GAINS AND COLLAPSE IN ON-POLICY DISTILLATION: A REINFORCEMENT LEARNING PERSPECTIVE

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy distillation (OPD) has become an important approach to language model post-training. However, despite its performance gains, OPD can also collapse into excessively long and repetitive generation, and the mechanism underlying these divergent outcomes remains poorly understood. We explain these outcomes through a reinforcement learning perspective: the teacher implicitly rewards student behaviors, even those it rarely exhibits itself. From this perspective, our experiments show that OPD improves performance without expanding the student's capabilities. When the implicit reward model is reliable, OPD makes correct responses easier to sample. In contrast, when the preference misaligns with quality, reward hacking happens: the implicit reward model amplifies overlong, repetitive student rollouts, even though it rarely generates such text itself. Guided by this diagnosis, we find that masking unhealthy responses during training and using SFT initialization can each effectively mitigate the collapse. Together, these findings show that OPD amplifies student behaviors favored by the teacher's implicit feedback, shifting the focus from how well the teacher generates to how reliably it evaluates student rollouts.

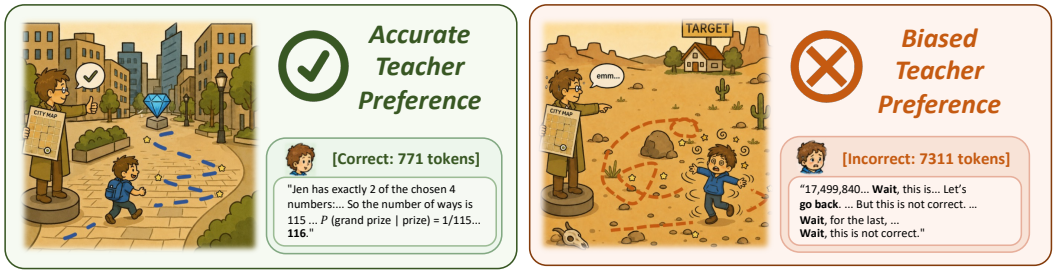


Figure 1: **OPD amplifies what the student already samples.** The teacher scores student-generated trajectories rather than supplying training trajectories of its own; the examples illustrate how its preferences can amplify either coherent or degenerate student behavior.

1 INTRODUCTION

On-policy distillation (OPD) has become an important approach for developing frontier language models and has been adopted in large-scale post-training pipelines (DeepSeek-AI et al., 2026; Kimi Team et al., 2026; Xiao et al., 2026; Yang et al., 2025; Zeng et al., 2026). At each iteration, the current student samples its own responses, and the teacher supervises it through token-level KL divergence at student-generated prefixes (Agarwal et al., 2024; Gu et al., 2024). While prior work has demonstrated the potential of OPD to improve student performance, recent studies have reported unstable training and degenerate behaviors, including excessive response length and repetition (Li et al., 2026; Zhu et al., 2026). It remains unclear what OPD actually learns during training, and why this learning process can sometimes lead to collapse.

In this paper, we view OPD from an RL perspective, in which the teacher implicitly rewards the student's own responses. We investigate OPD on mathematical tasks using different types of teachers, including RL-trained reasoning models and larger general-purpose models. Our experiments show

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.

that **OPD does not directly extend the student's capability. Instead, it changes the relative probability of responses the student already produces according to the teacher's preferences.** When these preferences align with response quality, this reweighting can improve sampling efficiency; when they do not, it can lead to reward hacking, with the student increasingly producing responses that receive high implicit rewards despite poor quality. We substantiate this view through three findings.

First, we show that successful OPD reinforces correct responses the student can already generate. We compare the initial and trained students across sampling budgets, combining pass@k evaluation with additional sampling and manual audits. Although OPD substantially improves single-sample accuracy(pass@1), the gains progressively diminish as the sampling budget increases. At the same time, we find no expansion of the observed solvable set: problems solved after OPD can also be solved by the initial student given sufficient sampling. Together, these results suggest that OPD makes existing capabilities easier to elicit rather than expanding the student model's problem-solving capabilities.

In contrast, we find that the observed OPD collapse reflects reward hacking: training reinforces poor-quality responses favored by the teacher. We compare a successful OPD run with a reproduction of the collapsed setting reported by Li et al. (2026). Despite their sharply different outcomes, both runs converge normally and concentrate on responses with low negative log-likelihood under the initial student, suggesting that collapse is not simply a failure of optimization or convergence. In the collapsed run, the student produces increasingly overlong and repetitive responses. At the same time, its response distribution shifts markedly toward responses preferred by the teacher over those the teacher disfavors, indicating that the student is fitting the teacher's preference signal increasingly well. The student thus learns the teacher's preferences even as its generation quality deteriorates.

Building on this diagnosis, we mitigate OPD collapse with a standard remedy for reward hacking: limiting the reinforcement of pathological responses. Before training, higher implicit rewards favor correct and concise responses in the successful setting, but overlong and repetitive responses in the collapsed setting, with both types of behavior later amplified during training. Notably, the teacher itself rarely generates such pathological responses, showing that generation quality does not reliably reflect feedback quality on student rollouts. Keeping the teacher fixed, masking rollouts that reach the generation limit mitigates degeneration and improves average evaluation accuracy by 3.08 percentage points, and SFT warmup provides a complementary solution by improving the quality of the student's initial rollout distribution. Together, these results show that collapse can be mitigated by controlling which student rollouts are available for reinforcement.

