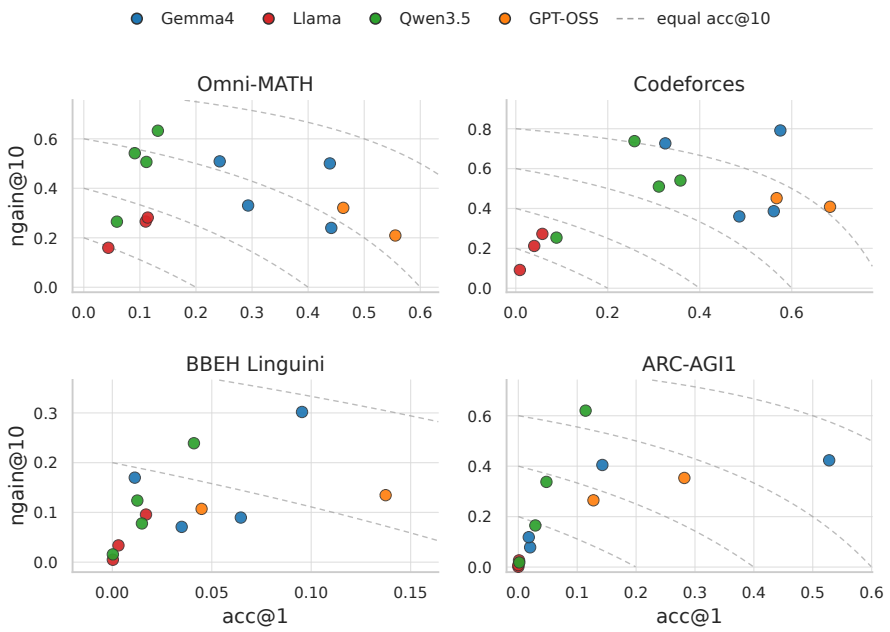

Figure 6: **First-turn accuracy and feedback uptake are separate capabilities.** Each panel shows one environment, and each point is a student model averaged over teachers. Color denotes model family. The y-axis reports $\mathrm{ngain@10} = (\mathrm{acc@10} - \mathrm{acc@1})/(1 - \mathrm{acc@1})$, the fraction of initially unsolved problems recovered by the interaction. Dashed curves mark equal final $\mathrm{acc@10}$ values.

attempts. This cross-product lets us ask whether interactive outcomes are explained more by the model receiving feedback or by the model generating it. Figure 3 shows a concrete slice of this matrix for Omni-MATH, while the full dense matrices for all environments and metrics are reported in section B.3.

RF2: First-turn accuracy and feedback adherence are different capabilities. A high initial solve rate does not imply that a model makes the best use of feedback. Figure 6 shows that this distinction is not a one-off comparison: students with similar $\mathrm{acc@1}$ can have very different ngain. A qualitative example can be seen in A.8. It showcases how teacher and student cooperate to get to correct solution. A concrete comparison is given in table 6 of section B.2: GPT-OSS-120B starts higher on Omni-MATH, Codeforces, and BBEH Linguini, but Gemma4-31B usually has much larger gain and reaches the better final outcome. ARC-AGI1 is the complementary case: GPT-OSS-120B has slightly larger gain, but Gemma4-31B starts much higher and remains the best final student. Thus $\mathrm{acc@1}$ measures what the student can solve immediately, while ngain measures how efficiently it converts feedback rounds into recovery opportunities.

RF3: Interactive performance is mostly student-dependent. We use the dense matrix to marginalize over roles: for each student, we average performance over teachers; for each teacher, we average performance over students. Figure 5 shows the resulting role means for gain at $K = 10$. Across all environments, the spread across students is much larger than the spread across teachers, indicating that the model receiving feedback explains more of the interactive gain than the model generating it. The appendix fixed-effect analysis in table 3 supports the same conclusion. Student fixed effects explain 77.1%–96.5% of pair-level gain variation, while teacher fixed effects explain only 1.3%–12.4%. Adding teacher identity after student identity gives a small incremental improvement, at most 0.124 R^2. This does not mean that teacher identity is irrelevant. For a fixed student, the gap between its best and worst teacher can be substantial, as shown by the appendix teacher-range analysis in table 4. The dense-matrix results suggest that teacher effects are smaller, but can become meaningful once we condition on a particular student.

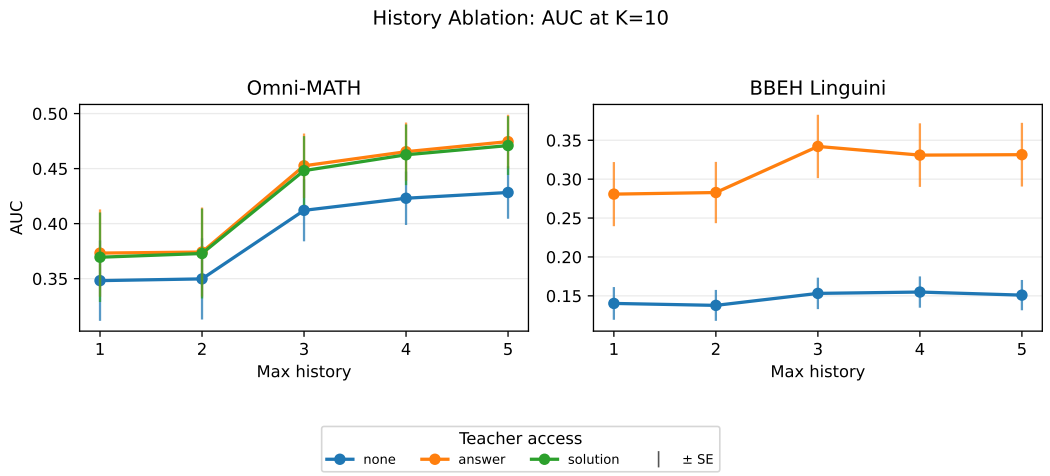

Figure 7: **History and privileged teacher information help, but for different reasons.** Each panel reports auc at $K = 10$ as we vary the maximum number of previous interaction turns retained in the student and teacher prompts for Gemma4 student-teacher pairs, with line color indicating teacher reference access.

RF4: Teacher task competence only partially predicts interactive teacher competence. We use each model's $\mathrm{acc@1}$ as a student as a noisy proxy for single-turn task competence, and ask whether this ranking predicts its performance as a teacher for a fixed student. If models that solve more problems on the first turn were always stronger teachers, teacher rank and interactive auc would be monotonic within each student. Table 5 shows that this monotonicity only partially holds. Codeforces has the strongest relation, with mean Spearman $\rho = 0.56 \pm 0.02$, but the other environments are weaker (0.21 ± 0.06 to 0.39 ± 0.06). Inversions are also common: in $28.9\% \pm 1.1\%$–$41.4\% \pm 1.9\%$ of comparable teacher pairs, the lower-$\mathrm{acc@1}$ teacher produces higher interactive auc. This suggests that single-turn solving ability and teaching ability are related but not equivalent. A stronger solver may be more likely to know the correction, but interactive performance also depends on diagnosing the student's specific error, phrasing actionable feedback, and matching the student's ability to use it.

4.3 Gemma4 ablations: interaction history and privileged teacher information

The dense matrix uses a relatively cheap and scalable protocol: short visible history and no ground-truth answer or solution access for the teacher. We use this as the main setting because both alternatives create tradeoffs. Longer histories may help models diagnose repeated failures, but they increase cost and can run into context limits. Privileged information may make feedback more accurate, but answers or solutions are not always available, may be costly to obtain, and can leak into the teacher's feedback. The Gemma4 ablations measure how much performance these more expensive sources of information buy.

