Learning from a Thoughtful Teacher: Adaptive On-Policy Self-Distillation for Mathematical Reasoning

Anonymous authors
Paper under double-blind review

Abstract

On-policy self-distillation (OPSD) trains a question-only student with token-level feedback from a teacher given training-only privileged information (PI). OPSD therefore provides dense, on-policy supervision, and is free of a larger external teacher, but its effectiveness rests on how PI is designed and utilized. Our preliminary diagnostics suggest a significant gap between teacher utility and student learnability, where a small fraction of high-disagreement tokens dominate the distillation signal, and short teacher continuations at these positions further expose more explicit PI leakage than transferable correction cues, indicating a strong intent on injecting PI-conditioned shortcuts. We propose Adaptive On-Policy Self-Distillation (AOPSD), which adapts what information the teacher receives and how strongly its feedback influences learning. AOPSD encodes each solution as a reasoning DAG, orders problems by the student's evolving capability, and reveals only the affordable subgraph and its next frontier as PI. For high-disagreement tokens, AOPSD utilizes short teacher continuations as probes to encourage useful guidance while mitigating PI-conditioned shortcuts among teacher supervisions. On HMMT25, AIME24, AIME25, and BRUMo25, AOPSD achieves 72.5% Pass@8, which is 6.7 percentage points above OPSD and 4.2 above the strongest competing baseline while reducing 15 percentage points of training time at lower cost.

1 Introduction

Mathematical problem solving is a demanding test of language-model reasoning: a solver must preserve constraints, choose among plausible routes, and revise intermediate claims over long trajectories. Teaching mathematical reasoning requires more than supplying a solution: the student must learn to make the underlying decisions on its own. Supervised fine-tuning (SFT) supplies dense targets, but these targets come from fixed solution traces rather than the evolving student's attempts (Yue et al., 2024; Toshniwal et al., 2025). Reinforcement learning with verifiable rewards (RLVR) trains on those attempts, but a final-answer reward does not identify which intermediate choices should change (Shao et al., 2024; Guo et al., 2025; Yu et al., 2025). The challenge is to provide dense supervision at the reasoning states the student actually visits.

On-policy distillation addresses this challenge by asking a teacher to score the student's own trajectories (Agarwal et al., 2024; Gu et al., 2024). It provides token-level feedback at sampled prefixes, but requires a suitable teacher with accessible probabilities and a compatible output space (Zhang et al., 2024; Boizard et al., 2025). On-policy self-distillation (OPSD) obtains this teacher from the same base model by giving it a richer context (Zhao et al., 2026): the student sees only the question, while the teacher also receives training-only privileged information (PI), such as a verified solution. PI thus supplies the teacher's informational advantage without requiring a larger external teacher.

This contextual advantage introduces a different teaching problem: information that helps a teacher solve a problem may not help a student learn to solve it unaided. We distinguish *conditional utility*, the benefit of PI while it remains in context, from *question-only transfer*, the distilled student's performance when PI is absent. A teacher can draw directly on a supplied derivation or conclusion, whereas the student must learn to produce the reasoning that supports it.

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.

Our preliminary diagnostics motivate treating utility and transfer as separate criteria (Section 3). Full solutions, plans, and subproblem decompositions significantly improve the teacher's utility, yet the corresponding question-only students score below the reported no-PI reference. At the token level, a small high-disagreement tail accounts for a large share of the absolute teacher–student log-probability gap and dominates the distillation signal. Short teacher continuations at these positions expose both grounded corrections and explicit PI leakage, which motivates looking beyond disagreement magnitude when deciding how to use privileged feedback.

We propose Adaptive On-Policy Self-Distillation (AOPSD), which adapts two teaching decisions to the evolving student: *what information the teacher receives* and *how strongly its feedback influences learning*. Specifically, we first identify *reasoning DAGs as privileged information*, where solutions are compressed into prerequisite-linked mathematical states and can be decomposed without breaking their logic. Then, we design an *adaptive PI disclosure* mechanism. According to the current student's capability, AOPSD adaptively discloses the affordable subgraph as PI for every trained problem, and dynamically reorders the curriculum during the training process. Third, we apply *continuation-based selective distillation* for high-disagreement tokens, where the teacher generates short continuations and the student scores them via negative log-likelihood loss, in order to recognize and follow grounded guidance while mitigating the PI-conditioned bias from the teacher.

Our contributions are:

- We expose a utility–transfer gap in mathematical OPSD: PI can make a substantially stronger conditional teacher while the resulting student underperforms the reported no-PI condition. We localize this failure to a small high-disagreement tail and use short continuations to diagnose whether its dominant preferences correct the problem or expose PI-conditioned behavior.
- We propose Adaptive On-Policy Self-Distillation (AOPSD), which adapts what information the teacher receives and how strongly its feedback influences learning. AOPSD encodes each solution as a reasoning DAG, updates the curriculum based on the student's evolving capability, and reveals only the affordable subgraph and its next frontier as PI. For high-disagreement tokens, AOPSD utilizes short teacher continuations as probes to encourage useful guidance while mitigating PI-conditioned shortcuts among teacher supervision.
- On four competition-mathematics benchmarks, AOPSD achieves the highest final Pass@8 on every benchmark and a 72.5% macro-average, exceeding OPSD by 6.7 pp and the strongest competing baseline by 4.2 pp. Compared to the OPSD baseline, our method has lower theoretical memory usage and reducing 15.6 pp training time in empirical tests.

2 Related Work

Post-training techniques for mathematical reasoning. Mathematical post-training draws supervision from worked solutions and verifiable outcomes. MAmmoTH combines chain-of-thought and program-of-thought rationales (Yue et al., 2024), MetaMath expands mathematical questions through bootstrapping (Yu et al., 2024), and OpenMathInstruct-2 scales synthetic solution data (Toshniwal et al., 2025). These SFT methods provide dense targets on fixed traces. DeepSeekMath, DeepSeek-R1, and DAPO instead optimize sampled attempts using verifiable rewards (Shao et al., 2024; Guo et al., 2025; Yu et al., 2025), but outcome supervision does not directly identify which intermediate decisions should change. AOPSD uses reference solutions as context for dense feedback on the student's own attempts, with disclosure tied to its demonstrated progress.

On-policy distillation. OPD moves teacher supervision onto prefixes the student actually visits. GKD learns from self-generated sequences with flexible divergence objectives (Agarwal et al., 2024), while MiniLLM uses reverse KL to avoid overestimating low-probability teacher outputs (Gu et al., 2024). Entropy-Aware On-Policy Distillation (EOPD) augments reverse-KL OPD with forward KL at positions of high teacher entropy, preserving probability mass across plausible alternatives, and adapting distribution matching to teacher uncertainty(Jin et al., 2026). However, OPD requires direct matching of token distributions from open-source teachers, and different output vocabularies introduce an additional alignment problem (Zhang et al., 2024; Boizard et al., 2025). Therefore, our work mainly focuses on self-distillation and studies how to use the contextual advantage of a teacher with the same initialization.

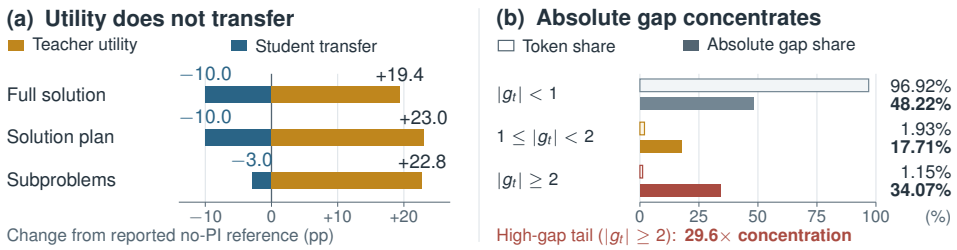


Figure 1: **Preliminary diagnostics of PI utility and transfer.** (a) Change relative to the baseline model: every PI makes the privileged teacher substantially stronger (amber) while the resulting question-only student gets weaker (blue). (b) Token share and share of total absolute sampled-token log gap, grouped by $|g_t|$.

On-policy self-distillation. OPSD conditions the teacher on a reference solution and matches its predictions along student-generated trajectories (Zhao et al., 2026). OP^{2}SD replaces the paired reference with another problem's worked example, showing that context-induced teacher behavior can preserve substantial gains without target-specific solutions (Ichihara et al., 2026). DASH adapts token-level supervision by aggregating local divergences over horizons determined by their variation along the rollout (Hou et al., 2026). These methods establish the value of contextual teaching and adaptive supervision, but do not jointly align the disclosed information and the feedback weights with the student's current reasoning capability. AOPSD couples these decisions: DAG-based progress controls disclosure up to the next dependency frontier and orders the problem curriculum, while teacher continuations scored by the question-only student determine feedback strength.

3 Motivation: When Privileged Utility Fails to Transfer

A useful teacher must do more than solve a problem with extra information: its feedback must help a student that will not receive that information. We distinguish *conditional utility*, the benefit of PI to a solver while it remains in context, from *question-only transfer*, the performance of the student after distillation with PI removed. The following preliminary diagnostics examine this distinction at two levels: final outcomes and local teacher–student disagreements.

3.1 Observation 1: A good privileged solver can be a bad teacher

Our initial comparison uses Qwen3-4B-Instruct-2507 and 725 competition-mathematics problems with verified solutions. We derive three teacher contexts from each solution: the full solution, a solution plan, and a subproblem decomposition. Conditional utility is measured by solving these problems with each context; transfer is assessed by evaluating the distilled question-only students on AIME 2025. Appendix A provides the diagnostic settings and examples of the PI formats.