Our contributions are summarized as follows:

- We interpret OPD as an RL-like process with the teacher acting as an implicit reward model, linking its performance gains to improved sampling efficiency.
- We identify reward hacking in the observed collapse: the student fits the teacher's preferences increasingly well even as its generation quality deteriorates.
- We trace the collapse to a biased teacher reward signal. Limiting the reinforcement of this biased signal through masking, or reducing exposure to pathological rollouts through SFT warmup, mitigates collapse.

2 On-policy distillation as implicit reward optimization

We formulate the sampled-token reverse-KL variant of on-policy distillation (OPD) studied in this paper as policy optimization with a teacher-derived reward. This formulation makes explicit what the teacher scores, how those scores guide student updates, and why their reliability must be examined on student-generated trajectories.

2.1 On-policy teacher supervision

Let $q \sim \mathcal{D}$ be a training question with prompt x_q. We denote the fixed teacher by T, the initial student by S_{INIT}, the trainable student by S_θ, and the final trained student by S_{OPD}. At each rollout iteration, the current student policy $\bar{S} = S_{\bar{\theta}}$ generates a response $y = (y_1, \ldots, y_L)$. The teacher evaluates tokens conditioned on the resulting prefixes $h_t = (x_q, y_{<t})$.

Table 1: OPD settings used in our analysis. Each setting is named after its teacher.

Teacher	Student initialization	Training data
JustRL-1.5B	DeepSeek-R1-Distill-Qwen-1.5B	DeepMath-103K (≥ 6)
DeepScaleR-1.5B-Preview	DeepSeek-R1-Distill-Qwen-1.5B	DeepMath-103K (≥ 6)
Qwen3-4B	Qwen3-1.7B-Base	OpenThoughts3

Let $d^{\bar{S}}$ denote the prefix distribution induced by the rollout policy. The conditional reverse-KL objective is

$$\mathcal{L}_T(\theta; \bar{S}) = \mathbb{E}_{h\sim d^{\bar{S}}}\left[D_{\mathrm{KL}}\Big(\pi_{S_\theta}(\cdot \mid h) \,\|\, \pi_T^{(\tau_T)}(\cdot \mid h)\Big)\right], \tag{1}$$

where τ_T is the teacher scoring temperature. We hold the rollout-prefix distribution fixed during each optimization step and refresh it through subsequent student rollouts. The student thus determines where supervision is evaluated, while the teacher supplies the target token probabilities at those states.

2.2 The teacher as an implicit reward model

Reward interpretation. By expanding the KL divergence in Equation (1), we can recast the objective into a maximum-entropy policy objective over sampled prefixes in Equation 2:

$$\mathcal{L}_T(\theta; \bar{S}) = -\mathbb{E}_{h\sim d^{\bar{S}}}\Big[\mathbb{E}_{a\sim\pi_{S_\theta}(\cdot|h)}\big[r_T(h,a)\big] \; + \mathcal{H}(\pi_{S_\theta}(\cdot \mid h))\Big], \tag{2}$$

where $r_T(h,a) = \log \pi_T^{(\tau_T)}(a \mid h)$ and H denotes entropy. Minimizing the conditional reverse KL is therefore equivalent to maximizing the expected teacher reward with entropy regularization on the sampled prefix distribution. This is the token-level reward interpretation underlying policy-gradient implementations of OPD (Lu & Thinking Machines Lab, 2025).

Policy-gradient update. The reward interpretation also yields the sampled-token update. For each rollout token, define

$$A_t^T = r_T(h_t, y_t) - \log \pi_{\bar{S}}(y_t \mid h_t). \tag{3}$$

Writing $A^T(h,a) = r_T(h,a) - \log \pi_{\bar{S}}(a \mid h)$, the unclipped estimator at a fixed prefix h satisfies

$$\begin{aligned} &\mathbb{E}_{a\sim\pi_{\bar{S}}(\cdot|h)}\big[A^T(h,a)\nabla_\theta \log \pi_{S_\theta}(a \mid h)\big]\big|_{\theta=\bar{\theta}} \\ &\quad = -\nabla_\theta D_{\mathrm{KL}}\Big(\pi_{S_\theta}(\cdot \mid h) \,\|\, \pi_T^{(\tau_T)}(\cdot \mid h)\Big)\Big|_{\theta=\bar{\theta}}. \end{aligned} \tag{4}$$

In practice, we treat A_t^T as a stop-gradient advantage in a PPO-style clipped surrogate and add reference-KL regularization toward S_{INIT}. The teacher score r_T and the update weight A_t^T are distinct: the latter includes the student log-probability term arising from entropy regularization. These updates use immediate token-level signals on student-generated prefixes. When most advantages are negative, the main effect is to reduce token probabilities by different amounts. More negative advantages lead to stronger pressure to reduce the probability, while less negative advantages lead to weaker pressure.

2.3 Experimental settings

We study three teacher–student settings, summarized in Table 1. The JustRL and DeepScaleR settings share the same student initialization and training data. Their teachers, JustRL-1.5B (He et al., 2025) and DeepScaleR-1.5B-Preview (Luo et al., 2025), were obtained through RL from DeepSeek-R1-Distill-Qwen-1.5B (Guo et al., 2025), which also initializes our student. Both settings use the level ≥ 6 subset of DeepMath-103K and a 16,384-token rollout limit. The Qwen3-4B setting follows Li et al. (2026), using Qwen3-4B to supervise Qwen3-1.7B-Base on OpenThoughts3 prompts (Guha et al., 2025) with an 8,192-token rollout limit. The teacher remains fixed in all settings. We train for 100 rollout iterations using the Slime training framework (Zhu et al., 2025), with a reference-KL coefficient of 0.001. We evaluate mathematical reasoning on AIME24–26, reporting mean@4 for routine evaluation and using 256 responses per problem for the coverage and sampling efficiency analyses. Full training and evaluation protocols are provided in Appendix A.1.