RF5: Longer visible history is a capability-dependent tradeoff. Longer histories expose repeated failure modes to the teacher and give the student access to earlier corrections. This information is not free: every additional visible turn increases cost and makes context overflow more likely. It is also not automatically useful, the models must be able to identify which parts of the history matter and use them to revise the next attempt. The max-history ablation shows little change from history 1 to 2, followed by a larger jump at history 3. Averaged over teacher-access settings and Gemma4 pair cells, moving from history 1 to 5 increases auc by 0.094 on Omni-MATH. In the full grid, Codeforces shows the largest average history benefit, with auc increasing by 0.123. BBEH Linguini shows a smaller but positive gain, rising from 0.210 at history 1 to 0.248 at history 3. These averages suggest that history can help, but the benefit should be interpreted as conditional on model capacity rather than as a free improvement from longer context.

This capability dependence is visible when we split by teacher-student strength. In the matched history-5 minus history-1 diagnostic (figure 17), Codeforces gains about 0.21 auc for pairs where

the teacher's single-turn $\mathrm{acc@1}$ exceeds the student's, compared with about 0.05 for the remaining pairs. This is consistent with stronger teachers using history to diagnose repeated failures, although the ablation still does not isolate teacher adaptation from student-side conditioning.

RF6: Privileged information helps selectively, not uniformly. We ask whether giving the teacher additional task-side information reliably improves the interaction. The answer is task-dependent. At history 5, BBEH Linguini shows the largest effect: answer access increases auc from 0.151 to 0.331 and $\mathrm{acc@10}$ from 0.195 to 0.463. This suggests that privileged information is most useful when the teacher would otherwise struggle to infer the target correction. This large BBEH separation is visible in figure 7 The effect is much smaller elsewhere. On Omni-MATH, answer access improves auc from 0.428 to 0.475, while full solution access reaches only 0.471. On Codeforces, where the teacher already receives structured execution feedback, adding the reference solution changes auc only from 0.547 to 0.556. Thus, additional teacher-side information is not automatically useful; it helps when it resolves a real teacher uncertainty and can be converted into actionable feedback. The teacher-level lift diagnostic in figure 18 shows the same non-uniformity across teachers and environments. Because answers and solutions are not always available and may leak into the feedback, we treat these conditions as upper-bound ablations rather than the default scalable protocol.

5 Discussion

Summary Our findings have practical consequences for evaluating and building feedback-based agents. First, repeated-attempt and self-feedback controls should be included whenever multi-turn gains are reported; otherwise, extra inference compute can be mistaken for feedback use. Second, teacher selection should not be based only on single-turn task accuracy. The teacher monotonicity analysis shows that stronger solvers are not always stronger teachers for a particular student. Third, longer histories and privileged teacher information should be treated as targeted interventions rather than default improvements: they help when they resolve a real diagnostic bottleneck, but they also increase cost, context pressure and leakage risk.

Limitations Our study has several limitations. First, the current ARC-AGI1 analysis does not yet include the same ablations as the other environments, neither does it contain a leakage audit for settings where teachers receive privileged answers or solutions. Although we provided preliminary qualitative analysis of the feedback trajectories, conducting a broader analysis in the future will be necessary to achieve more conclusive results. These analyses are important for understanding whether improvements come from actionable feedback rather than formatting artifacts, repeated sampling, or direct leakage from the teacher context. Second, our experiments focus on a zero-shot setting with short-horizon verifiable tasks and open-weight models. This makes the protocol controlled and reproducible, but it leaves open how the conclusions transfer to models trained to use feedback, longer-horizon interactive domains, and closed-source systems.

Future Work A natural direction for future work is to connect this diagnostic protocol to post-training. In supervised fine-tuning or reinforcement-learning settings, feedback trajectories could be used not only to measure whether a model improves within an episode, but also to train models that become better feedback users over time. Extending the study to longer-horizon domains such as computer usage would test whether the same student-side bottleneck appears when feedback must be integrated across many actions, persistent state, and delayed consequences. Finally, comparing open-weight and closed-source models would clarify which findings reflect general properties of current language models and which depend on the model families studied here.

References

François Chollet. On the measure of intelligence. *arXiv preprint arXiv:1911.01547*, 2019.

Sanjiban Choudhury and Paloma Sodhi. Better than your teacher: Llm agents that learn from privileged ai feedback, 2024. URL `https://arxiv.org/abs/2410.05434`.

Jonathan Cook, Diego Antognini, Martin Klissarov, Claudiu Musat, and Edward Grefenstette. Learning to learn from language feedback with social meta-learning, 2026. URL `https://arxiv.org/abs/2602.16488`.

Abhimanyu Dubey, Akhil Jauhri, Abhinav Pandey, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Xidong Feng, Ziyu Wan, Mengyue Yang, Ziyan Wang, Girish A. Koushik, Yali Du, Ying Wen, and Jun Wang. Natural language reinforcement learning, 2024. URL https://arxiv.org/abs/2402.07157.

Yibo Gao et al. Omni-math: A universal olympiad level mathematics benchmark for large language models, 2024.

Google DeepMind Gemma Team. Gemma 4: Open models for advanced reasoning and efficiency. Technical report, Google DeepMind, 2026.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL http://dx.doi.org/10.1038/s41586-025-09422-z.

Adit Gupta, Jennifer Reddig, Tommaso Calo, Daniel Weitekamp, and Christopher J. MacLellan. Beyond final answers: Evaluating large language models for math tutoring, 2025. URL https://arxiv.org/abs/2503.16460.

juvi21. Competitive-coding-benchmark. https://github.com/juvi21/Competitive-Coding-Benchmark/, 2024.

Seyed Mehran Kazemi et al. Big-bench extra hard. *arXiv preprint arXiv:2502.19187*, 2025.

Samuel Kessler, Menglin Xia, Daniel Madrigal Diaz, Dongge Han, Helia Heshemi, Saravan Rajmohan, Victor Ruehle, and Jordan T. Ash. Towards active synthetic data generation for finetuning language models, 2026. URL https://arxiv.org/abs/2512.00884.

Martin Klissarov, Jonathan Cook, Diego Antognini, Hao Sun, Jingling Li, Natasha Jaques, Claudiu Musat, and Edward Grefenstette. Improving interactive in-context learning from natural language feedback, 2026. URL https://arxiv.org/abs/2602.16066.

Philippe Laban, Hiroaki Hayashi, Yingbo Zhou, and Jennifer Neville. Llms get lost in multi-turn conversation, 2025. URL https://arxiv.org/abs/2505.06120.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Thomas Mesnard, Johan Ferret, Kellie Lu, Colton Bishop, Ethan Hall, Victor Carbune, Abhinav Rastogi, and Sushant Prakash. Rlaif vs. rlhf: Scaling reinforcement learning from human feedback with ai feedback, 2024. URL https://arxiv.org/abs/2309.00267.

Xiaoyuan Li, Wenjie Wang, Moxin Li, Junrong Guo, Yang Zhang, and Fuli Feng. Evaluating mathematical reasoning of large language models: A focus on error identification and correction. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.673. URL https://aclanthology.org/2024.findings-acl.673/.

Kevin Lu and Thinking Machines Lab. On-policy distillation. *Thinking Machines Lab: Connectionism*, 2025. doi: 10.64434/tml.20251026. https://thinkingmachines.ai/blog/on-policy-distillation.

OpenAI. Gpt-oss: Open-weight reasoning models. Technical report, OpenAI, 2025. URL https://openai.com/research/gpt-oss.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2024. URL https://arxiv.org/abs/2305.18290.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023. URL https://arxiv.org/abs/2303.11366.

Qwen Team. Qwen 3.5 technical report: Advancements in multilingual and reasoning capabilities. *arXiv preprint arXiv:2602.09123*, 2026.

Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. Position: Will we run out of data? Limits of LLM scaling based on human-generated data, 2024. URL https://arxiv.org/abs/2211.04325.

Xingyao Wang, Zihan Wang, Jiateng Liu, Yangyi Chen, Lifan Yuan, Hao Peng, and Heng Ji. Mint: Evaluating llms in multi-turn interaction with tools and language feedback, 2024. URL https://arxiv.org/abs/2309.10691.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models, 2023. URL https://arxiv.org/abs/2203.11171.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.