All three PI formats substantially improve conditional solving in this diagnostic. Pass@8 rises from $554/725$ (76.4%) with the question alone to $695/725$ (95.9%) with the full solution, $721/725$ (99.4%) with a plan, and $719/725$ (99.2%) with subproblems. Yet the corresponding question-only students score 56%, 56%, and 63% after vanilla OPSD training, below the reported 66% baseline (Figure 1a). We further examined students' rollouts on AIME25. A striking phenomenon was that students, seeing only the questions during the test, generated statements like "the given solution", which would only be reasonable given PI as prompts, which suggests that some PI-conditioned behaviors may have been distilled and transferred to the question-only student.

To inspect the feedback behind these outcomes, we measure the teacher-student disagreement along question-only student rollouts. For a problem x, rollout y, and privileged context P, let p denote the student and q^P the PI-conditioned teacher. The sampled-token log-probability gap is

$$g_t(P) = \log q^P(y_t \mid x, y_{<t}) - \log p(y_t \mid x, y_{<t}). \tag{1}$$

In the collected rollouts, only 3.08% of tokens satisfy $|g_t| \ge 1$, but they account for 51.78% of $\sum_t |g_t|$. The $|g_t| \ge 2$ subset contains 1.15% of tokens and 34.07% of the absolute gap (Figure 1b). This concentration motivates closer inspection of high-gap positions.

Table 1: **A short continuation makes high-disagreement intent legible.** Cumulative counts of recognizable correction cues and explicit PI leakage within a total continuation of k tokens.

k	Correction cues		Explicit PI references	
(tokens)	Count	Representative excerpt	Count	Representative excerpt
2	**3**	*an error*	**8**	*original DAG*
4	**7**	*Not quite—*	**70**	*checking the private reasoning DAG*
6	**22**	*the density argument must be flawed*	**153**	*go back to the private reasoning DAG*
8	**31**	*Or maybe we made a mistake?*	**238**	*This matches the reasoning in the DAG*
16	**64**	*Wait—not necessarily.*	**507**	*earlier in the DAG, it says …*

3.2 Observation 2: Teacher introduces shortcuts instead of useful guidance

A large token-level gap provides little context about the behavior that follows it. The sampled token may be an ordinary connector such as `But`. In a separate diagnostic using a reasoning DAG as PI, we select non-overlapping events with $|g_t| \geq 2$, hold the student's token y_t fixed, and let the privileged teacher continue greedily. Each continuation contains y_t and up to $k - 1$ new tokens, exposing a short branch after the original disagreement.

Longer windows reveal both correction cues and explicit references to the private DAG (Table 1). A phrase such as "an error" suggests an attempted correction; a reference to "the private reasoning DAG" explicitly invokes context unavailable to the student, which implies that magnitude alone does not characterize trusted signal from the teacher, and teachers' on-policy guidance is actually more about introducing shortcuts than correcting errors.

4 AOPSD: Teaching at the Learner's Frontier

4.1 Overview

Vanilla OPSD fixes the teacher's privileged context and applies the same distillation rule across sampled tokens. The diagnostics in Section 3 suggest that neither choice should be taken for granted: useful PI may fail to transfer, and a large teacher–student gap alone does not reveal which direction the teacher is encouraging. AOPSD makes both choices relative to the current student rollout. It controls what part of PI the teacher receives and, at high-gap positions, how strongly the teacher's feedback affects the student.

The method links three components (Figure 2). A reasoning DAG represents the reference solution as mathematical checkpoints with explicit prerequisites (Section 4.2). The checkpoints established by the student's rollout determine both the subgraph disclosed to the teacher and the problem's position in a curriculum (Section 4.3). At high-disagreement tokens, a short teacher continuation is scored by the question-only student to set a distillation weight (Section 4.4). Together, these components adapt the teacher's privileged view and feedback strength to the student's current state.

4.2 Reasoning DAGs as privileged information

Selective disclosure requires a representation that can be partially revealed without breaking the dependencies of a solution. An arbitrary text prefix can end midway through an inference, whereas a dependency graph makes its prerequisites explicit. We represent one verified solution route by a nonempty reasoning DAG $\mathcal{G}_i = (\mathcal{V}_i, \mathcal{E}_i)$. Each node is a mathematical checkpoint, corresponding to an intermediate state required by the solution; an edge $u \to v$ states that u is an immediate prerequisite of v. Fan-out represents parallel arguments, and fan-in records where those arguments must be combined.

We construct the graphs once, offline. A producer converts a verified solution into a dense dependency graph, which an independent judge reviews for fidelity and completeness. A second producer compresses the accepted graph into key checkpoints and is instructed to remove the terminal-answer layer. Appendix B details the construction procedure and gives examples of the resulting DAGs.

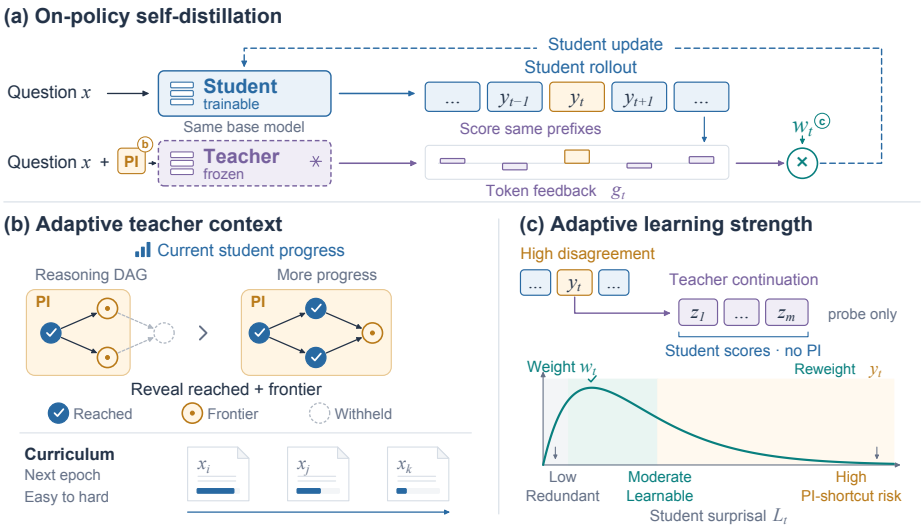


Figure 2: **AOPSD: adapt what to reveal and how strongly to learn.** (a) Teacher and student share an initialization, and the teacher scores the student's question-only trajectory using privileged information. (b) Based on the student's current capability, the DAG is adaptively disclosed, and all questions are presented through curriculum learning in order of difficulty. (c) High-gap tokens trigger teacher continuations scored by the question-only student. Band-pass weighting suppresses redundant and potentially PI-dependent guidance while retaining the learnable and useful guidance.

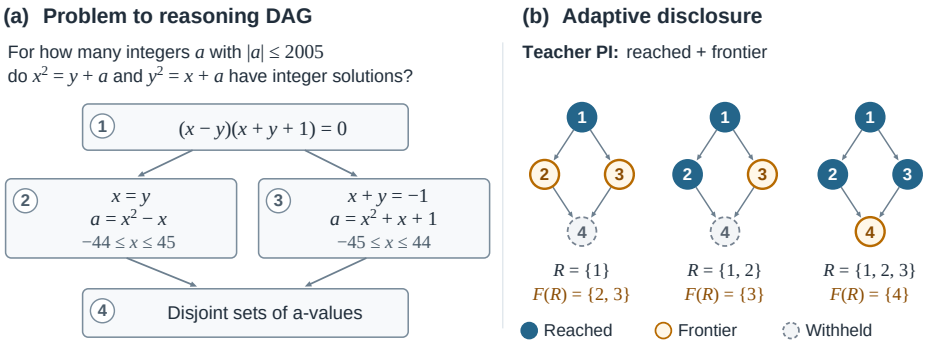


Figure 3: **A reasoning DAG reveals one complete dependency frontier at a time.** (a) A verified route is compressed into route-defining mathematical checkpoints; edges encode immediate prerequisites. (b) For a question-only rollout, reached states R are blue, immediately enabled states $F(R)$ are orange, and later states are gray. As more prerequisites are established, disclosure advances by one dependency layer rather than by an arbitrary text prefix.

The graph serves two roles: it supplies decomposable units of PI and a reference against which to assess a student's completeness and capability (Figure 3). Its checkpoints make both disclosure and progress assessment explicit at the level of one verified route.

4.3 Adaptive PI disclosure

Measure progress along the reference route. OPSD already generates a question-only response before teacher feedback. We reuse that rollout for an inference-only alignment against $\mathcal{G}_i$. Let $R_i(y)$ be the largest set of checkpoints judged to be established by y after removing any checkpoint whose prerequisites are absent. By construction, $R_i(y)$ is prerequisite closed. We define

$$r_i(y) = \frac{|R_i(y)|}{|\mathcal{V}_i|} \in [0, 1]. \tag{2}$$

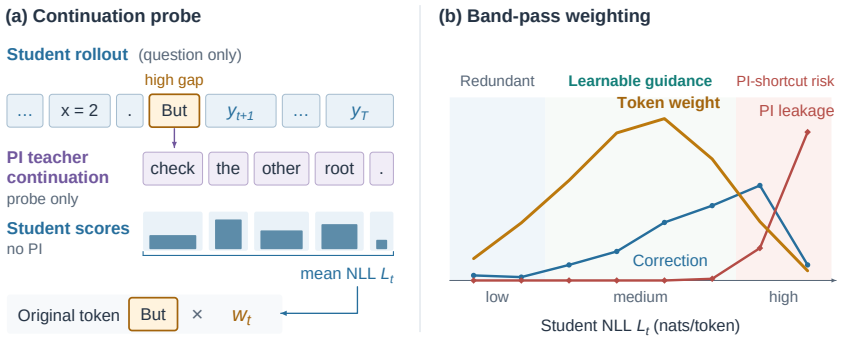


Figure 4: **Continuation surprisal controls the token's weight.** (a) For a large gap, the PI-conditioned teacher generates a continuation after the targeted token. The student then scores it via mean NLL L_t. (b) The band-pass rule favors intermediate surprisal relative to the two extremes.