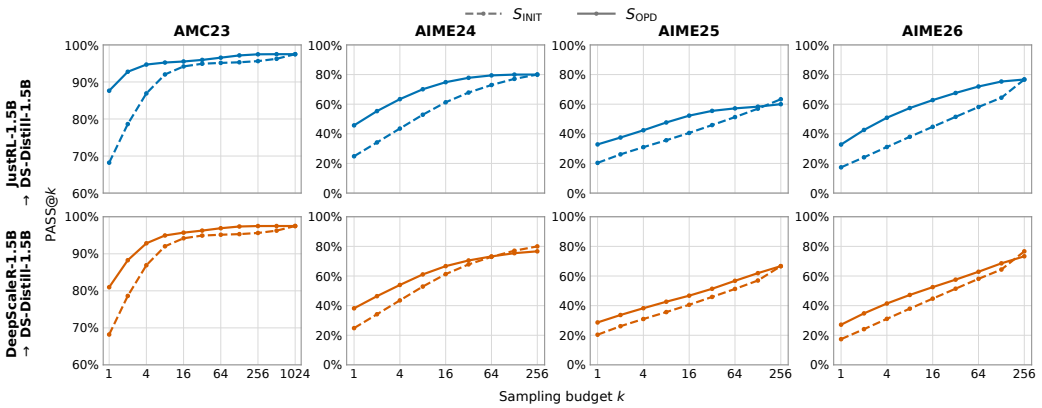


Figure 2: **PASS@k curves of S_{INIT} and S_{OPD}.** Across AMC23 and AIME24–26, OPD gains are largest at small k and shrink as the sampling budget grows. AIME curves extend to $k = 256$, while AMC23 extends to $k = 1024$.

3 Does OPD Expand Student's Capability Boundary?

The implicit-reward perspective suggests that successful OPD can improve performance by reinforcing correct student responses. We examine this successful case: does teacher feedback make problems already solvable by the initial student easier to solve? We compare the initial student S_{INIT} with the trained student S_{OPD} using pass@k across increasing sampling budgets, audited problem coverage, and per-problem success rates.

Gains diminish as the sampling budget grows. We compare the initial student S_{INIT} with the trained student S_{OPD} under both teachers, JustRL-1.5B and DeepScaleR-1.5B-Preview, on AIME24–26 and AMC23. We compute PASS@k from each response pool for $k \in \{1, 2, 4, 8, 16, 32, 64, 128, 256\}$ on AIME and additionally $k \in \{512, 1024\}$ on AMC23, averaged across all test cases.

As shown in Figure 2, we observe a consistent trend under both teachers. When k is small (e.g., $k = 1$, equivalent to average-case accuracy), S_{OPD} clearly outperforms S_{INIT}. However, as k increases, S_{INIT} steadily catches up. At the maximum available k, PASS@k equals the observed coverage: the fraction of problems solved at least once in the response pool. This suggests that OPD improves accuracy under a small sampling budget, but does not improve the observed coverage at higher k.

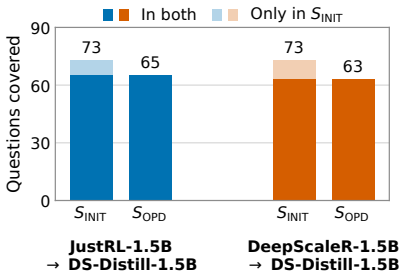


Figure 3: **Change in observed coverage from S_{INIT} to S_{OPD}.** No problem is solvable by S_{OPD} but not by S_{INIT} in our observation. Counts come from the audited pools with conditional extension to 1,024 responses.

No Observed Expansion in Problem Coverage After OPD. To distinguish new coverage from finite-sample misses, we first identify the problems solved by S_{OPD} but not by S_{INIT} within the original 256-response pools of AIME24-26. For each of these problems, we draw 768 additional responses from S_{INIT}, bringing its pool to 1024 responses. Manual review checks derivation validity and corrects verifier misses in the existing outputs.

After additional sampling and manual review, no problem remains solved only by S_{OPD}: candidate cases are resolved either by finding a valid S_{INIT} solution or by excluding an invalid S_{OPD} derivation. The audited S_{INIT} set contains 73 solvable problems, against which S_{OPD} covers 65 under JustRL-1.5B and 63 under DeepScaleR-1.5B-Preview. These counts are larger than the $k = 256$ endpoints of Figure 2, which are computed before the audit. Detailed coverage analyses are provided in Appendix B. These indicate that OPD does not induce an observed expansion of the solvable set.

Already-solvable problems become easier to solve. To identify the source of OPD's small-k gains, we compare per-problem success rates between S_{INIT} and S_{OPD}. For each problem q, let $\hat{p}_m(q)$ denote the fraction of correct responses among the 256 samples from model m.

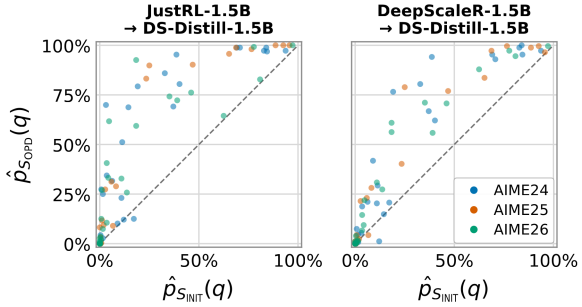


Figure 4: **Per-problem mean@256 of S_{INIT} (x-axis) and S_{OPD} (y-axis).** OPD raises success rates on problems already solvable by S_{INIT}. The dashed line marks equal score.