Wanqiao Xu, Allen Nie, Ruijie Zheng, Aditya Modi, Adith Swaminathan, and Ching-An Cheng. Provably learning from language feedback, 2025. URL https://arxiv.org/abs/2506.10341.

A Experimental Setup Details

A.1 Protocol Definition

Our evaluation protocol follows the didactic feedback loop of Klissarov et al. [2026], but uses it only as a zero-shot evaluation environment. A single episode contains a task instance, a student model, a teacher model, and a maximum number of student attempts. The first student message contains the task but no teacher feedback. After each student attempt, the environment evaluates correctness. Correct attempts terminate the episode immediately. Incorrect attempts trigger a teacher turn unless the attempt budget has already been exhausted.

Let x be the public task prompt, z be optional privileged task information, and h_t be the prompt-visible interaction history before the t-th student attempt. The student generates

$$a_t \sim \pi_S(\cdot \mid x, h_t, f_{t-1}),$$

where f_{t-1} is empty on the first turn. The verifier returns $c_t = 1$ if a_t is correct. If $c_t = 0$ and $t < K_{\max}$, the teacher generates

$$f_t \sim \pi_T(\cdot \mid x, a_t, h_t, z_T),$$

where z_T is determined by the teacher-information condition. In the dense model matrix, z_T excludes the final answer and full solution for math and BBEH, and excludes the trusted reference solution for code. In code, the teacher can still see execution feedback unless explicitly ablated.

The teacher is always instructed to provide diagnostic feedback rather than the final answer. In the main prompt style, the teacher may reason in `<think>` tags, but the feedback passed to the student is parsed from the final `<feedback>` block. If parsing fails, the raw teacher text is passed through.

History truncation. The implementation maintains separate student-side and teacher-side chat histories. The parameter `max_history` controls how many completed interaction turns are kept in the prompt. When the history window is full, older refinement turns are removed before adding the next user message. The original user prompt containing the task description is preserved, so every later student or teacher call still has the problem context and the chat remains a natural continuation of the same task.

A.2 Run Configuration

Environment	Episodes	Teacher privileged information	Extra teacher context
Omni-MATH	512	none	none
Codeforces	512	none	structured execution feedback
BBEH Linguini	300	none	none
ARC-AGI1	400	none	structured ARC feedback

Table 2: **Current dense-matrix run settings.**

The current dense-matrix runs use the following common settings:

- Agent: `hierarchical_separate`, with separate vLLM actor instances for student and teacher.
- Interaction budget: $K_{\max} = 10$ student attempts.
- History: `max_history=1`.
- Student sampling: temperature 0.7, top-$p = 0.95$, max tokens 8192.
- Teacher sampling: temperature 1.0, top-$p = 0.95$, max tokens 8192.
- Context length: 32768 tokens.
- Max output tokens: 8192 tokens.
- Number of workers: 16 evaluation workers.

ARC-AGI uses max output tokens 16000 for both student and teacher because the task JSON and output grids can be long.

A.3 Model Set

The dense matrix uses four model families totaling to thirteen model identifiers:

- Gemma4:
  - `gemma-4-E2B-it`,
  - `gemma-4-E4B-it`,
  - `gemma-4-26B-A4B-it`,
  - `gemma-4-31B-it`,
- Qwen3.5:
  - `Qwen3.5-2B`,
  - `Qwen3.5-4B`,
  - `Qwen3.5-9B`,
  - `Qwen3.5-27B`,
- Llama:
  - `Llama-3.2-1B-Instruct`,
  - `Llama-3.2-3B-Instruct`,
  - `Llama-3.1-8B-Instruct`,
- GPT-OSS:
  - `gpt-oss-20b`,
  - `gpt-oss-120b`.

Each model appears once as the student and once as the teacher, for 169 pairs per dense-matrix environment.

A.4 Datasets and Verifiers

Omni-MATH. Each example contains a problem, final answer, ground-truth solution, and source. The student is asked to end with one boxed final answer. Correctness is evaluated by two math answer checkers: a PRIME-style normalization and symbolic equivalence checker, followed by a `math_verify`-based checker. The environment returns reward 1 if one checker accepts the answer.

Codeforces. Each example contains a prompt, optional starter code, trusted reference solution, test specification, source, and difficulty metadata. The student returns a complete Python solution in a fenced code block. Candidate code is run in a subprocess sandbox on at most 16 test cases. A task is solved only when all tests pass. Current timeouts use dataset-aware scaling: per-case timeout defaults to 10 seconds, is multiplied by 10 when a dataset time limit is available, and is clamped between 5 and 20 seconds; the full-task timeout is derived from the case timeout and number of cases.

BBEH (Linguini). Each example contains a task name, problem text, reference answer, source, and deterministic problem id derived from the task, problem, and answer. We filter the BBEH dataset to the Linguini task. Correctness is evaluated by lowercasing and normalizing the prediction and reference, stripping simple LaTeX wrappers, handling common answer prefixes, and applying a small set of fuzzy matches for parenthesized choices, numeric strings, quotes, brackets, and trailing question marks. The environment returns reward 1 if the normalized prediction matches the reference under this evaluator.

ARC-AGI1. Each example contains a task id, training input/output grid pairs, test input grids, and hidden target output grids used only by the verifier. The student must return a JSON object, using the schema `{"outputs": [[[...]], ...]}`. Correctness is evaluated by parsing the JSON, validating that each predicted output is a well-formed ARC grid with the expected number of test outputs, and comparing the predicted grids to the hidden target grids. The environment returns reward 1 only when all test outputs match exactly. Structured feedback reports parsing failures, shape errors, or grid mismatch diagnostics, truncated to 2000 characters.

A.5 Teacher Information Conditions

We use the following teacher reference modes:

- `none`: the teacher receives no ground-truth answer or solution.
- `answer`: the teacher receives only the final answer or reference target.
- `solution`: the teacher receives the full ground-truth solution or trusted reference solution.

For code, reference mode is crossed with execution-context mode:

- `structured`: number of tests passed, pass rate, timeout status, and first-failure details when available.
- `feedback`: the environment's compact textual execution feedback.
- `traceback`: only stderr or traceback text when available.
- `none`: no execution-side signal beyond the student code and problem prompt.

A.6 Metrics from Episode Logs

Each completed episode log contains at least `solved`, `attempts_used`, `max_turns`, and, when available, `problem_id`. For each cutoff k, the postprocessing code marks an episode as solved by k if `solved=1` and `attempts_used <= k`. If multiple episodes share a problem id, the code averages within problem first and then averages across problems. Otherwise each episode is treated as its own problem.

The main reported metrics are:

$$\mathrm{acc@}k = \frac{1}{N}\sum_i \mathbf{1}[\text{problem } i \text{ solved by } k] \tag{1}$$

$$\mathrm{gain@}K = \mathrm{acc@}K - \mathrm{acc@}1, \tag{2}$$

$$\mathrm{ngain@}K = \begin{cases} (\mathrm{acc@}K - \mathrm{acc@}1)/(1 - \mathrm{acc@}1), & \mathrm{acc@}1 < 1 \\ 0, & \mathrm{acc@}1 = 1 \end{cases} \tag{3}$$

$$\mathrm{auc} = \frac{1}{K_{\max}}\sum_{k=1}^{K_{\max}} \mathrm{acc@}k \tag{4}$$

We also report $\mathrm{acc@}2 - \mathrm{acc@}1$ because a very large first refinement jump can indicate a formatting or first-turn protocol artifact rather than steady feedback use.

A.7 Prompt Templates

This section records the prompt templates used by the current implementation. Braced names such as `{problem}` denote task-specific fields inserted at runtime.