$r_i(y)$ measures progress along the reference route, giving a finer estimate of the current problem's difficulty for the trained student model than final correctness, which is what both adaptations need. In practice, we prompt student model as the judger for correctness estimation.

Local adaptation: disclose the reached subgraph and its frontier. The teacher sees what the student has reached, plus the next checkpoints whose prerequisites are already reached:

$$F_i(R) = \{v \in \mathcal{V}_i \setminus R : \mathrm{pa}(v) \subseteq R\}, \qquad P_i(y) = \phi(\mathcal{G}_i[R_i(y) \cup F_i(R_i(y))]), \tag{3}$$

where $\mathrm{pa}(v)$ denotes the parents of v, $\mathcal{G}_i[S]$ is the subgraph induced by S, and ϕ serializes that subgraph as teacher context. A weak student unlocks little beyond the roots, and a stronger student unlocks a deeper subgraph (Figure 3b). The boundary is deliberate on both sides. Revealing more would place the teacher's preferences on states the student has shown no route to, which is the failure of Section 3; revealing less would withhold the one transition the student is ready to make. This keeps PI near the next missing step, rather than adding downstream details whose prerequisites the student has not yet established.

Global adaptation: update problem curriculum with problem competence. Averaged over a problem's attempts, the same completion is a per-problem competence. We initialize an exponential moving average from K question-only attempts and update it at epoch boundaries:

$$c_i^{(0)} = \frac{1}{K} \sum_{y \in \mathcal{Y}_i^{(0)}} r_i(y), \qquad c_i^{(e+1)} = (1-\lambda) c_i^{(e)} + \lambda \, \mathbb{E}_{y \in \mathcal{Y}_i^{(e)}}[r_i(y)]. \tag{4}$$

The next epoch visits all problems from higher to lower c_i, while bridge batches mix neighboring coverage strata to soften abrupt transitions. That is, within a single batch, difficult, moderate, and easy questions are mixed in a 1:2:1 ratio relative to the current model's capabilities. Thus, the local and global rules share one measured quantity: $r_i(y)$ controls how much of a route is disclosed for the current rollout, while c_i controls where the problem appears in the next epoch.

4.4 Continuation-based selective distillation

Disclosure controls the teacher's context, but individual token preferences can still reflect behavior that is difficult to transfer. We therefore supplement the sampled-token gap with a short continuation probe. Let p_θ denote the student policy that collected y. At collection time, define

$$g_t^{\mathrm{col}} = \log q_{\bar{\theta}}(y_t \mid x_i \oplus P_i(y), y_{<t}) - \log p_\theta(y_t \mid x_i, y_{<t}), \tag{5}$$

where $x_i \oplus P_i(y)$ denotes the teacher prompt containing the problem and disclosed PI. We probe positions in $\mathcal{H}(y) = \{t : |g_t^{\mathrm{col}}| \geq \delta\}$ for a threshold $\delta > 0$, where the teacher and student disagree significantly. For each $t \in \mathcal{H}(y)$, we choose an anchor a_t: retain $a_t = y_t$ for a positive gap, or let the PI-conditioned teacher choose a_t greedily from $y_{<t}$ for a negative gap. The teacher then generates

a greedy suffix $z^{(t)} = (z_1, \ldots, z_{m_t})$ of at most $k-1$ tokens after the anchor. The question-only student scores the nonempty suffix, excluding the anchor:

$$L_t = -\frac{1}{m_t} \sum_{j=1}^{m_t} \log p_\theta(z_j \mid x_i, y_{<t}, a_t, z_{<j}), \qquad 1 \le m_t \le k-1. \tag{6}$$

Thus L_t measures how surprising the teacher's short continuation is under the student policy. Our design hypothesis is that highly predictable continuations often add little guidance, whereas extremely surprising ones may rely on PI unavailable to the student. Intermediate surprisal offers a candidate balance between novelty and local support. We test this hypothesis with an NLL-blind semantic audit of 15,043 training-time continuation events (Appendix F.3). Useful guidance is enriched at intermediate NLL, whereas 91.35% of events with $L_t \ge 4$ are labeled as PI leakage. We encode this hypothesis with a band-pass coefficient:

$$w_t = \begin{cases} L_t/\beta_t \times \exp(1 - L_t/\beta_t), & t \in \mathcal{H}(y) \text{ and } m_t > 0, \\ 1, & \text{otherwise}, \end{cases} \qquad \beta_t > 0. \tag{7}$$

We use $\beta_t = \beta_+$ for positive gaps and $\beta_t = \beta_-$ for negative gaps. The weight vanishes as $L_t \to 0$ or $L_t \to \infty$ and peaks at $L_t = \beta_t$ with value 1. Thus, β_t sets the preferred surprisal while the maximum weight remains fixed. Untriggered tokens and empty probes retain unit weight.

4.5 Training objective

AOPSD trains the student with a sampled-token policy-gradient update. The collection-time gap in Equation (5) determines the probes. During each actor forward pass, we form a clipped, weighted advantage using the current student log-probability:

$$\widehat{A}_t = \text{clip}(\text{sg}[w_t \left(\log q_{\bar{\theta}}(y_t \mid x_i \oplus P_i(y), y_{<t}) - \log p_\theta(y_t \mid x_i, y_{<t})\right)], -C, C), \tag{8}$$

where sg stops gradients and $C > 0$ bounds the advantage magnitude. The policy-gradient update uses $\widehat{A}_t$ on the original student token y_t: a positive advantage favors that token, and a negative advantage discourages it. The weighted advantage enters a clipped policy-gradient objective on the sampled responses; the continuation tokens supply weights rather than additional training targets. Appendix C gives the pseudocode, key configuration, and implementation details.

5 Experiments

5.1 Experimental Setup

Data and evaluation. We use the same 725 problems from LIMO-V2 (Ye et al., 2025) as training data for all methods and evaluate on four benchmarks: HMMT25 (FlagEval Team, 2025) (February), AIME24 (Hugging Face H4, 2025), AIME25 (Test-Time Compute Organization, 2025), and BRUMo25 (MathArena, 2025). We sample eight responses per problem at temperature 0.6, with at most 12,288 output tokens, and report Pass@8: the percentage of problems with at least one correct response. We compare checkpoints after each of the first three training epochs. "Avg." is the unweighted mean over the four benchmarks for baseline comparisons. All gains are in percentage points (pp). Appendix F provides complete baseline, component, and sensitivity results and details the evaluation scope.

Baselines and configuration. We utilize Qwen3-4B-Instruct-2507 (Qwen Team, 2025) as the base model for the main evaluations. We additionally evaluate Qwen3.5-2B on the same four benchmarks in Appendix H. We compare with the base model before task-specific training, supervised fine-tuning (SFT), GRPO (Shao et al., 2024), and three distillation baselines: OPSD (Zhao et al., 2026), OP^{2}SD (Ichihara et al., 2026), and DASH (Hou et al., 2026). OPSD, OP^{2}SD, and DASH condition the teacher on an example solution. AOPSD uses adaptive DAG disclosure with continuation threshold $\delta = 2$, total probe length $k = 8$, and sign-specific scales $(\beta_+, \beta_-) = (1, 2.5)$. Student rollouts use temperature 1.0 and teacher probes are greedy. Appendix C gives the algorithm and key settings; Appendix E gives the prompts.

Table 2: **Overall performance.** Pass@8 (%, ↑). The left block compares the final reported checkpoint; the right block shows the four-benchmark macro-average trajectory.

	Across benchmarks				Avg. by epoch		
Method	HMMT25	AIME24	AIME25	BRUMo25	E1	E2	E3
Base	46.7	76.7	66.7	73.3	65.9	65.9	65.9
SFT	46.7	80.0	63.3	76.7	65.0	66.7	66.7
GRPO	46.7	83.3	66.7	76.7	70.8	70.8	68.4
OPSD	43.3	76.7	66.7	76.7	63.4	68.4	65.9
OP^{2}SD	46.6	80.0	70.0	76.7	69.2	68.3	68.3
DASH	46.7	76.7	66.7	76.7	70.1	65.9	66.7
AOPSD	**50.0**	**86.7**	**73.3**	**80.0**	**74.2**	**72.5**	**72.5**

5.2 Main Results: Stronger Transfer Throughout Training

The gains extend across benchmarks. AOPSD reaches 72.5% average Pass@8, improving over OPSD by 6.7 pp and over the strongest competing average by 4.2 pp (Table 2). It also leads every individual benchmark, with gains over the base model of 3.3 pp on HMMT25, 10.0 pp on AIME24, 6.6 pp on AIME25, and 6.7 pp on BRUMo25. AOPSD achieves 74.2% in Epoch 1 and retains 72.5% in both later checkpoints (Table 2, right), while GRPO finishes at 68.4% and OPSD at 65.9%. AOPSD leads or ties the baseline methods in all twelve benchmark–checkpoint comparisons, sustaining its advantage throughout training.