As shown in Figure 4, the problem-level distribution is strongly shifted above the diagonal under both teachers. OPD increases the per-problem success rate for 90.6% of the problems already solvable by S_{INIT} with JustRL-1.5B as the teacher, and for 84.4% with DeepScaleR-1.5B-Preview. Averaged over these problems, the increase in mean@256 is 22.6% and 14.7%, respectively.

Problems unimproved by OPD are those that S_{INIT} could hardly solve, and no high-performing problem drops to near-zero accuracy (lower-right corner). Some even jump from low to high success rates (upper-left corner), showing that OPD can make problems the initial student rarely solves much easier to solve. Thus, OPD primarily eases sampling of correct answers on already-solvable problems, which accounts for the small-k performance gain.

Summary. In the two successful settings studied here, OPD improves sampling efficiency on problems already solvable by the initial student. Its positive accuracy gains disappear at the largest sampling budgets tested, and no validated OPD-only problem remains after the targeted AIME coverage audit. These findings are consistent with the teacher acting as an implicit reward model whose feedback reinforces correct student behavior, making successful responses easier to sample.

4 Reward Hacking in OPD: Convergence to a Biased Reward Signal

The preceding analysis shows that OPD improves performance without expanding the set of problems the student can solve. Yet OPD training does not always succeed: some teacher-student pairs produce collapsed behaviors instead. From a diagnostic perspective, we then examine whether the collapse constitutes reward hacking: optimization succeeding at the wrong target. We first verify that the optimization converges normally, then characterize the behavioral shifts that emerge during training.

4.1 Collapsed OPD Training Still Converges Normally

Our analysis starts by comparing a reported failure case of OPD with a successful one. Prior work (Li et al., 2026) reports that OPD can fail when the teacher and student distributions are substantially mismatched, attributing the resulting degradation to this large distribution gap. We therefore compare the full training process of the two settings: the **JustRL setting** (JustRL-1.5B → DS-Distill-1.5B) where training improves the student, and the **Qwen3-4B setting** (Qwen3-4B → Qwen3-1.7B-Base, without warm-up) where training collapses. The loss and advantage in the Qwen3-4B setting converge as smoothly as those in the JustRL setting, with no qualitative difference in the shape of the curves.

We measure response concentration using the Min-10NN distance (Zhang et al., 2026), which averages the ten smallest nearest-neighbor distances among responses to the same problem; lower values indicate greater similarity (described in detail in Appendix C). The distance keeps decreasing during training in both settings: mildly in the JustRL setting, and much more sharply in the Qwen3-4B one. Both students shrink toward a narrower space of behaviors, a phenomenon also reported in other work on on-policy RLVR post-training (Yue et al., 2025; Zhao et al., 2025; Dang et al., 2025).

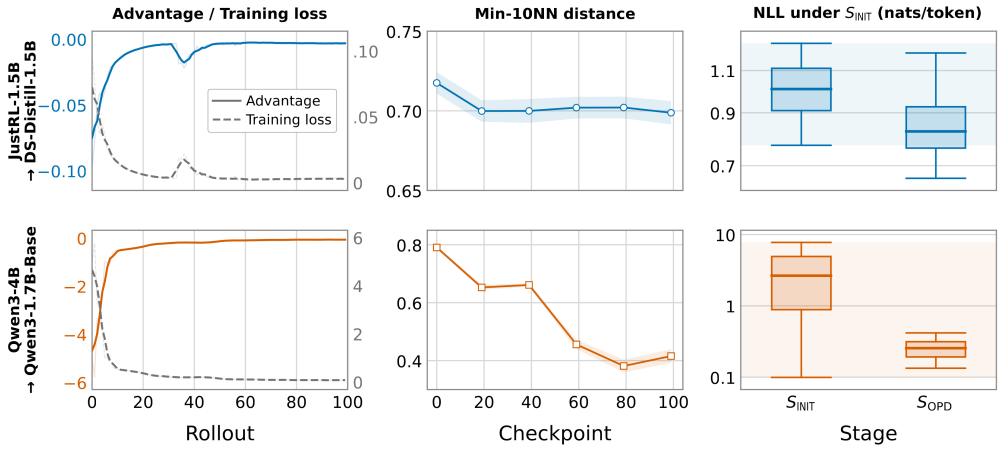


Figure 5: **Training process of the JustRL (top) and Qwen3-4B (bottom) settings.** Optimization converges normally in both settings, and the trained student's responses remain highly likely under S_{INIT}. Columns: the advantage and training loss; the Min-10NN distance over six checkpoints; the NLL under S_{INIT} of responses from S_{INIT} and S_{OPD}.

If training exploits the reward rather than learning from it, the selected behaviors should already be accessible to S_{INIT} rather than genuinely novel. To examine this, we compute the NLL under S_{INIT} for responses generated by the trained student and compare it with the NLL of S_{INIT}'s own responses to the same problems. Figure 5 summarizes these diagnostics; the right panel reports this NLL comparison. In both settings, the trained student's responses lie in the low NLL region of S_{INIT}, which indicates that they are already familiar to S_{INIT}.

4.2 The Student Faithfully Fits the Reward Signal, Even When the Teacher Is Hacked

If the collapse constitutes reward hacking, the trained student should have successfully internalized the teacher's preferences, but those preferences should be misaligned with solution quality.

Model	Acc. (%)	Len. (k)	Trunc. (%)	Rep. (%)
JustRL-1.5B → DS-Distill-1.5B				
Teacher	39.3	9.30	13.8	0.0
S_{INIT}	20.9	11.19	12.5	0.0
S_{OPD}	37.1 (+16.2)	9.72	16.5	0.0
Qwen3-4B → Qwen3-1.7B-Base				
Teacher	19.6	3.09	2.1	0.0
S_{INIT}	0.2	1.18	4.1	4.2
S_{OPD}	5.4 (+5.2)	**8.16**	**99.4**	**38.0**

Table 2: **Endpoint generation characteristics on AIME24–26.** Model pairs denote teacher → student. Bold values mark length-limit saturation and severe repetition.