Shared teacher system prompt.

```
You are a careful tutor. Read the student's latest attempt, identify the most
important mistake, and give targeted feedback without revealing the final answer.
```

Shared teacher output format.

```
Pinpoint the step in which I am making a mistake.
Provide the most informative piece of information for me to succeed on the next
try, without telling me the final answer.
You may reason about the mistake first inside <think>...</think> tags.
```

```
Your final feedback message must appear exactly once inside
<feedback>...</feedback> tags.

Return only:
<feedback>
your feedback
</feedback>
```

Omni-MATH student prompts.

```
System:
You are a mathematician solving competition-level math problems.

Initial user prompt:
Problem:
{problem}

Solve the problem carefully.

You may reason in <think>...</think>. End with exactly one final line in this
format:
\boxed{your answer}

Refinement user prompt, feedback condition:
Feedback:
{teacher_feedback}

Revise your previous solution so it uses the feedback.

You may reason in <think>...</think>. End with exactly one final line in this
format:
\boxed{your answer}

Refinement user prompt, self-refinement/no-feedback condition:
Revise your previous solution and produce a corrected solution.

You may reason in <think>...</think>. End with exactly one final line in this
format:
\boxed{your answer}
```

Omni-MATH teacher prompts.

```
Initial teacher user prompt:
I am trying to solve this question:
{problem}

Here is my current attempt:
{previous_attempt}

{reference_context}

{teacher_output_rules}

Follow-up teacher user prompt:
Here is my current attempt:
{previous_attempt}

{teacher_output_rules}
```

The reference context is one of:

```
You do not have access to the ground truth solution or final answer.

Here is the ground-truth final answer:
{answer}

Here is the ground-truth solution:
{solution}
```

A.8 Example of interaction between student and teacher

This section provides an example of a complete dialogue between teacher gpt-oss 120B and student Gemma-4-31B. To make it more parsable outputs of student have been summarized. Teacher in this trajectory provided student with necessary refinement to complete the task.

Agent Interaction: Student-Teacher Correction

Student [Step 0] — Reasoning & Initial Hypothesis:
Analyzes the input grids and identifies the bounding box of the active shape (color 4). Hypothesizes that the target output is a 3×3 grid of blocks forming a cross shape. Attempts to deduce the geometric transformations for each arm of the cross, but incorrectly formulates the policy: assuming the right arm is a horizontal reflection and the bottom arm is a vertical reflection, rather than pure rotations.

Student [Step 0] — Action:
Submits a 17×17 output grid. (*Result: Incorrect, Reward: 0.0*)

Teacher [Step 1] — Feedback:
"You correctly noticed that the output is built from four copies of the input's colored shape arranged in a cross, but the transformations you applied to the arms are not the ones used in the examples. In the training pairs, the top arm is the shape rotated $90°$ clockwise, the left arm is the original shape, the right arm is the shape rotated $180°$ (i.e., both axes flipped), and the bottom arm is the shape rotated $270°$ clockwise (or equivalently a $90°$ counter-clockwise rotation).

For non-square shapes, the rotated arms swap height and width, so the top and bottom blocks have the dimensions of the input $W \times H$, while the left and right blocks keep the original $H \times W$. Because you used a horizontal flip... several cells are placed in the wrong positions. Re-examine the transformations: left = original, top = $90°$ cw, right = $180°$, bottom = $270°$ cw, and assemble the output accordingly."

Student [Step 1] — Reasoning & Correction:
Integrates the teacher's feedback to update the transformation policy. Re-evaluates the active bounding box, explicitly noting it is non-square ($H = 5, W = 6$). Calculates the necessary dimensional swaps for the top and bottom arms ($W = 5, H = 6$). Methodically maps the pixel coordinates for S_{90}, S_{180}, and S_{270} applied to the base shape S, and reconstructs the spatial layout of the final grid using the corrected rotational rules.

Student [Step 1] — Action:
Submits the corrected 17×17 output grid. (*Result: Success, Reward: 1.0*)

Table 3: **Most pair-level gain variation is explained by the student.** Metrics are averaged over student-teacher pair cells at $K = 10$. Student FE and teacher FE are single-factor fixed-effect R^2 values for pair-level $\mathrm{gain} = \mathrm{acc@10} - \mathrm{acc@1}$; the last column is the incremental R^2 from adding teacher identity after student identity.

Environment	acc@1	acc@10	gain	auc	Student FE R^2	Teacher FE R^2	ΔR^2 teacher|student
Omni-MATH	0.238	0.521	0.283	0.452	0.858	0.093	0.093
Codeforces	0.332	0.604	0.272	0.544	0.827	0.096	0.096
BBEH Linguini	0.037	0.144	0.107	0.111	0.799	0.132	0.132
ARC-AGI1	0.101	0.278	0.178	0.219	0.963	0.012	0.012

Table 4: **Teacher effects are smaller globally but still consequential for a fixed student.** For each student, we compute the gap between the best and worst teacher at $K = 10$, then average these within-student gaps over students.

Environment	Mean worst auc	Mean best auc	Mean Δauc	Mean Δacc@10
Omni-MATH	0.395	0.511	0.116	0.158
Codeforces	0.483	0.608	0.125	0.163
BBEH Linguini	0.079	0.163	0.085	0.117
ARC-AGI1	0.191	0.256	0.065	0.090

B Additional Results

The initial sections present detailed dense-matrix tables and performance visualizations for all model pairings across the tested environments, including Omni-MATH, Codeforces, BBEH Linguini, and ARC-AGI1. They do it for all types of our metrices we used: accuracy, cumulative accuracy, and performance gains.

Subsequent sections focus on specific ablation studies and deeper mechanistic insights, In them we are using gemma 4 dense matrices. We isolate the impact of relative teacher strength, information access and maximum history, providing a thorough analysis of the variables that govern reasoning improvements in multi-turn feedback settings.

B.1 Supplementary Dense-Matrix Tables

Table 5: **Teacher task competence is only partially predictive of interactive teacher competence.** We rank teachers by empirical acc@1 when acting as a student in the same environment, using it as a noisy proxy for single-turn task competence. Spearman correlations are computed within each fixed student between this teacher-rank proxy and interactive auc. Inversion rate is the fraction of comparable teacher pairs where the lower-acc@1 teacher gives higher interactive auc. Values are means over fixed-student groups $\pm$ one standard error.

Environment	Mean Spearman ρ	Inversion rate
Omni-MATH	0.21 ± 0.06	$41.4\% \pm 1.9\%$
Codeforces	0.53 ± 0.04	$29.9\% \pm 1.7\%$
BBEH Linguini	0.44 ± 0.07	$33.4\% \pm 2.5\%$
ARC-AGI1	0.33 ± 0.11	$35.4\% \pm 4.0\%$

B.2 Supplementary Plasticity Table

Table 6: **High acc@1 can lose to better feedback adherence.** The table compares GPT-OSS-120B and Gemma4-31B, averaging metrics over teachers. GPT-OSS-120B often starts from a stronger first-turn baseline, but Gemma4-31B achieves higher interactive auc across environments because it gains more from feedback. This supports the claim that feedback adherence is distinct from initial task competence. Grey terms report standard errors over teachers.