Computational cost. AOPSD preserves OPSD's dominant computation: student rollout, teacher scoring, and one actor update. Relative to the KL-based OPSD, OP^{2}SD, and DASH, the main concerns on cost are the sampled-token policy-gradient objective (PG) and selective continuation probes. The objective reduces the target-specific loss from K_d terms to one per token and supervision storage from $O(NK_d)$ to $O(N)$; the resulting payload is 89.8% smaller than Top-16 and 94.9% smaller than Top-32 KL. Besides, probes require inference only: with a hypothetical 1% trigger rate and cached-prefix reuse, their analytical overhead is just 2.14% of the reference distillation cost. In empirical tests, PG keeps selective feedback lightweight: PG with continuation takes 27 minutes per epoch, versus 30 for Top-16 KL and 32 for Top-32 KL (Table 3). This reduces epoch time by 10.0% and 15.6%, respectively. Appendix D details the timing scope and the computation costs.

Table 3: **Empirical test on training time**

Configuration	Time ↓ (min/epoch)	Relative time
Sampled-token PG	24	1.000×
PG + continuation	**27**	1.125×
Top-16 forward KL	30	1.250×
Top-32 forward KL	32	1.333×

5.3 Ablation Analysis: Context, Curriculum, and Feedback

Figure 5 compares six configurations through their average performance across HMMT25, AIME24, and AIME25: component additions to OPSD and DAG-disclosure variants under the OPSD objective. Table 6 reports every benchmark–checkpoint score.

Context and feedback adaptation both contribute. At Epoch 3, continuation alone raises the average from 62.2% to 67.8%, while adaptive PI with curriculum reaches 68.9%; the full method reaches 70.0%. Combining the adaptations improves over the stronger individual component by 3.3 pp in Epoch 1 and 1.1 pp in both later epochs. The gains support adapting both the context that induces teacher preferences and the weight those preferences receive in the update.

Selective disclosure matters beyond DAG structure. With the OPSD objective fixed, replacing the full target DAG with the student-dependent frontier raises the final average from 62.2% to 68.9%. Merely structuring the complete solution therefore does not yield the same transfer as limiting what the teacher sees. Full AOPSD further improves over frontier disclosure by 4.4 pp in Epoch 1 and 1.1 pp at both later checkpoints.

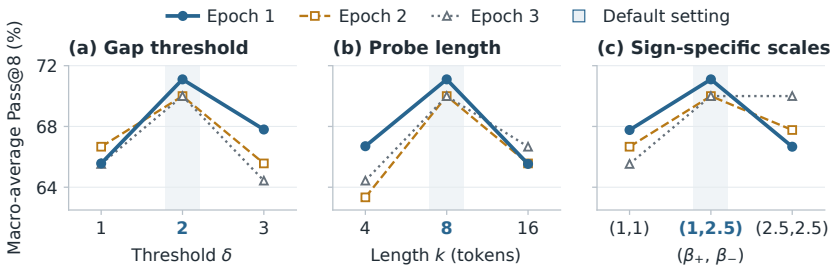


Figure 6: **Hyperparameter analysis.** Each panel varies one factor around $(\delta, k, \beta_+, \beta_-) = (2, 8, 1, 2.5)$, and Full benchmark scores appear in Table 7.

5.4 Hyperparameter Analysis

Probe the high-gap tail selectively. The intermediate threshold $\delta = 2$ outperforms $\delta = 1$ and $\delta = 3$ at every checkpoint, reaching 70.0% versus 65.5% and 64.4% in Epoch 3 (Figure 6a). Broader and narrower triggering both weaken transfer in this sweep: disagreement magnitude identifies where to inspect feedback, but the most extreme cutoff is not best.

A Longer continuation would not help. The eight-token budget gives the strongest average at all three checkpoints, exceeding the four- and sixteen-token alternatives by 5.6 and 3.3 pp at Epoch 3 (Figure 6b). This supports using a short local probe: additional teacher generation is useful only insofar as it improves the weight on the original student token.

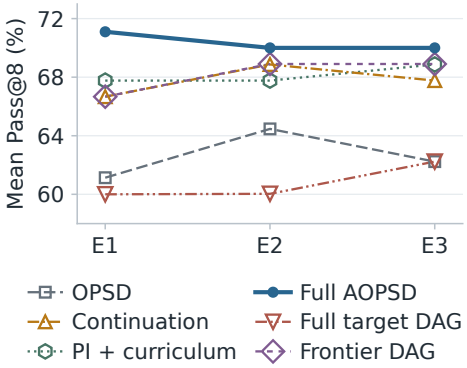


Figure 5: **Ablation study.** Continuation and PI + curriculum add one adaptation to OPSD; Full AOPSD combines both. The DAG variants change context under the OPSD objective. Table 6 gives all scores.

Sign-specific scales improve early learning. The asymmetric choice $(1, 2.5)$ gives the highest average in Epochs 1 and 2; the symmetric $(2.5, 2.5)$ choice catches up at Epoch 3, while $(1, 1)$ remains at 65.5% (Figure 6c). The early advantage is consistent with treating reinforcement of the sampled token and redirection toward a teacher alternative differently. The late tie also shows that asymmetry is not necessary to reach the final observed average.

6 Conclusion

On-policy supervision does not by itself make privileged guidance transferable. Our study shows that a teacher can become a stronger conditional solver yet provide feedback that fails to improve question-only reasoning. AOPSD addresses this gap by regulating both the information that induces teacher preferences and the influence those preferences have on learning. Reasoning-DAG progress guides disclosure and curriculum, while student-scored continuation probes calibrate high-disagreement feedback according to its surprisal without privileged information. Across four benchmarks, AOPSD achieves a final macro-average Pass@8 of 72.5%, exceeding OPSD by 6.7 pp. Ablations further show that student-dependent disclosure transfers better than exposing the full DAG, and that jointly adapting context and feedback improves the final average over either component alone. Together, these findings suggest a learner-centered principle for privileged self-distillation: use the teacher's informational advantage to extend the student's reasoning frontier, rather than substitute for the reasoning the student must learn to perform unaided.

AI USE STATEMENT

Generative AI assistance in preparing this manuscript was limited to polishing language, clarity, and presentation. The research ideas, method design, experiments, and scientific conclusions are the authors' own. Language-model calls within the research workflow, including DAG construction and progress assessment, are described in Section 4 and Appendices B and E. The authors take full responsibility for the final content of this paper.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, 2024.

Nicolas Boizard, Kevin El Haddad, Céline Hudelot, and Pierre Colombo. Towards cross-tokenizer distillation: the universal logit distillation loss for LLMs. *Transactions on Machine Learning Research*, 2025.

FlagEval Team. HMMT february 2025 dataset. Hugging Face dataset, 2025. URL https://huggingface.co/datasets/FlagEval/HMMT_2025.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *International Conference on Learning Representations*, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638, 2025. doi: 10.1038/s41586-025-09422-z.

ZhiYan Hou, Xinyu Tang, Hongyan An, Jianjin Zhang, Weizhen Wang, Yunyun Han, Gengsheng Li, Xiangzhao Hao, Haiyun Guo, Wenbin Hu, Jinqiao Wang, and Yafeng Deng. DASH: Divergence-adaptive supervision horizons for on-policy self-distillation of reasoning models. *arXiv preprint arXiv:2608.06243*, 2026. URL https://arxiv.org/abs/2608.06243.

Hugging Face H4. AIME 2024 dataset. Hugging Face dataset, 2025. URL https://huggingface.co/datasets/HuggingFaceH4/aime_2024.

Yuki Ichihara, Naoto Iwase, Mohammad Atif Quamar, and Junpei Komiyama. Privileged solutions or context-induced teacher behavior? dissecting on-policy self-distillation. *arXiv preprint arXiv:2608.09228*, 2026. URL https://arxiv.org/abs/2608.09228.

Woogyeol Jin, Taywon Min, Yongjin Yang, Dennis Wei, Yi Zhou, Swanand Ravindra Kadhe, Nathalie Baracaldo, and Kimin Lee. Entropy-aware on-policy distillation of language models. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2603.07079.

MathArena. BRUMO 2025 dataset. Hugging Face dataset, 2025. URL https://huggingface.co/datasets/MathArena/brumo_2025.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

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. *arXiv preprint arXiv:2402.03300*, 2024.

Test-Time Compute Organization. AIME 2025: Unified test-time scaling format. Hugging Face dataset, 2025. URL https://huggingface.co/datasets/test-time-compute/aime_2025.

Shubham Toshniwal, Wei Du, Ivan Moshkov, Branislav Kisacanin, Alexan Ayrapetyan, and Igor Gitman. OpenMathInstruct-2: Accelerating AI for math with massive open-source instruction data. In *International Conference on Learning Representations*, pp. 19243–19275, 2025.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. LIMO: Less is more for reasoning. In *Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=T2TZ0RY4Zk.

Longhui Yu, Weisen Jiang, Han Shi, Jincheng Yu, Zhengying Liu, Yu Zhang, James T. Kwok, Zhenguo Li, Adrian Weller, and Weiyang Liu. MetaMath: Bootstrap your own mathematical questions for large language models. In *International Conference on Learning Representations*, pp. 45040–45061, 2024.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, LingJun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. In *Advances in Neural Information Processing Systems*, volume 38, pp. 113222–113244, 2025. doi: 10.52202/085713-3775.

Xiang Yue, Xingwei Qu, Ge Zhang, Yao Fu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. MAmmoTH: Building math generalist models through hybrid instruction tuning. In *International Conference on Learning Representations*, 2024.

Songming Zhang, Xue Zhang, Zengkui Sun, Yufeng Chen, and Jinan Xu. Dual-space knowledge distillation for large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 18164–18181, 2024. doi: 10.18653/v1/2024.emnlp-main.1010.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. In *International Conference on Machine Learning*, 2026.