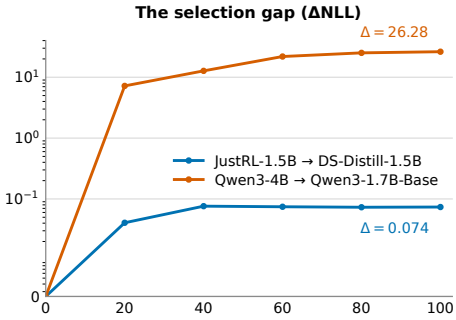


Figure 6: **Selection gap before and after training.** Both students learn to discriminate teacher-preferred responses, with the Qwen3-4B setting showing overfitting.

Endpoint behaviors are pathological. Table 2 compares the free generations of the teacher, S_{INIT}, and S_{OPD} across both settings. In the JustRL setting, the trained student's behaviors closely match the teacher's: accuracy improves by 16.2 points, and no severe repetition is detected. In contrast, the Qwen3-4B setting exhibits a dramatically different outcome. Although accuracy improves modestly (+5.2 points), nearly all responses hit the 8k generation limit (99.4%), and 38% exhibit severe repetition—far worse than either the teacher (2.1% truncation, 0% repetition) or S_{INIT} itself. The trained student has not learned to reason like the teacher; instead, it has concentrated probability mass on a narrow subset of pathological behaviors.

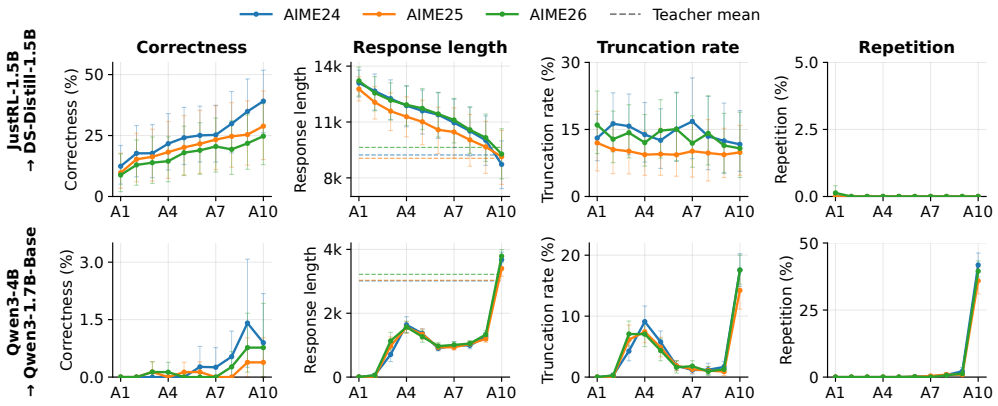


Figure 7: **Teacher advantage over S_{INIT} responses from A1 (lowest advantage) to A10 (highest advantage).** The JustRL teacher (top) assigns higher advantage to correct and shorter responses; the Qwen3-4B teacher (bottom) assigns higher advantage to overlong, repetitive ones. Dashed lines mark the teacher's own mean length.

The trained student discriminates teacher-preferred responses. We measure how the trained student scores a fixed set of S_{INIT} responses. We compare the 10% of responses the teacher prefers most with the 10% it prefers least, as ranked by teacher advantage. For each response, $\Delta\text{NLL} = \text{NLL}_{S_t} - \text{NLL}_{S_0}$ measures the change in the student's NLL during training. The *selection gap* Γ is the mean ΔNLL of the least-preferred 10% minus that of the most-preferred 10%. A positive Γ indicates that training favors the teacher's most-preferred responses over its least-preferred ones.

Figure 6 shows whether this degradation reflects learned teacher preference rather than random drift. Both students increase their discrimination of teacher-preferred responses: JustRL's gap grows modestly from near zero to 0.07, while the Qwen3-4B setting's gap increases dramatically to approximately 26. Combined with the endpoint generation table, this reveals that the student in the Qwen3-4B setting has *overfit* to the subset of S_{INIT} responses that receive high teacher advantage but exhibit pathological behaviors.

5 Diagnosing and Mitigating Reward Hacking

To address reward hacking, prior work identifies which training samples exploit the reward model, then removes or reweights them during optimization (Liu et al., 2025a; Beigi et al., 2026; Zhang et al., 2025). Following this strategy, we first diagnose which S_{INIT} responses the teacher assigns high advantage to, then test whether masking pathological responses or reinitializing from a cleaner distribution can prevent the collapse.

5.1 The Teacher's Reward Signal Can Be Misaligned with Response Quality

To characterize the responses favored by the teacher before training, we rank all sampled S_{INIT} responses by their mean teacher advantage and divide them into ten equally sized groups, A1–A10. Each group contains 10% of the responses, with A1 containing the lowest-advantage responses and A10 the highest. Figure 7 reports correctness, response length, generation-cap hit rate, and severe-repetition rate across advantage deciles. In the JustRL setting, teacher advantage is aligned with response quality: correctness increases toward A10, response length decreases toward the teacher's own average, and no severe repetition is detected. The Qwen3-4B setting shows the opposite pattern. The highest-advantage responses are not the most reliable ones: their correctness remains close to zero, whereas their length, truncation rate, and repetition rate increase substantially across advantage deciles. Although these pathological responses are rare under S_{INIT}, the teacher assigns them high advantage, causing OPD to amplify them during training. We also find similar failure pattern when the teacher is replaced as Qwen3-8B and Qwen3-30B-A3B. Appendix E presents three high-advantage S_{INIT} responses that exhibit excessive length and mechanical repetition.