	GPT-OSS-120B				Gemma4-31B			
Environment	acc@1	acc@10	gain	auc	acc@1	acc@10	gain	auc
Omni-MATH	0.555 ±0.002	0.648 ±0.007	0.093 ±0.008	0.624 ±0.006	0.439 ±0.002	0.720 ±0.002	0.281 ±0.004	0.679 ±0.002
Codeforces	0.683 ±0.002	0.813 ±0.009	0.129 ±0.008	0.782 ±0.008	0.575 ±0.003	0.911 ±0.003	0.336 ±0.004	0.864 ±0.003
BBEH Linguini	0.137 ±0.005	0.254 ±0.015	0.116 ±0.013	0.213 ±0.012	0.095 ±0.002	0.368 ±0.008	0.273 ±0.008	0.303 ±0.006
ARC-AGI1	0.282 ±0.003	0.536 ±0.011	0.254 ±0.010	0.453 ±0.009	0.528 ±0.003	0.728 ±0.004	0.200 ±0.005	0.670 ±0.003

B.3 Supplementary Model-Matrix Figures

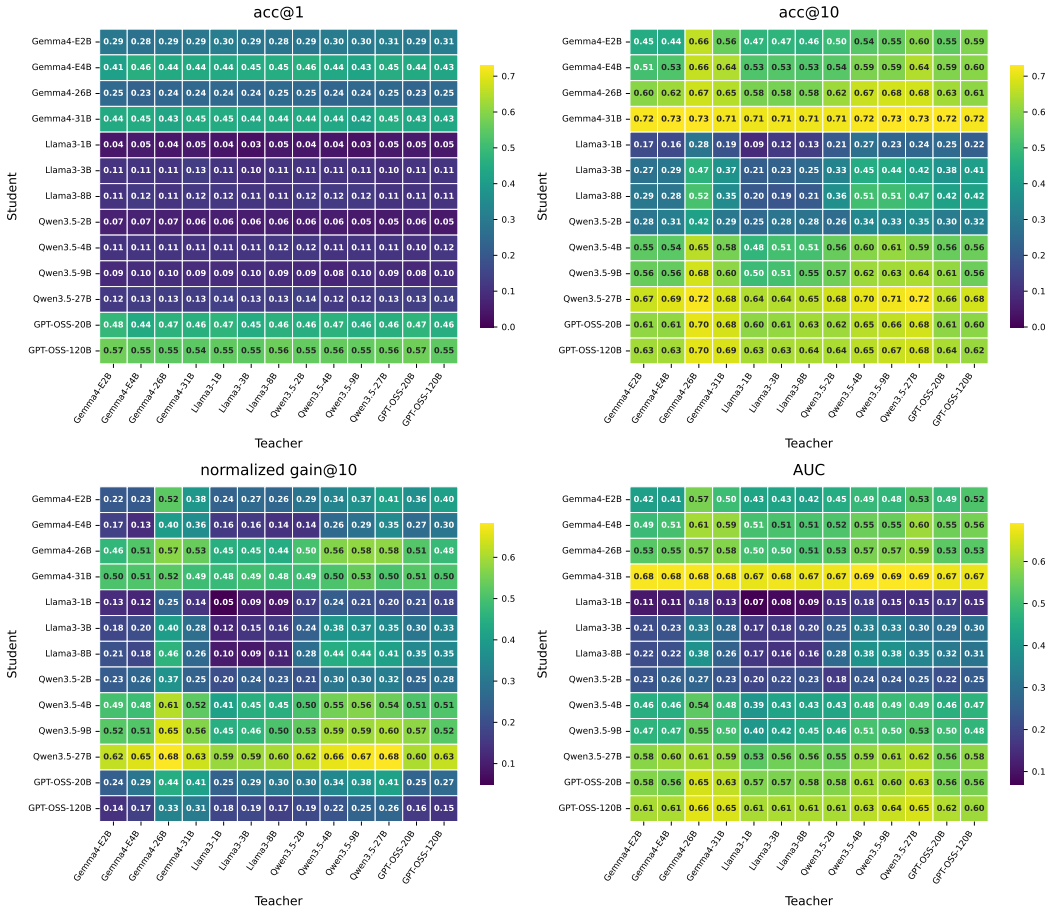

Figure 8: **Omni-MATH dense model matrices**. Rows are students and columns are teachers. We annotate all four metrics: acc@1, acc@10, gain, and cumulative accuracy auc.

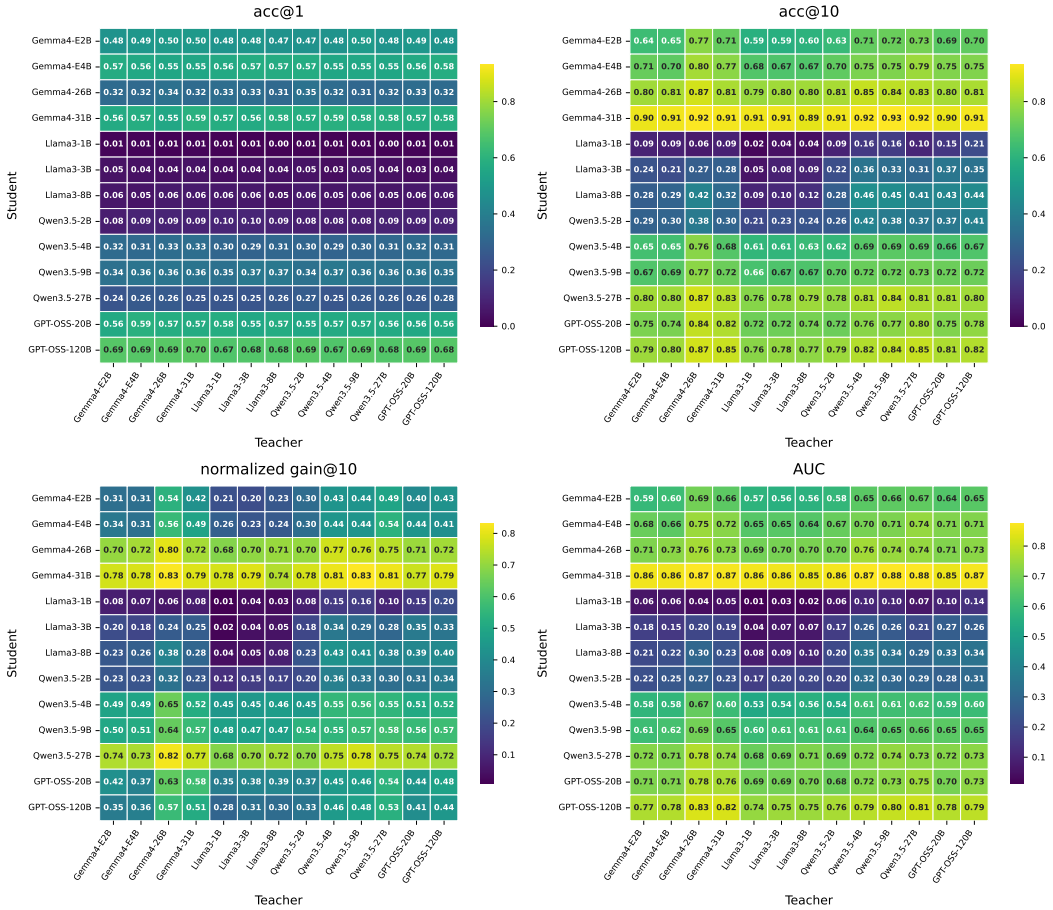

Figure 9: **Codeforces dense model matrices** Rows are students and columns are teachers. We annotate all four metrics: acc@1, acc@10, gain, and cumulative accuracy auc. *Note: Results for Gemma4-26B/Qwen3.5-4B are omitted due to technical failure during computation.*