A Diagnostic protocols and privileged-information formats

A.1 Conditional utility and question-only transfer

The preliminary diagnostic in Figure 1a separates *solving with PI* from *learning to solve without PI*. It uses Qwen3-4B-Instruct-2507 and 725 training problems with verified solutions. For each PI format, the reported OPSD run uses four epochs, learning rate 3×10^{-6}, and a response limit of 8,192 tokens. Conditional utility is evaluated on those 725 problems with PI present; transfer is evaluated on AIME 2025 with only the question. Both evaluations use temperature 0.6, a 12,288-token output limit, and eight attempts per problem.

A.2 Token and continuation audits

The token audit in Figure 1b uses training-set rollouts with temperature 0.6 and an 8,192-token response limit. Teacher and student token probabilities are scored at temperature 0.6. Tokens are grouped by the absolute sampled-token gap $|g_t|$ into $[0, 1)$, $[1, 2)$, and $[2, \infty)$. Each bin reports its share of response tokens and of $\sum_t |g_t|$ which quantifies concentration of the sampled-token disagreement.

The continuation audit in Table 1 covers 9,721 non-overlapping events with DAG PI and $|g_t| \geq 2$. At each event, the student token is held fixed and the teacher greedily generates up to 15 further tokens; the next event begins after this window. Counts accumulate over prefixes of $2, 4, 6, 8$, and 16 tokens, including the fixed token. Thus, columns at different lengths describe the same events with increasing context, rather than independent samples. The recorded categories are recognizable correction cues and explicit PI references.

A.3 Privileged-information examples

The following formats are condensed from the same training record. The full solution supplies a worked derivation; the plan organizes its operations; the decomposition pairs subquestions with answers; and the DAG exposes dependencies between mathematical checkpoints.

Problem. Find the greatest positive integer m such that $\text{lcm}(40, m) = 120$ and $\text{lcm}(m, 45) = 180$.

Full solution. Factor $40 = 2^3 \cdot 5$, $45 = 3^2 \cdot 5$, $120 = 2^3 \cdot 3 \cdot 5$, and $180 = 2^2 \cdot 3^2 \cdot 5$. Write $m = 2^a 3^b 5^c$ with nonnegative integer exponents. The first LCM gives $a \leq 3$, $b = 1$, $c \leq 1$; the second gives $a = 2$, $b \leq 2$, $c \leq 1$. Together, $m = 12 \cdot 5^c$ with $c \in \{0, 1\}$, so the maximum is $\boxed{60}$.

Solution plan. **Variables:** positive integer m; two LCM constraints; maximize m. **Plan:** compare prime exponents under each constraint, then intersect them. **Calculation:** write $m = 2^a 3^b 5^c$; obtain $(a \leq 3, b = 1, c \leq 1)$ and $(a = 2, b \leq 2, c \leq 1)$; hence $m = 12 \cdot 5^c \in \{12, 60\}$. **Answer:** $\boxed{60}$.

Subproblem decomposition. (1) Which primes can divide m? Only $2, 3, 5$, so write $m = 2^a 3^b 5^c$. (2) What does the first LCM imply? $a \leq 3$, $b = 1$, $c \leq 1$. (3) What does the second imply? $a = 2$, $b \leq 2$, $c \leq 1$. (4) What is the greatest m? Combining gives $m \in \{12, 60\}$, so $\boxed{60}$.

Reasoning DAG. $S_1 \to \{S_2, S_3\} \to S_4$.

Node	Parents	Checkpoint	Incoming inference
S_1	$\varnothing$	$m = 2^a 3^b 5^c$; no other prime factors.	—
S_2	S_1	$a \leq 3$, $b = 1$, $c \leq 1$.	Compare exponents in the first LCM.
S_3	S_1	$a = 2$, $b \leq 2$, $c \leq 1$.	Compare exponents in the second LCM.
S_4	S_2, S_3	$a = 2$, $b = 1$, $c \leq 1$.	Intersect both sets of constraints.

Full solutions give derivations, plans organize operations, decompositions pair subquestions with answers, and DAGs expose prerequisite relations for selective disclosure.

B Reasoning DAG construction and disclosure

B.1 Construction

Each graph encodes one verified solution route, preserving the intermediate states needed to understand its dependencies. Construction has two stages:

1. **Dense graph.** A producer converts one verified solution route into mathematical states linked by immediate prerequisites. A judge checks the graph against the solution for correctness, completeness, and dependency consistency, then returns revision feedback.
2. **Sparse graph.** A second producer retains route-defining states and their dependencies, preserving parallel branches and necessary merges. It is instructed to remove routine calculations and the terminal-answer layer. A second judge reviews the compressed graph.

Both construction stages use `glm5.2-fp8` for the producer and judge roles, with role-specific prompts given in Appendix G. Each model call has a maximum output budget of $24{,}000$ tokens. The dense and sparse stages each allow at most five producer–judge rounds (`MAX_ITERATIONS=5` per stage). The four node fields are `step` (identifier), `parents` (immediate prerequisites), `node` (mathematical state), and `edge` (incoming inference). Parents precede their children and roots have empty parent lists. The sparse producer is instructed to remove routine calculations while retaining the dependencies needed to interpret each checkpoint. Across the 725 records, dense graphs contain a mean of 9.01 checkpoints (median 8; range 2–28), whereas sparse graphs contain 4.13 (median 4; range 1–12). This is a 54.1% reduction in mean checkpoint count, providing smaller units for selective disclosure.

From a rollout to disclosed PI. The student judges which checkpoints its own response has established. We then remove any reported checkpoint whose parents are absent, repeating this operation until the reached set is prerequisite closed. The disclosed JSON contains this reached set and one frontier layer. Consequently, an empty reached set reveals the roots; a fully reached route reveals the full sparse graph; and a merge becomes eligible only when all its parents have been reached. The inference prompt is given in Appendix E.5.

B.2 Examples from the training data

A chain advances one restriction at a time. Find the least positive integer n satisfying $n \equiv 2 \pmod 5$, $n \equiv 3 \pmod 7$, and $n \equiv 4 \pmod 9$.

Node	Parents	Checkpoint	Incoming inference
S_1	$\varnothing$	$n = 4 + 9k$, $k \in \mathbb{Z}$.	—
S_2	S_1	$n = 31 + 63m$, $m \in \mathbb{Z}$.	Impose $n \equiv 3 \pmod 7$ and substitute $k = 3 + 7m$.
S_3	S_2	$m \equiv 2 \pmod 5$.	Impose $n \equiv 2 \pmod 5$.

A merge requires both branches. How many distinct rational numbers k satisfy $|k| < 200$ and make $5x^2 + kx + 12 = 0$ have an integer solution?

Node	Parents	Checkpoint	Incoming inference
S_1	$\varnothing$	$k = -5x - 12/x$, with $x \in \mathbb{Z} \setminus \{0\}$.	—
S_2	S_1	For $x > 0$: $k < 0$ and $5x + 12/x < 200$.	Apply $\|k\| < 200$ to the positive branch.
S_3	S_1	For $x = -m < 0$: $k = 5m + 12/m > 0$ and $5m + 12/m < 200$.	Apply $\|k\| < 200$ to the negative branch.
S_4	S_2, S_3	The two branches give disjoint sets of k.	Compare their signs.

For the chain, reaching S_1 reveals S_2 but not S_3. For the branching graph, reaching S_1 reveals both S_2 and S_3; S_4 enters the frontier only after both branches have been established. These cases illustrate why dependency closure, rather than textual order, determines which information is available next.

C Training algorithm and mechanism settings

Algorithm 1 summarizes the formulation in Section 4. Each response serves three purposes: it is the on-policy training sample, the evidence for DAG progress, and the prefix on which the teacher supplies feedback. The short continuation is used only to set the original token's weight.

Algorithm 1 Adaptive on-policy self-distillation

Input: Questions and DAGs $\{(x_i, \mathcal{G}_i)\}$; student θ_0; fixed teacher $\bar{\theta} = \theta_0$; initialization budget K; epochs E.

Initialize c_i from K question-only attempts using Equation (4).
for each epoch **do**
  Order problems by decreasing c_i.
  for each batch $\mathcal{B}$ **do**
    Freeze $\theta_{\text{old}} \leftarrow \theta$; sample $y_i \sim p_{\theta_{\text{old}}}(\cdot \mid x_i)$ for $i \in \mathcal{B}$.
    Align y_i to its DAG to obtain prerequisite-closed $R_i(y_i)$ and progress $r_i(y_i)$.
    Disclose $P_i = \phi(\mathcal{G}_i[R_i \cup F_i(R_i)])$; score the original response with the teacher.
    Compute g_t^{col} (Equation (5)); initialize $w_t \leftarrow 1$.
    for each token with $|g_t^{\text{col}}| \geq \delta$ **do**
      Set $a_t = y_t$ for a positive gap, or choose a_t greedily with the teacher for a negative gap. Generate up to $k - 1$ suffix tokens after a_t.
      For a nonempty suffix, compute its question-only NLL L_t, excluding a_t, and set w_t by Equation (7).
    end for
    Form $\widehat{A}_t$ using current p_θ (Equation (8)); update θ on the original responses with clipped policy gradient.
  end for
  Update each c_i from this epoch's observed progress using Equation (4).
end for; **return** the question-only student p_θ.

Table 4: **Mechanism settings reported for AOPSD.**

Parameter	Setting
High-gap threshold δ	2
Continuation budget k	8: one anchor and at most seven suffix tokens
Preferred surprisal (β_+, β_-)	$(1,\ 2.5)$ for positive and negative gaps
Advantage bound C	10
Competence EMA λ	0.2
Batch size	30
Student rollout / token scoring temperature	1.0
Teacher continuation decoding	Greedy
DAG lookahead	One prerequisite-complete frontier layer

Reference optimization preset. The archived launcher provides a learning rate of 5×10^{-6} with cosine decay, weight decay 0.01, gradient clipping at 1.0, a policy-ratio clip of 0.2, and token-mean loss aggregation. It samples one response per question with a training response limit of 8,192 tokens.