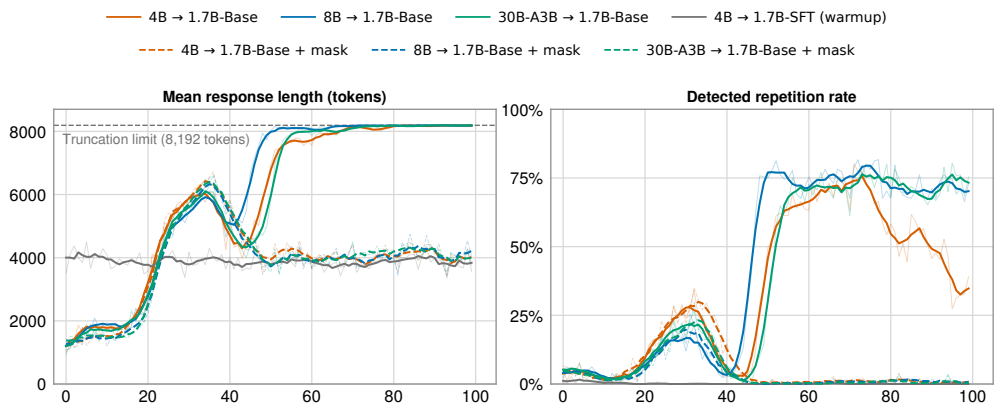


Figure 8: **Student-rollout characteristics during training.** All the masking settings recover from the collapse, whereas Qwen3-4B-warmup avoids it from the beginning.

5.2 Candidate-Pool Interventions Mitigate Reward Hacking

The preference bias discussed previously is already visible in responses sampled from S_{INIT}, before any OPD update takes place. This suggests that the collapse may depend on which responses are supplied to the teacher for supervision. To test this hypothesis, we intervene on the candidate pool in the Qwen3-4B setting while keeping the teacher, training data, and optimization recipe fixed. We consider two interventions: (1) The **Qwen3-4B-masked setting**: masking the loss on responses that reach the generation length limit. (2) The **Qwen3-4B-warmup setting**: changing only the student initialization, from Qwen3-1.7B-Base to an SFT-initialized student released by Li et al. (2026). Appendix F provides the training details for both interventions.

Masking recovers from collapse, while warmup avoids it. Figure 8 shows how response length, truncation, and repetition evolve over the course of training. The original Qwen3-4B setting eventually enters a persistent collapsed state, where responses become increasingly long, truncated, and repetitive. For the Qwen3-4B-masked setting, the student model exhibits few abnormal behaviors in the early training phase, so its learning trajectory closely aligns with that of the original Qwen3-4B. As the undesirable behaviors accumulate, the effect of the masking intervention becomes clear: the Qwen3-4B-masked setting recovers, while the original setting collapses. On the other hand, the warmup setting never develops this failure mode. Warmup has been adopted as a remedy for OPD collapse (Li et al., 2026). The SFT-initialized student rarely produces the pathological responses favored by the teacher, leaving fewer such responses for OPD to reinforce. Together, these results suggest that the collapse is not irreversible; it depends on whether abnormal responses continue to receive reinforcement during training.

Table 3: **Masking improves average accuracy across teachers.** Accuracy (%) and the four-benchmark mean; teachers are Qwen3 models, and Base/SFT denote Qwen3-1.7B initializations.

Setting	AMC	AIME			Avg.
	23	24	25	26	
Base (pre-OPD)	2.41	1.67	0.00	1.67	1.44
4B → Base	29.82	3.33	3.33	4.17	10.16
+ mask	34.64	8.33	5.83	4.17	13.24
8B → Base	28.31	5.00	3.33	5.00	10.41
+ mask	32.83	5.00	2.50	3.33	10.92
30B-A3B → Base	28.61	6.67	2.50	4.17	10.49
+ mask	33.13	10.00	4.17	5.00	13.07
4B → SFT† (warmup)	38.86	9.17	10.00	7.50	16.38

Both interventions improve final accuracy. Table 3 extends the accuracy comparison to Qwen3-8B and Qwen3-30B-A3B teachers. Masking improves the average accuracy over AMC23 and AIME24–26 by 3.08, 0.51, and 2.58 percentage points for the 4B, 8B, and 30B-A3B teachers, respectively. These gains are not uniform across benchmarks: with the 8B teacher, AIME24 accuracy

†Warmup uses the Qwen3-1.7B-SFT checkpoint released by Li et al. (2026).

is unchanged while AIME25 and AIME26 accuracy decrease from 3.33% to 2.50% and from 5.00% to 3.33%, respectively. The 4B warmup setting starts from a stronger SFT student and achieves the best final average accuracy (16.38%), exceeding the corresponding Base-initialized OPD setting by 6.22 points. In the 4B masking setting, accuracy improves even though only approximately 10% of the sampled responses retain nonzero loss after abnormal responses are masked.

6 Related Work

Distillation paradigms and capability transfer. Knowledge distillation matches teacher distributions (Hinton et al., 2015) or trains on teacher sequences (Kim & Rush, 2016); for reasoning models, the supervision is teacher rationales and trajectories (Ho et al., 2023; Hsieh et al., 2023; Mukherjee et al., 2023; Mitra et al., 2023; Guo et al., 2025), which extend coverage beyond the base model (Yue et al., 2025). OPD instead supervises student-generated trajectories at student-visited states (Gu et al., 2024; Agarwal et al., 2024), with extensions to self-distillation (Zhao et al., 2026; Ye et al., 2026; He et al., 2026b), representation-level supervision (Yang et al., 2026a), multiple teachers (Ma et al., 2026; Yang et al., 2026b; Kimi Team et al., 2026), and cross-tokenizer transfer (He et al., 2026a; Wang et al., 2026a). These broaden teacher supervision but retain the reliance on student rollouts, over which our runs mainly redistribute probability rather than expanding the student's observed solvable set.