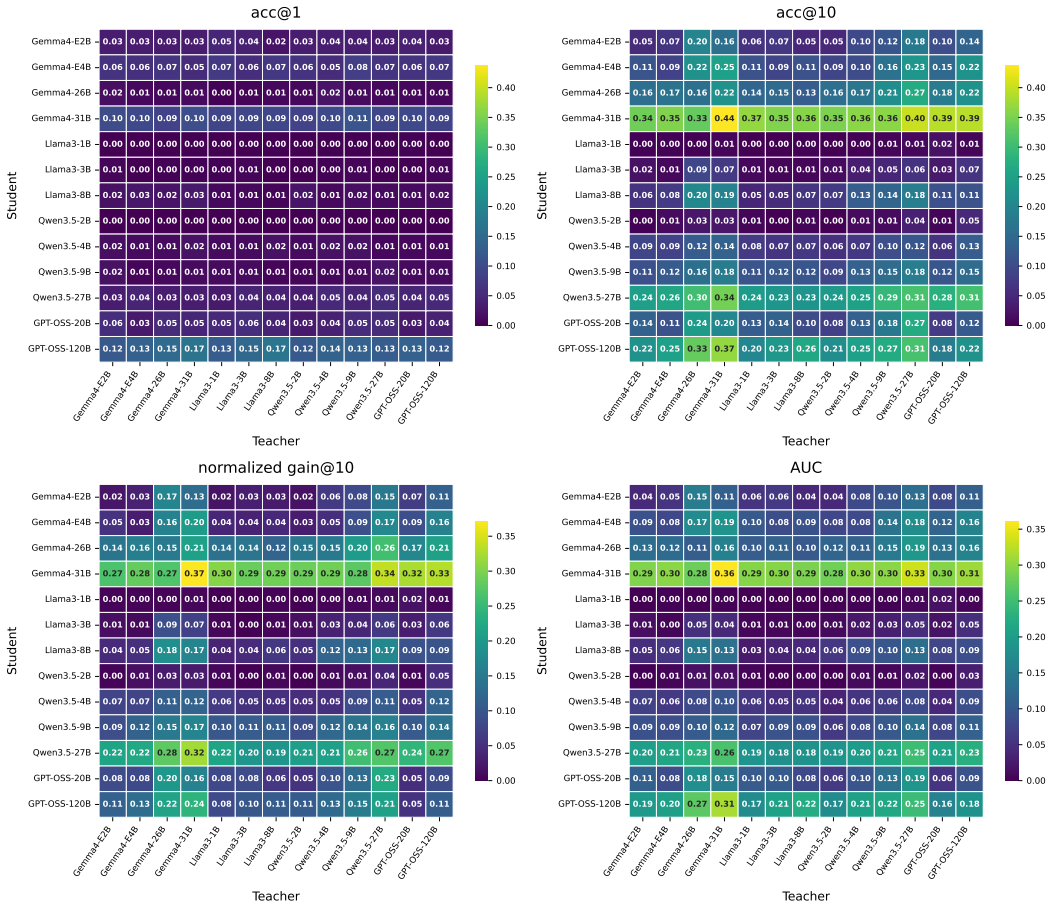

Figure 10: **BBEH Linguini dense model matrices**. Rows are students and columns are teachers. We annotate all four metrics: acc@1, acc@10, gain, and cumulative accuracy auc.

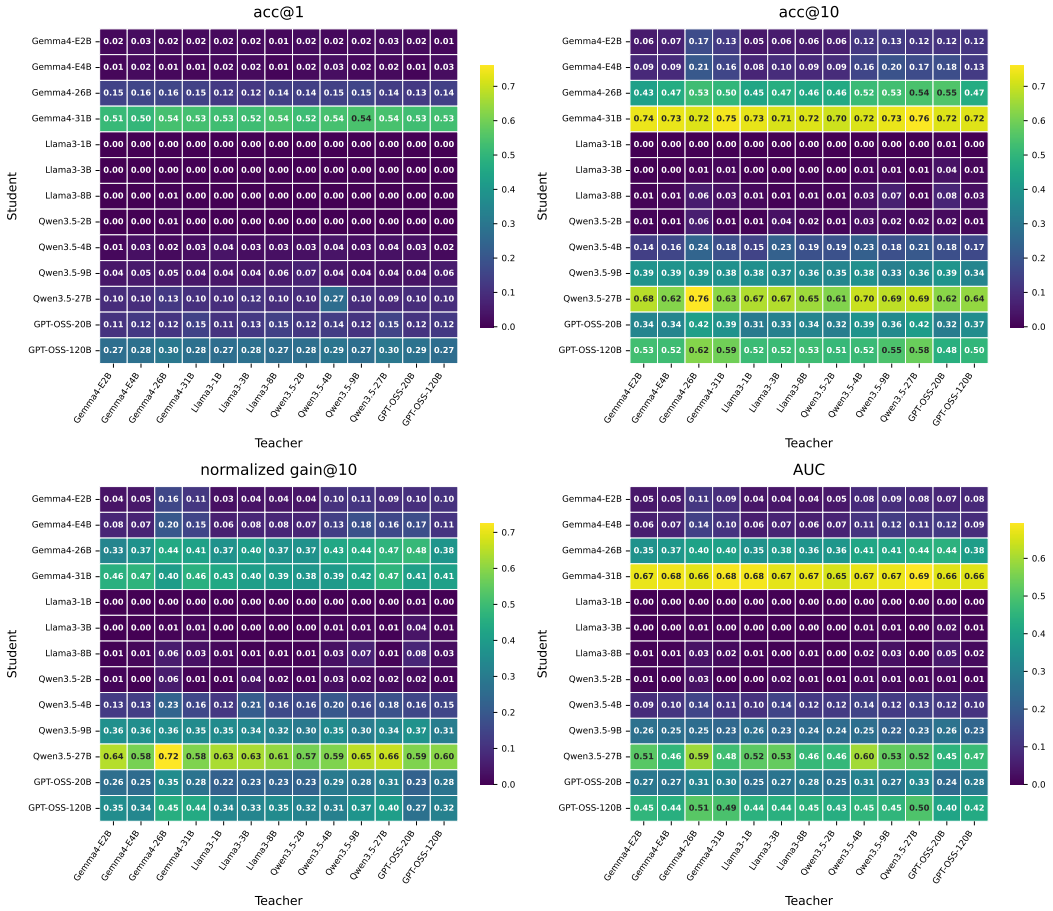

Figure 11: **ARC-AGI1 dense model matrices**. Rows are students and columns are teachers. We annotate all four metrics: acc@1, acc@10, gain, and cumulative accuracy auc.

B.4 Supplementary Trajectory Figure

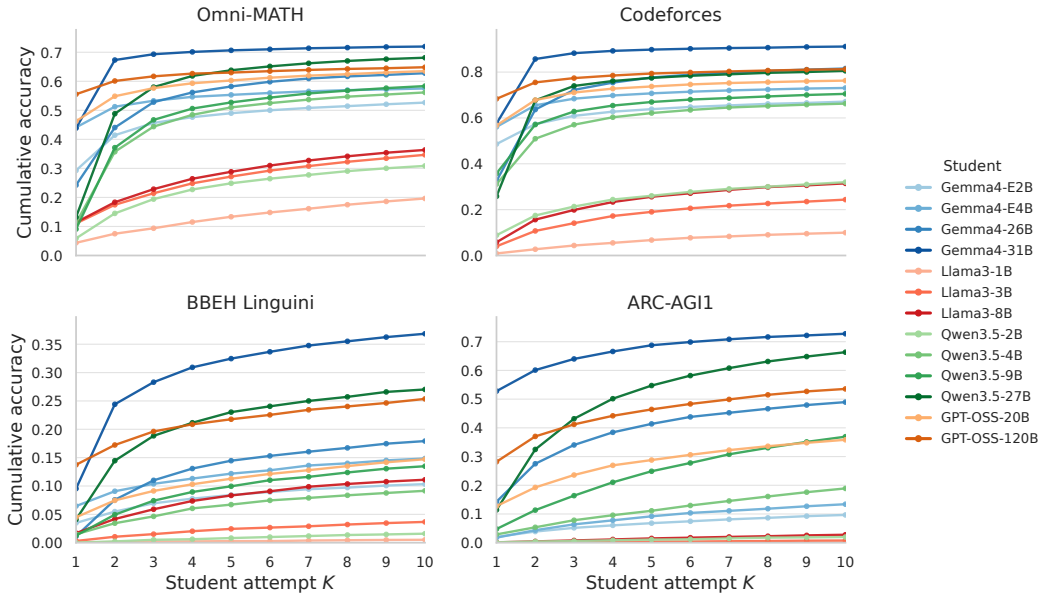

Figure 12: **Cumulative accuracy curves are separated primarily by the student model.** Each panel shows one environment, and each curve averages across teachers for one student. The broad vertical separation between students is larger than the within-student variation induced by changing teachers.