The sign determines the probe, not an extra target. A positive gap means that the teacher assigns greater probability than the student to the sampled token. Its probe retains that token and examines what the teacher would write next. A negative gap means that the teacher assigns lower probability to the sampled token; its probe starts from the teacher's greedy choice at the same prefix. This choice need not differ from the sampled token. In both cases, the suffix is scored without PI, the anchor is excluded from its mean NLL, and the update remains attached to the original sampled token. An empty suffix retains unit weight.

Frozen feedback and current-policy optimization. The collection policy determines triggers and continuation weights once per rollout. During the actor update, the teacher score and weight remain fixed, while the current student's log-probability determines the stopped-gradient advantage. The

weight is applied before clipping the advantage to $[-C, C]$. Neither the progress judgment nor the continuation requires an additional training loss.

D Gradient objectives, memory, and continuation cost

AOPSD retains the same dominant distillation schedule as OPSD. Its cost differences arise from the gradient objective, selective continuation, and progress assessment. We first report measured epoch time, then explain the objective's cost structure and the conditions that keep probing inexpensive.

Measured epoch time. Table 3 compares four configurations with the same model, data, and otherwise identical training settings on four NVIDIA H100 GPUs: 24 minutes for sampled-token PG, 27 for PG with continuation, 30 for Top-16 forward KL, and 32 for Top-32 forward KL. PG alone uses 20.0% less time than Top-16 and 25.0% less than Top-32. Adding continuation costs 3 minutes per epoch (12.5%), while retaining a net time reduction of 10.0% and 15.6%, respectively.

One sampled-token target instead of a distribution target. Let $z_{t,v}$ be a student logit and $p_t = \mathrm{softmax}(z_t)$. Holding the advantage fixed as in Section 4.5, the basic PG surrogate and its logit gradient are

$$\ell_t^{\mathrm{PG}} = -\,\mathrm{sg}(\widehat{A}_t)\log p_t(y_t), \qquad \frac{\partial \ell_t^{\mathrm{PG}}}{\partial z_{t,v}} = \widehat{A}_t\big(p_t(v) - \mathbf{1}[v = y_t]\big).$$

For support-renormalized Top-K_d forward KL, let S_t be the retained support and $\widetilde{p}_t, \widetilde{q}_t$ the student and teacher distributions normalized on S_t. Then

$$\ell_t^{\mathrm{KL}} = \sum_{v \in S_t} \widetilde{q}_t(v) \log \frac{\widetilde{q}_t(v)}{\widetilde{p}_t(v)}, \qquad \frac{\partial \ell_t^{\mathrm{KL}}}{\partial z_{t,v}} = \mathbf{1}[v \in S_t]\big(\widetilde{p}_t(v) - \widetilde{q}_t(v)\big).$$

PG reduces target-specific loss terms from K_d to one per token; policy-ratio weighting and clipping introduce no additional vocabulary dependencies.

Analytical supervision storage. For N response tokens, Top-K_d KL stores K_d int32 token IDs and FP32 teacher log-probabilities per token. PG with continuation stores one ID, one log-probability, one FP32 weight, and one Boolean trigger mask:

$$M_{\text{Top-}K_d} = 8NK_d \text{ bytes}, \qquad M_{\text{ours}} = 13N \text{ bytes}. \tag{9}$$

This representation reduces the supervision payload by 89.8% relative to Top-16 and 94.9% relative to Top-32. For $N = 50{,}000$, the payloads are 6.10, 12.21, and 0.62 MiB for Top-16, Top-32, and PG with continuation, respectively.

Analytical continuation cost. To explain probe scaling, we retain a common FLOP reference using the 4B architecture, prompt length $P = 2048$, and response length $T = 8192$ for both models. Counting matrix multiplications, including causal attention, at two FLOPs per multiply-add gives 111.7 TFLOPs for student rollout, 113.3 for teacher scoring, and 445.2 for one actor update: $F_0 \approx 670.2$ TFLOPs per sample. The actor update includes one Transformer recomputation and backward computation at twice the forward cost, with no output-layer recomputation.

Let ρ denote the fraction of response tokens that trigger probes and α the positive-gap fraction among them. A positive-gap probe retains its anchor and generates up to $k-1$ tokens; a negative-gap probe also selects its anchor. Charging teacher generation and student scoring gives a nominal budget of $2(k-\alpha)$ model-token executions per trigger. With perfect prefix reuse,

$$\Delta F \approx \rho T\, 2(k-\alpha) f(\bar{\ell}), \qquad \bar{\ell} = P + (T-1)/2, \tag{10}$$

where $f(\bar{\ell})$ is the incremental execution cost at the mean prefix length. Uniform trigger locations give $\bar{\ell} = 6143.5$ and $f(\bar{\ell}) \approx 11.67$ GFLOPs. At the illustrative values $\rho = 0.01$, $k = 8$, and $\alpha = 0.5$, probes add 14.3 TFLOPs, or 2.14% of F_0.

E Training and evaluation prompts

We report the task-level instructions for student rollouts, privileged teacher feedback, progress judgments, and evaluation. Angle-bracketed placeholders denote sample-specific content. Line breaks and escape characters are normalized for display; the model's chat template supplies conversation formatting.

E.1 Question-only training prompt

The student receives only the target problem, denoted by <PROBLEM>, with no privileged information.

```
Solve the problem step by step, and put your final answer within
\boxed{}.

Problem:
<PROBLEM>
```

E.2 OPSD teacher prompt

<SOLUTION> is the reference solution to the target problem.

```
You are a careful mathematical problem solver.

You will be given a problem together with a verified private
solution. The solution provides trusted mathematical guidance for a
correct solution. It contains the reasoning, derivations,
calculations, and final answer of a correct solution.

Problem:
<PROBLEM>

Private solution:
<SOLUTION>

Use the solution as trusted guidance for understanding the reasoning
direction and key mathematical relationships of a correct solution.
Reason through the problem naturally from its conditions.

Trust the mathematical information in the solution, but do not
mechanically reproduce its wording or structure. Do not mention the
solution in your response.

Solve the problem step by step in a clear and natural way, and put
the final answer within \boxed{}.
```

E.3 OP^{2}SD teacher prompt

<EXAMPLE_PROBLEM> and <EXAMPLE_SOLUTION> form a worked example from another problem, as stored in the training file.

```
You are a careful mathematical problem solver.
Target problem:
<PROBLEM>

Below is a worked example from a different and unrelated mathematics
problem. It is provided only to demonstrate careful mathematical
reasoning and solution structure. It is not a solution or hint for
the target problem. Do not reuse its final answer, variables,
numerical values, or problem-specific conclusions.

=== Unrelated Worked Example Problem Begin ===
<EXAMPLE_PROBLEM>
=== Unrelated Worked Example Problem End ===

=== Unrelated Worked Example Solution Begin ===
<EXAMPLE_SOLUTION>
=== Unrelated Worked Example Solution End ===
```

```
Solve the problem step by step in a clear and natural way, and put
the final answer within \boxed{}.
```

E.4 AOPSD TEACHER PROMPT

<DAG_JSON> denotes the JSON-encoded reasoning DAG. Adaptive selection replaces this field with the reached checkpoints and their frontier, keeping the surrounding instructions unchanged.

```
You are a careful mathematical problem solver.

You will be given a problem together with a verified private
reasoning DAG. The DAG provides trusted mathematical checkpoints for
a correct solution. It captures the key states and reasoning
transitions that characterize a correct solution, but it is not a
step-by-step solution script.

Each node in the DAG has four fields:

* `step`: the node id.
* `parents`: the prerequisite nodes.
* `edge`: the reasoning transition.
* `node`: the key mathematical state.

Problem:
<PROBLEM>

Private reasoning DAG:
<DAG_JSON>

Use the DAG as trusted guidance for understanding the key
mathematical states and reasoning direction of a correct solution.
Reason through the problem naturally from its conditions.

Trust the mathematical information in the DAG, but do not
mechanically reproduce its wording or structure. Do not mention the
DAG, its nodes, or its fields in the solution.

Solve the problem step by step in a clear and natural way, and put
the final answer within \boxed{}.
```

E.5 PROGRESS-ALIGNMENT PROMPT

The student judges its rollout in an inference-only continuation of the conversation. After the question and the student's response, the following user message supplies the full DAG. The result is filtered to a prerequisite-closed set before computing coverage.

```
Compare your response with the full DAG below. List only the nodes
whose mathematical states you correctly and sufficiently
established. Include a node only if all its parent nodes are also
included. Return exactly one JSON object with no explanation:
{"achieved_steps":["S1","S2"]}. Use an empty list if no node was
achieved.
Full DAG:
<FULL_DAG_JSON>
```

The judge uses greedy decoding. The full graph in this judgment is used only to assess progress; it is separate from the selected subgraph in the teacher's distillation context.