Failure modes and stabilization of OPD. OPD failures arise from mismatches among student rollouts, teacher supervision, and token-level objectives. Li et al. (2026) show that a stronger teacher need not help under reasoning-pattern mismatch; Fu et al. (2026) and Zhu et al. (2026) identify unreliable guidance on student prefixes and bias from approximate reverse-KL updates. Remedies include off-policy warm starts, data selection, stop-gradient objectives, trust regions (Xing et al., 2026), and prefix correction (Jiang et al., 2026). Rollout-signal feedback creates another failure class: Wang et al. (2026b) identify signal-correctness mismatch and length exploitation, while Luo et al. (2026) link length inflation to truncation and repetition collapse, motivating clipping, reference regularization, rollout mixing, and length control (He et al., 2026a; Heo et al., 2026). We instead view OPD as selecting among student-accessible behaviors by teacher preference, and analyze how biases in that preference affect training.

Rollout pathologies in on-policy post-training. On-policy optimization trains only on trajectories sampled from the current model and is therefore exposed to the pathologies of its own rollout distribution. For RLVR, pass@k and likelihood analyses show that training mainly redistributes probability over solution paths already accessible to the base model (Yue et al., 2025), and related studies find that reinforcement learning can amplify behaviors acquired during pretraining (Zhao et al., 2025) and reduce the diversity of reasoning trajectories (Dang et al., 2025). Rollouts can further carry degenerate patterns such as malformed formats, excessive length, truncation, and self-reinforcing repetition. Repetition can be reinforced by token co-occurrence effects (Yan et al., 2024), while RLVR systems counter rollout pathologies with format rewards (Guo et al., 2025), length-debiased advantage estimation (Liu et al., 2025b), and dynamic sampling with overlong filtering (Yu et al., 2025). OPD inherits this dependence on student rollouts but evaluates them through a teacher-derived signal; we show that degenerate student behaviors can bias the induced preference itself and thereby become preferentially retained during training.

7 Conclusion

We interpret on-policy distillation as an RL-like process in which the teacher acts as an implicit reward model for the student's own responses. Across our experiments on competition mathematics, OPD improves sampling efficiency without expanding the observed solvable set. Its gains primarily come from making correct responses already accessible to the initial student easier to sample. This view also helps explain why OPD can collapse despite normal convergence. In the collapsed setting, the student increasingly follows the teacher's preferences while producing longer and more repetitive responses. These responses receive high teacher advantage even before training, although the teacher rarely generates them itself. Masking the loss on truncated responses and initializing the student from an SFT checkpoint mitigate this failure while keeping the teacher fixed.

LIMITATIONS

Computational constraints limit our experiments to small models and mathematical reasoning tasks, where prior work has reported clear cases of OPD success and collapse. Whether our findings extend to larger models or other domains remains to be studied. Our interventions are designed specifically to address the overlong and repetitive generations observed in our experiments. We use them mainly to test our explanation of OPD collapse, rather than propose them as general solutions for OPD training. Finally, our coverage results are based on a finite number of samples. Although we use additional sampling and manual inspection to check apparently new solutions, we cannot rule out that larger sampling budgets or different decoding settings would uncover additional correct responses from the initial model.

AI USE STATEMENT

We used generative AI tools to polish the manuscript, assist with writing portions of the experimental code, and design the cartoon characters in the main figure. The authors take responsibility for the final manuscript, experimental implementation, and figures.

ETHICS STATEMENT

This work studies the reasoning behavior and training dynamics of language models using publicly available models and mathematical reasoning datasets. We do not identify ethical concerns specific to the experiments presented here beyond those associated with the underlying models and datasets.

REPRODUCIBILITY STATEMENT

All models and datasets used in this work are publicly available, either through open releases or accompanying prior publications. The appendix describes the training configurations, evaluation protocols, and analysis procedures. We plan to release all evaluation and analysis scripts to facilitate reproduction of our results.

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. URL https://arxiv.org/abs/2306.13649.

Mohammad Beigi, Ming Jin, Junshan Zhang, Qifan Wang, and Lifu Huang. Adversarial reward auditing for active detection and mitigation of reward hacking, 2026. URL https://arxiv.org/abs/2602.01750.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Xingyu Dang, Christina Baek, J Zico Kolter, and Aditi Raghunathan. Assessing diversity collapse in reasoning. In *Scaling Self-Improving Foundation Models without Human Supervision*, 2025.