B.5 Supplementary Gain-by-Role Figures

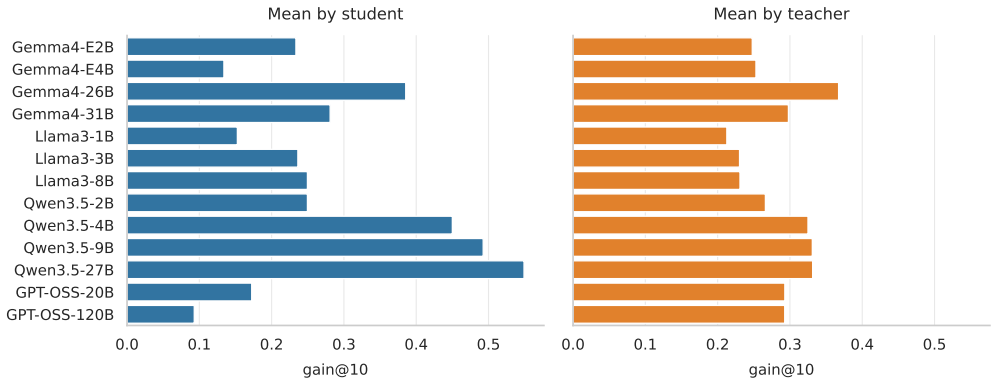

Figure 13: Omni-MATH. We decompose gain by student(left) and teacher(right) for each model we compute mean gain at $K = 10$ across all teachers(left) and students(right).

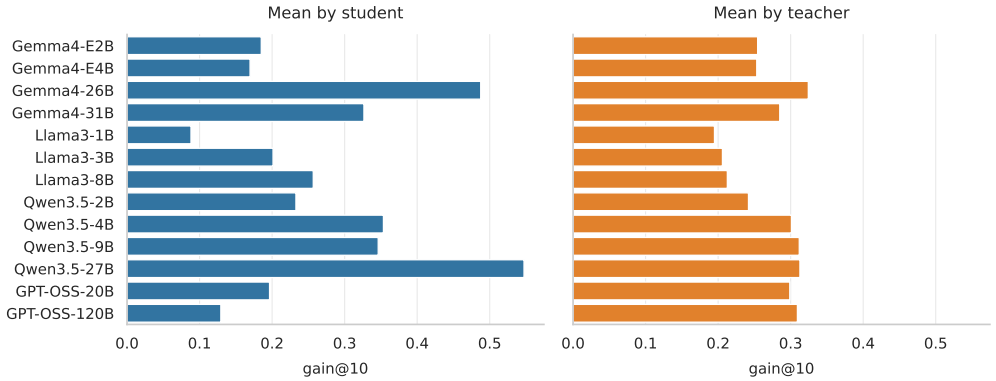

Figure 14: Codeforces. We decompose gain by student(left) and teacher(right) for each model we compute mean gain at $K = 10$ across all teachers(left) and students(right).

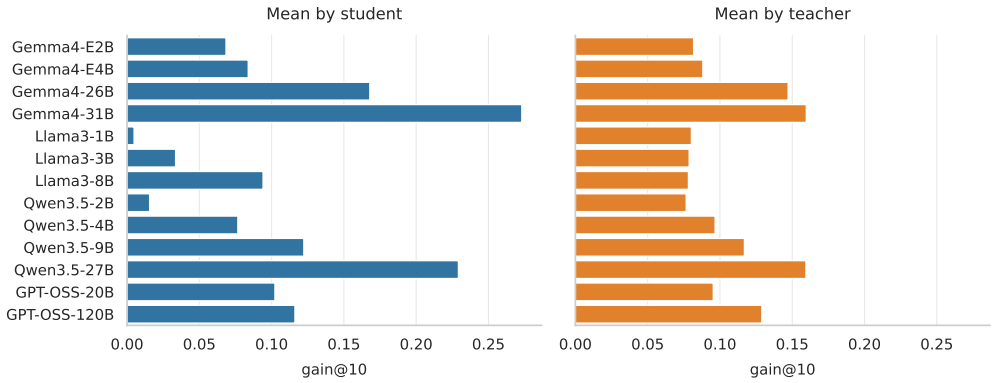

Figure 15: BBEH Linguini. We decompose gain by student(left) and teacher(right) for each model we compute mean gain at $K = 10$ across all teachers(left) and students(right).

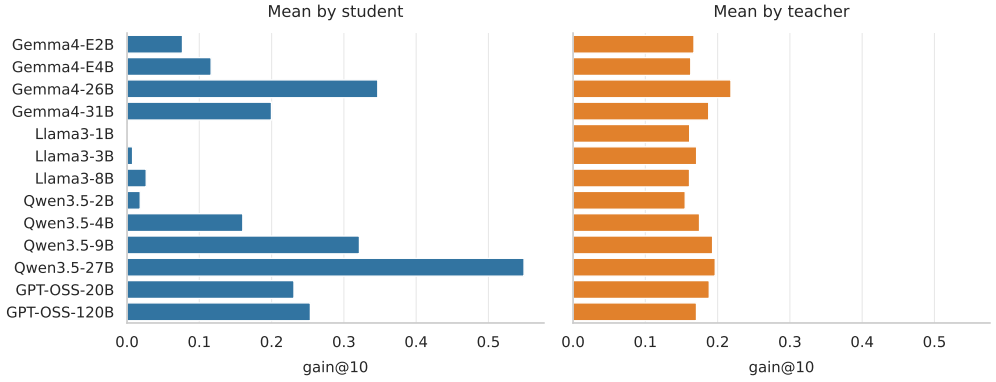

Figure 16: ARC-AGI1. We decompose gain by student (left) and teacher (right) for each model we compute mean gain at $K = 10$ across all teachers(left) and students(right).

B.6 Supplementary Gemma4 Ablation Figures

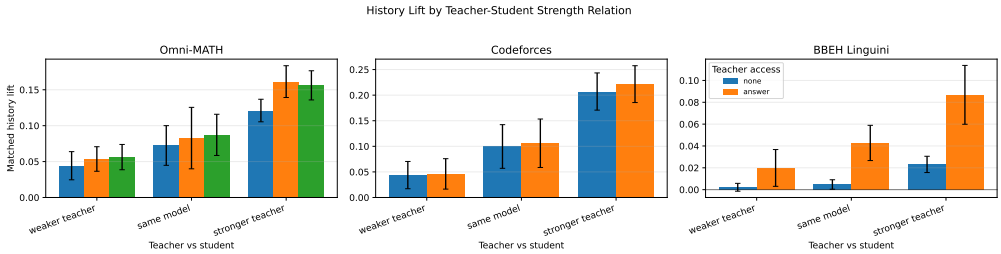

Figure 17: **History lift by relative teacher capability.** Matched auc lift from history 1 to history 5 is largest when the teacher is stronger than the student.

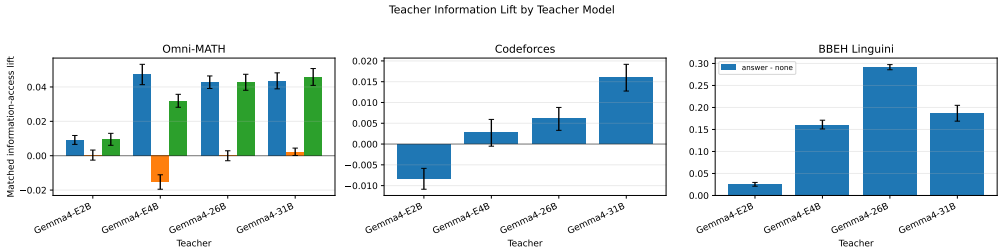

Figure 18: **Information-access lift by teacher.** Matched information-access gains vary by teacher and environment, rather than appearing as a uniform benefit of revealing the answer or solution.

B.7 Feedback-Control Tables

Table 7: **Per-student controls disentangling interaction, feedback, and feedback quality.** Cells report acc@10 with gain@10 in parentheses, in percentage points. Self-refinement is the generic retry condition, self-feedback is the dense-matrix diagonal, and best feedback selects the highest-acc@10 teacher available for the student and environment.