E.6 QUESTION-ONLY EVALUATION PROMPT

All benchmark attempts receive the following user message, without PI:

```
<PROBLEM>
Reason step by step. End your response with exactly one final answer
in the form \boxed{answer}. Do not write anything after the box.
```

F Complete Results and Evaluation Details

F.1 Metric and data

For benchmark b with $n_b = 30$ questions and eight responses per question,

$$\text{Pass@8}_b = \frac{100}{n_b} \sum_{i=1}^{n_b} \mathbf{1}\left\{ \sum_{j=1}^{8} \mathbf{1}[\hat{a}_{ij} = a_i] > 0 \right\}, \qquad \text{Avg.} = \frac{1}{4} \sum_{b} \text{Pass@8}_b \,. \tag{11}$$

Data sources. The training subset is selected from `GAIR/LIMO-v2` (Ye et al., 2025). The evaluation releases are `FlagEval/HMMT_2025` (FlagEval Team, 2025) (the February tournament), `HuggingFaceH4/aime_2024` (Hugging Face H4, 2025), BRUMo25 (MathArena, 2025), and `test-time-compute/aime_2025` (Test-Time Compute Organization, 2025).

F.2 Complete checkpoint comparisons

Tables 5–7 give the complete 4B baseline, component and disclosure ablation, and continuation-sensitivity scores; triplets denote Epoch 1 / Epoch 2 / Epoch 3. Table 6 includes all six configurations plotted in Figure 5, with every benchmark score at all three epochs. Averages and differences are computed from the displayed benchmark values before rounding.

Table 5: **Complete baseline comparison.** Qwen3-4B-Instruct-2507; Pass@8 (%, ↑), ordered E1 / E2 / E3. Bold marks the best score for each metric and epoch, including ties.

Method	HMMT25	AIME24	AIME25	BRUMo25	Avg.
Base	46.7	76.7	66.7	73.3	65.9
SFT	46.7 / **50.0** / 46.7	76.7 / 80.0 / 80.0	60.0 / 63.3 / 63.3	76.7 / 73.3 / 76.7	65.0 / 66.7 / 66.7
GRPO	**53.3** / **50.0** / 46.7	**83.3** / 83.3 / 83.3	**76.7** / **73.3** / 66.7	70.0 / 76.7 / 76.7	70.8 / 70.8 / 68.4
OPSD	46.7 / 46.7 / 43.3	76.7 / 80.0 / 76.7	60.0 / 66.7 / 66.7	70.0 / **80.0** / 76.7	63.4 / 68.4 / 65.9
OP^{2}SD	46.6 / 46.6 / 46.6	80.0 / 80.0 / 80.0	73.3 / **73.3** / 70.0	76.7 / 73.3 / 76.7	69.2 / 68.3 / 68.3
DASH	**53.3** / **50.0** / 46.7	80.0 / 76.7 / 76.7	73.7 / 66.7 / 66.7	73.3 / 70.0 / 76.7	70.1 / 65.9 / 66.7
AOPSD	**53.3 / 50.0 / 50.0**	**83.3 / 86.7 / 86.7**	**76.7 / 73.3 / 73.3**	**83.3 / 80.0 / 80.0**	**74.2 / 72.5 / 72.5**

Benchmark-specific ablation effects. Full AOPSD's advantage over either individual component is largest on AIME25 at Epoch 1 (76.7% versus 70.0%). PI/curriculum alone remains stronger on HMMT25 later, while adding continuation improves final AIME24 by 6.7 pp. Thus, aggregate gains combine task-specific and early-training benefits; Table 6 reports the complete breakdown.

Table 6: **Component and disclosure ablations.** Qwen3-4B-Instruct-2507; Pass@8 (%, ↑), ordered E1 / E2 / E3. Bold marks the best score for each metric and epoch, including ties.

Configuration	HMMT25	AIME24	AIME25	Avg.
Vanilla OPSD	46.7 / 46.7 / 43.3	76.7 / 80.0 / 76.7	60.0 / 66.7 / 66.7	61.1 / 64.5 / 62.2
Full target DAG	40.0 / 46.7 / 46.7	**83.3** / 76.7 / 80.0	56.7 / 56.7 / 60.0	60.0 / 60.0 / 62.2
Frontier DAG	50.0 / 46.7 / 46.7	**83.3** / **86.7** / **86.7**	66.7 / **73.3** / **73.3**	66.7 / 68.9 / 68.9
+ Continuation only	46.7 / 50.0 / 46.7	**83.3** / 83.3 / 83.3	70.0 / **73.3** / **73.3**	66.7 / 68.9 / 67.8
+ PI + curriculum	**53.3** / **53.3** / **53.3**	80.0 / 83.3 / 80.0	70.0 / 66.7 / **73.3**	67.8 / 67.8 / 68.9
Full AOPSD	**53.3** / 50.0 / 50.0	**83.3** / **86.7** / **86.7**	**76.7** / **73.3** / **73.3**	**71.1** / **70.0** / **70.0**

Table 7: **Continuation sensitivity.** Qwen3-4B-Instruct-2507; Pass@8 (%, ↑), ordered E1 / E2 / E3. Bold marks the best score for each metric and epoch, including ties.

$(\delta, k, \beta_+, \beta_-)$	HMMT25	AIME24	AIME25	Avg.
$(1, 8, 1, 2.5)$	46.7 / 46.7 / 43.3	76.7 / 83.3 / 83.3	73.3 / 70.0 / 70.0	65.6 / 66.7 / 65.5
$(2, 4, 1, 2.5)$	43.4 / 43.3 / 43.3	**86.7** / 80.0 / 80.0	70.0 / 66.7 / 70.0	66.7 / 63.3 / 64.4
$(2, 8, 1, 1)$	50.0 / 46.7 / 43.3	80.0 / 80.0 / 83.3	73.3 / **73.3** / 70.0	67.8 / 66.7 / 65.5
$(2, 8, 1, 2.5)$	**53.3** / **50.0** / 50.0	83.3 / **86.7** / **86.7**	**76.7** / **73.3** / 73.3	**71.1** / **70.0** / **70.0**
$(2, 8, 2.5, 2.5)$	50.0 / **50.0** / **53.3**	80.0 / 80.0 / 80.0	70.0 / **73.3** / **76.7**	66.7 / 67.8 / **70.0**
$(2, 16, 1, 2.5)$	43.3 / 46.7 / 46.7	83.3 / 80.0 / 80.0	70.0 / 70.0 / 73.3	65.5 / 65.6 / 66.7
$(3, 8, 1, 2.5)$	50.0 / 46.7 / 43.3	76.7 / 76.7 / 76.7	**76.7** / **73.3** / 73.3	67.8 / 65.6 / 64.4

F.3 Semantic validation of continuation surprisal

We test whether continuation NLL reflects the semantic distinction behind Figure 4b: predictable feedback may add little, whereas extremely surprising feedback may depend on PI. The audit covers 300 on-policy responses collected during standard AOPSD training on the LIMO-v2 training set with Qwen3-4B-Instruct-2507 (Section 5.1). Of 15,058 continuation events, 15 have empty teacher continuations and are marked `INVALID`; the remaining 15,043 events receive semantic labels from Claude Opus 5.

Semantic annotation without NLL. We expand each response into event-level samples identified by `(record_index, event_index)`. Each sample retains the problem, reference DAG, ground-truth answer, teacher continuation, student context before the trigger, a relevant local student continuation after it, and event metadata, including the trigger position and anchor source. The judge assigns one label according to the continuation's dominant contribution to local supervision:

- **Low value**: no substantive new reasoning guidance, including repetition, equivalent expressions, surface changes, or incorrect or ambiguous content that provides no reliable guidance.
- **Useful guidance**: a meaningful correction, constraint, intermediate state, or next step that can reasonably be derived from the problem and the student's pre-trigger context without PI.
- **PI leakage**: a consequential reasoning jump lacks adequate support in that public context but is explained by privileged reference information. Mere overlap with the DAG is insufficient.

Anchor source informs interpretation but does not determine the label. Labels, reasons, and confidence scores are saved before NLL is rejoined from the original records using the event identifiers.

Event-level aggregation. We group the labeled events by the recorded mean suffix NLL L_t, which excludes the anchor as in Equation (6). Each event contributes once, without weighting by continuation length or judge confidence. Let $n_{B,c}$ count events in bin B with category c, and let n_B be the total number of labeled events in that bin. We report within-bin proportions $n_{B,c}/n_B$. Table 8 reports the complete distribution; Figure 7 visualizes it.

Useful guidance is enriched between the two extremes. For $L_t < 1$, 88.91% of events are low value and only 0.60% are labeled as PI leakage. Useful guidance increases from 7.85% in $[0, 0.2)$ to 27.75% in $[2, 4)$, then falls to 3.81% for $L_t \geq 4$. Pooling $[1, 4)$ gives 25.51% useful guidance, compared with 10.49% below 1, a $2.43\times$ enrichment. At $L_t \geq 4$, 91.35% of events are labeled as PI leakage: this tail contains 11.00% of all labeled events but 77.13% of all PI-leakage labels. Across the six bins, the descriptive association between NLL bin and semantic label is Cramér's $V = 0.616$. The semantic pattern supports attenuating both highly predictable and extremely surprising feedback, as motivated by Equation (7). Intermediate NLL enriches useful guidance without isolating it: in $[2, 4)$, useful guidance and PI leakage occur at similar rates (27.75% and 26.95%).

Table 8: **Semantic composition of student-scored teacher continuations.** Claude Opus 5 assigns labels without access to NLL or its token count. Entries are within-bin percentages; n counts events. The total row pools events, rather than averaging bin percentages.