Environment	Student	Self-refinement	Self-feedback	Best feedback	Best teacher
Omni-MATH	Gemma4-E2B	48.4 (+22.3)	44.9 (+15.8)	65.8 (+37.3)	Gemma4-26B
Omni-MATH	Gemma4-E4B	56.6 (+11.5)	52.9 (+7.0)	66.4 (+22.1)	Gemma4-26B
Omni-MATH	Gemma4-26B	49.8 (+26.4)	67.2 (+43.2)	68.0 (+43.8)	Qwen3.5-9B
Omni-MATH	Gemma4-31B	66.0 (+23.6)	71.5 (+27.0)	72.9 (+31.1)	Qwen3.5-9B
Omni-MATH	Llama-1B	11.1 (+7.4)	8.8 (+4.7)	28.3 (+24.0)	Gemma4-26B
Omni-MATH	Llama-3B	22.9 (+11.3)	22.9 (+13.3)	46.7 (+35.4)	Gemma4-26B
Omni-MATH	Llama-8B	24.6 (+15.2)	20.7 (+10.0)	51.8 (+40.8)	Gemma4-26B
Omni-MATH	Qwen3.5-2B	22.3 (+16.4)	25.6 (+19.9)	41.6 (+35.0)	Gemma4-26B
Omni-MATH	Qwen3.5-4B	40.0 (+28.9)	59.6 (+48.8)	65.4 (+54.3)	Gemma4-26B
Omni-MATH	Qwen3.5-9B	34.4 (+24.6)	62.9 (+53.1)	68.0 (+58.4)	Gemma4-26B
Omni-MATH	Qwen3.5-27B	61.7 (+49.6)	72.5 (+59.0)	72.5 (+59.0)	Qwen3.5-27B
Omni-MATH	GPT-OSS-20B	65.1 (+17.6)	60.5 (+13.3)	70.1 (+23.4)	Gemma4-26B
Omni-MATH	GPT-OSS-120B	67.8 (+14.3)	61.5 (+6.6)	69.7 (+15.0)	Gemma4-26B
Codeforces	Gemma4-E2B	59.0 (+9.8)	63.9 (+16.0)	77.0 (+26.6)	Gemma4-26B
Codeforces	Gemma4-E4B	66.8 (+12.5)	69.7 (+13.9)	80.3 (+25.4)	Gemma4-26B
Codeforces	Gemma4-26B	75.8 (+44.1)	86.7 (+52.5)	86.7 (+52.5)	Gemma4-26B
Codeforces	Gemma4-31B	84.2 (+27.1)	91.4 (+32.4)	93.0 (+35.2)	Qwen3.5-9B
Codeforces	Llama-1B	1.8 (+1.0)	1.6 (+0.8)	21.3 (+20.1)	GPT-OSS-120B
Codeforces	Llama-3B	13.7 (+8.5)	8.4 (+4.1)	36.5 (+33.6)	GPT-OSS-20B
Codeforces	Llama-8B	14.5 (+8.0)	12.5 (+7.8)	45.9 (+40.6)	Qwen3.5-4B
Codeforces	Qwen3.5-2B	19.5 (+10.5)	26.4 (+18.2)	41.6 (+33.4)	Qwen3.5-4B
Codeforces	Qwen3.5-4B	65.3 (+28.0)	68.6 (+39.1)	76.4 (+43.4)	Gemma4-26B
Codeforces	Qwen3.5-9B	64.6 (+22.1)	72.3 (+36.3)	77.0 (+40.8)	Gemma4-26B
Codeforces	Qwen3.5-27B	71.9 (+39.0)	81.4 (+55.3)	86.7 (+60.4)	Gemma4-26B
Codeforces	GPT-OSS-20B	72.3 (+11.3)	75.2 (+19.5)	84.4 (+27.0)	Gemma4-26B
Codeforces	GPT-OSS-120B	77.5 (+9.4)	82.4 (+14.1)	86.5 (+17.8)	Gemma4-26B
BBEH Linguini	Gemma4-E2B	11.0 (+7.0)	5.3 (+2.0)	19.7 (+16.3)	Gemma4-26B
BBEH Linguini	Gemma4-E4B	16.7 (+9.0)	8.7 (+2.7)	24.7 (+19.3)	Gemma4-31B
BBEH Linguini	Gemma4-26B	10.3 (+9.7)	16.0 (+15.0)	26.7 (+25.7)	Qwen3.5-27B
BBEH Linguini	Gemma4-31B	32.1 (+19.3)	43.7 (+33.3)	43.7 (+33.3)	Gemma4-31B
BBEH Linguini	Llama-1B	0.0 (+0.0)	0.0 (+0.0)	2.0 (+1.7)	GPT-OSS-20B
BBEH Linguini	Llama-3B	1.7 (+1.3)	0.7 (+0.0)	9.3 (+9.0)	Gemma4-26B
BBEH Linguini	Llama-8B	6.0 (+4.0)	6.7 (+6.0)	20.3 (+18.0)	Gemma4-26B
BBEH Linguini	Qwen3.5-2B	0.3 (+0.3)	0.0 (+0.0)	5.0 (+5.0)	GPT-OSS-120B
BBEH Linguini	Qwen3.5-4B	5.3 (+3.7)	7.0 (+5.0)	13.7 (+11.7)	Gemma4-31B
BBEH Linguini	Qwen3.5-9B	7.3 (+5.7)	15.3 (+14.0)	18.3 (+16.0)	Qwen3.5-27B
BBEH Linguini	Qwen3.5-27B	22.7 (+19.0)	31.0 (+25.7)	34.3 (+31.0)	Gemma4-31B
BBEH Linguini	GPT-OSS-20B	15.0 (+8.0)	7.7 (+4.7)	26.7 (+21.3)	Qwen3.5-27B
BBEH Linguini	GPT-OSS-120B	27.7 (+12.3)	22.0 (+9.7)	36.7 (+20.0)	Gemma4-31B
ARC-AGI1	Gemma4-E2B	5.8 (+3.8)	5.5 (+3.8)	17.2 (+15.2)	Gemma4-26B
ARC-AGI1	Gemma4-E4B	8.8 (+7.0)	9.0 (+6.8)	21.0 (+19.5)	Gemma4-26B
ARC-AGI1	Gemma4-26B	38.0 (+24.8)	52.5 (+36.8)	55.0 (+42.3)	GPT-OSS-20B
ARC-AGI1	Gemma4-31B	21.5 (+11.0)	74.8 (+21.5)	75.5 (+21.8)	Qwen3.5-27B
ARC-AGI1	Llama-1B	0.0 (+0.0)	0.2 (+0.2)	0.8 (+0.8)	GPT-OSS-20B
ARC-AGI1	Llama-3B	0.0 (+0.0)	0.0 (+0.0)	3.8 (+3.8)	GPT-OSS-20B
ARC-AGI1	Llama-8B	1.0 (+0.8)	1.0 (+0.8)	7.8 (+7.8)	GPT-OSS-20B
ARC-AGI1	Qwen3.5-2B	0.5 (+0.5)	1.0 (+1.0)	6.2 (+5.5)	Gemma4-26B
ARC-AGI1	Qwen3.5-4B	10.5 (+8.5)	23.0 (+19.0)	24.5 (+22.5)	Gemma4-26B
ARC-AGI1	Qwen3.5-9B	28.0 (+24.3)	32.5 (+29.0)	39.2 (+35.5)	GPT-OSS-20B
ARC-AGI1	Qwen3.5-27B	36.8 (+27.2)	69.2 (+60.5)	76.0 (+63.3)	Gemma4-26B
ARC-AGI1	GPT-OSS-20B	33.8 (+22.3)	31.5 (+20.0)	42.5 (+30.8)	Gemma4-26B
ARC-AGI1	GPT-OSS-120B	52.2 (+20.5)	50.0 (+23.0)	62.0 (+31.5)	Gemma4-26B