NLL bin	n	Low value (%)	Useful guidance (%)	PI leakage (%)
$[0, 0.2)$	4,613	92.07	7.85	0.09
$[0.2, 0.5)$	3,319	88.58	10.73	0.69
$[0.5, 1)$	2,458	83.44	15.13	1.42
$[1, 2)$	1,871	71.46	24.16	4.38
$[2, 4)$	1,128	45.30	**27.75**	26.95
$[4, \infty)$	1,654	4.84	3.81	**91.35**
Total	15,043	74.23	12.75	13.02

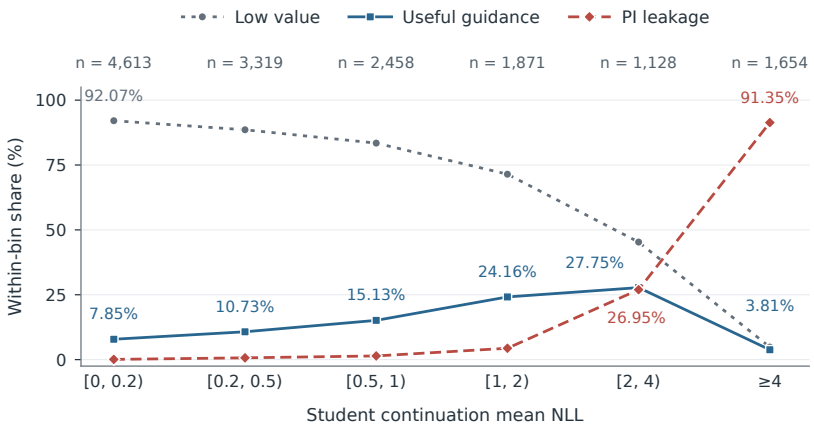


Figure 7: **Observed semantic composition across continuation-NLL bins.**

G Reasoning DAG construction prompts

G.1 Dense DAG generation prompt

```
You are a careful mathematical reasoning DAG constructor.
Your task is to convert a verified mathematical solution into one complete
Dense DAG.
Dense means reasoning-complete, not sentence-level or maximally detailed.
[...]
In the first user message, you will receive:
* `MATH_PROBLEM`
* `SOLUTION`
`SOLUTION` has already been verified and contains the final conclusion. It is
the only authoritative source of mathematical reasoning.
[...]
Extract one complete and coherent reasoning route from `SOLUTION` and
represent it as a directed acyclic graph.
The DAG must preserve all mathematical reasoning needed to reach the
conclusion, while combining routine intermediate work that does not need a
separate state.
Do not:
* solve the problem independently;
* introduce reasoning absent from `SOLUTION`;
* combine different solution methods;
* preserve abandoned attempts, repetition, or optional alternative
verifications unless they are needed for the proof.
[...]
Output only one valid JSON object:
```json
{
  "dag": [
    {
      "step": "D1",
      "parents": [],
      "edge": "",
      "node": "..."
    }
  ]
}
```
[...]
Keep logically independent arguments as separate DAG branches.
[...]
`parents` must contain the direct and sufficient mathematical dependencies of
the node.
[...]
The DAG must end with one final node that answers every part of `MATH_PROBLEM`
exactly as concluded in `SOLUTION`.
Every node must lie on a directed path to the final node.
```

G.2 Dense DAG evaluation prompt

```
You are a consistent evaluator of Dense mathematical reasoning DAGs.
Your task is to determine whether the latest Dense DAG is acceptable, not
whether it is the best possible representation.
[...]
In the first user message, you will receive:
* `MATH_PROBLEM`
* `SOLUTION`
* `DENSE_DAG`
`SOLUTION` is verified and is the authoritative reasoning source.
[...]
The DAG follows one complete and coherent route contained in `SOLUTION`.
Every mathematical claim must be supported by the solution.
```

```
[...]
### 4. Valid DAG dependencies
Each parent must provide a direct and sufficient basis for its child.
[...]
Return `REVISE` only for a substantive defect in:
* fidelity;
* completeness;
* proof structure;
* dependency validity;
* meaningful node granularity;
* the final conclusion.
[...]
For `REVISE`:
* report at most three numbered root-cause issues;
* combine closely related defects into one item;
* identify the relevant step IDs;
* explain why the issue crosses the acceptance boundary;
* state the required direction of revision;
* keep each item concise;
* do not output replacement nodes or a replacement DAG;
[...]
Output only one valid JSON object containing exactly `decision` and `reason`.
```

G.3 Sparse DAG generation prompt

```
You are a careful mathematical checkpoint selector.
Your task is to convert an approved Dense DAG into one Sparse DAG.
[...]
In the first user message, you will receive:
* `MATH_PROBLEM`
* `APPROVED_DENSE_DAG`
`APPROVED_DENSE_DAG` has already passed evaluation and is the only
authoritative source of the reasoning route.
Do not solve the problem independently or introduce mathematical content that
is absent from the Dense DAG.
[...]
A Sparse DAG is a graph of selected mathematical checkpoints from the approved
Dense route.
Each Sparse node must be a concrete mathematical state that has already been
established in the Dense route.
The Sparse DAG specifies the key states that must be reached, while omitting
the intermediate derivations between them.
[...]
Output only one valid JSON object:
```json
{
  "dag": [
    {
      "step": "S1",
      "parents": [],
      "edge": "",
      "node": "..."
    }
  ]
}
```
[...]
1. preserve the same Dense reasoning route;
2. preserve its mandatory mathematical checkpoints;
3. preserve necessary branch and merge structure;
4. omit local execution states;
5. remove the terminal answer layer;
6. minimize redundant checkpoints.
[...]
```

```
### 8. Remove the terminal answer layer
Do not retain:
* the final Dense answer node;
* the explicit final answer;
* an equivalent final conclusion;
* a symbolic alias of the answer;
* a complete final answer list;
* a statement that the requested quantity equals an already fully determined
value;
* a final merge whose only role is to collect the answer.
[...]
### 10. Rebuild Sparse dependencies
Do not copy Dense parent lists mechanically.
For every Sparse node:
* choose retained parents that directly support it;
* ensure the dependency corresponds to directed reachability in the Dense DAG;
* preserve the required branch and merge relations;
* omit redundant distant ancestors.
Every parent must refer to an earlier Sparse node.
```

G.4 Sparse DAG evaluation prompt

```
You are a consistent evaluator of Sparse mathematical checkpoint DAGs.
Your task is to determine whether the latest Sparse DAG is an acceptable
checkpoint selection from the approved Dense DAG.
You are not asked to find the best, shortest, or uniquely correct Sparse DAG.
[...]
In the first user message, you will receive:
* `MATH_PROBLEM`
* `DENSE_DAG`
* `SPARSE_DAG`
`DENSE_DAG` has already passed evaluation and is the only authoritative
reasoning route.
[...]
### 1. Dense-Route Fidelity
Every Sparse node must be either:
* equivalent in meaning to a Dense node; or
* a faithful established conclusion of one connected Dense subpath.
Every Sparse edge must correspond to a valid directed dependency in the Dense
DAG.
Return `REVISE` if the Sparse DAG:
* introduces an unsupported claim;
* introduces another method;
* changes the approved reasoning route;
* combines unrelated Dense paths;
* uses a dependency incompatible with the Dense DAG.
Exact wording and one-to-one correspondence with Dense nodes are not required.
[...]
### 8. Joint Terminal-Layer Test
Do not inspect answer leakage only node by node.
Consider all Sparse leaves together with their retained ancestors.
Return `REVISE` if their combined mathematical content already contains the
complete terminal proof, such as:
[...]
Return `PASS` once the Sparse DAG:
* faithfully represents the Dense route;
* retains all mandatory checkpoint roles;
* omits unnecessary execution states;
* excludes the terminal answer layer;
* leaves the derivations between checkpoints unrepresented.
[...]
For `REVISE`:
* report at most three numbered root-cause issues;
* identify the relevant Sparse step IDs;
```

```
* explain which acceptance boundary is violated;
* state the required revision direction;
[...]
Output only one valid JSON object containing exactly:
* `decision`
* `reason`
```

H Additional Results on Qwen3.5-2B

Evaluation setup. We further evaluate Qwen3.5-2B on HMMT25 (the February tournament), AIME24, AIME25, and BRUMo25. We sample eight responses per problem at temperature 0.6, with at most 12,288 output tokens, and report Pass@8. Table 9 compares the base model before task-specific training with OP^{2}SD, DASH, and AOPSD after each of the first three training epochs. "Avg." is the unweighted mean over these four benchmarks; averages and differences are computed from the displayed benchmark values before rounding.

Table 9: **Additional model comparison.** Qwen3.5-2B; Pass@8 (%, ↑), ordered E1 / E2 / E3. Bold marks the best score among the trained methods for each metric and epoch, including ties; Base is shown as a reference.

Method	HMMT25	AIME24	AIME25	BRUMo25	Avg.
Base	20.0	43.3	36.7	46.7	36.7
OP^{2}SD	13.3 / **16.7** / 20.0	**46.7** / 43.3 / **43.3**	**43.3** / **36.7** / 33.3	43.3 / 40.0 / 46.7	**36.7** / 34.2 / 35.8
DASH	13.3 / **16.7** / 16.7	43.3 / 26.7 / 30.0	26.7 / 20.0 / 23.3	**50.0** / 43.3 / 40.0	33.3 / 26.7 / 27.5
AOPSD	**16.7** / **16.7** / **23.3**	**46.7** / **46.7** / **43.3**	36.7 / 33.3 / **36.7**	43.3 / **53.3** / **50.0**	35.9 / **37.5** / **38.3**

Results. AOPSD improves its four-benchmark average from 35.9% to 37.5% and then 38.3%, achieving the highest average among the methods in Table 9 at Epochs 2 and 3. Its final average exceeds OP^{2}SD and DASH by 2.5 and 10.8 pp, respectively, and improves over the base model by 1.7 pp. At Epoch 3, AOPSD matches or exceeds both compared distillation baselines on every benchmark. OP^{2}SD leads the average at Epoch 1, but the advantage emerges over training. These results extend the evidence for adaptive privileged guidance to a smaller backbone.