DeepSeek-AI, Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chengyu Hou, Chenhao Xu, Chenze Shao, Chong Ruan, Conner Sun, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Donghao Li, Dongjie Ji, Erhang Li, Fang Wei, Fangyun Lin, Fangzhou Yuan, Feiyu Xia, Fucong Dai, Guangbo Hao, Guanting Chen, Guoai Cao, Guolai Meng, Guowei Li, Han Yu, Han Zhang, Hanwei Xu, Hao Li, Haofen Liang, Haoling Zhang, Haoming Luo, Haoran Wei, Haotian Yuan, Haowei Zhang, Haowen Luo, Haoyu Chen, Haozhe Ji, Hengqing Zhang, Honghui Ding, Hongxuan Tang, Huanqi Cao, Huazuo Gao, Hui Qu, Hui

Zeng, J Yang, JQ Zhu, Jia Luo, Jia Song, Jia Yu, Jialiang Huang, Jialu Cai, Jian Liang, Jiangting Zhou, Jiasheng Ye, Jiashi Li, Jiaxin Xu, Jiewen Hu, Jieyu Yang, Jin Chen, Jin Yan, Jingchang Chen, Jingli Zhou, Jingting Xiang, Jingyang Yuan, Jingyuan Cheng, Jingzi Zhou, Jinhua Zhu, Jiping Yu, Joseph Sun, Jun Ran, Junguang Jiang, Junjie Qiu, Junlong Li, Junmin Zheng, Junxiao Song, Kai Dong, Kaige Gao, Kang Guan, Kexing Zhou, Kezhao Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Wang, Leyi Xia, Li Zhang, Liang Zhao, Lihua Guo, Lingxiao Luo, Linwang Ma, Linyan Zhu, Litong Wang, Liyu Cai, Liyue Zhang, Longhao Chen, MS Di, MY Xu, Max Mei, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Mingxu Zhou, Minmin Han, Ning Wang, Panpan Huang, Panpan Wang, Peixin Cong, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qingyang Li, Qinyu Chen, Qiushi Du, Qiwei Jiang, Rui Tian, Ruifan Xu, Ruijie Lu, Ruiling Xu, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runqian Chen, Runqiu Yin, Runxin Xu, Ruomeng Shen, Ruoyu Zhang, Ruyi Chen, SH Liu, Shanghao Lu, Shangmian Sun, Shangyan Zhou, Shanhuang Chen, Shaofei Cai, Shaoheng Nie, Shaoqing Wu, Shaoyuan Chen, Shengding Hu, Shengyu Liu, Shiqiang Hu, Shirong Ma, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, Shuying Yu, Songyang Zhou, Tao Ni, Tao Yun, Tian Jin, Tian Pei, Tian Ye, Tianle Lin, Tianran Ji, Tianyi Cui, Tianyuan Yue, Tingting Yu, Tun Wang, W Zhang, WL Xiao, Wangding Zeng, Wei An, Weilin Zhao, Wen Liu, Wenfeng Liang, Wenjie Pang, Wenjing Luo, Wenjing Yao, Wenjun Gao, Wenkai Yang, Wenlve Huang, Wenqing Hou, Wentao Zhang, Wenting Ma, Xi Gao, Xiang He, Xiangwen Wang, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaokang Zhang, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingchen Liu, Xingkai Yu, Xingyou Li, Xinyu Yang, Xinyu Zhang, Xu Chen, Xuanyu Wang, Xuecheng Su, Xueyin Chen, Xuheng Lin, Xuwei Fu, YC Yan, YQ Wang, YW Ma, Yanfeng Luo, Yang Zhang, Yanhong Xu, Yanru Ma, Yanwen Huang, Yao Li, Yao Li, Yao Xu, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Qian, Yi Shao, Yi Yu, Yichao Zhang, Yifan Ding, Yifan Shi, Yijia Wu, Yiliang Xiong, Yiling Ma, Ying He, Ying Tang, Ying Zhou, Yingjia Luo, Yinmin Zhong, Yishi Piao, Yisong Wang, Yixiang Zhang, Yixiao Chen, Yixuan Tan, Yixuan Wei, Yiyang Ma, Yiyuan Liu, Yonglun Yang, Yongqiang Guo, Yongtong Wu, Yu Wu, YuKun Li, Yuan Cheng, Yuan Ou, Yuanfan Xu, Yuanhao Li, Yuduan Wang, Yuehan Yang, Yuer Xu, Yuhan Wu, Yuhao Meng, Yuheng Zou, Yukun Zha, Yunfan Xiong, Yupeng Chen, Yuping Lin, Yuqian Cao, Yuqian Wang, Yushun Zhang, Yuting Yan, Yutong Lin, Yuxian Gu, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuxuan Zhou, Yuyang Zhou, Yuzhen Huang, ZF Wu, Zehao Wang, Zehua Zhao, Zehui Ren, Zekai Zhang, Zhangli Sha, Zhe Fu, Zhe Ju, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zheren Gao, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhixian Huang, Zhixuan Chen, Zhiyu Wu, Zhizhou Ren, Zhongyu Wu, Zhuoshu Li, Zhuping Zhang, Zian Xu, Zihao Wang, Zihua Qu, Zihui Gu, Zijia Zhu, Zilin Li, Zipeng Zhang, Ziwei Xie, Ziyi Gao, Ziyi Wan, Zizheng Pan, and Zongqing Yao. DeepSeek-V4: Towards highly efficient million-token context intelligence, 2026. URL https://arxiv.org/abs/2606.19348.

Yuqian Fu, Haohuan Huang, Kaiwen Jiang, Jiacai Liu, Zhuo Jiang, Yuanheng Zhu, and Dongbin Zhao. Revisiting on-policy distillation: Empirical failure modes and simple fixes. *arXiv preprint arXiv:2603.25562*, 2026.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2306.08543.

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, Ashima Suvarna, Benjamin Feuer, Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, Wanjia Zhao, John Yang, Shreyas Pimpalgaonkar, Kartik Sharma, Charlie Cheng-Jie Ji, Yichuan Deng, Sarah Pratt, Vivek Ramanujan, Jon Saad-Falcon, Jeffrey Li, Achal Dave, Alon Albalak, Kushal Arora, Blake Wulfe, Chinmay Hegde, Greg Durrett, Sewoong Oh, Mohit Bansal, Saadia Gabriel, Aditya Grover, Kai-Wei Chang, Vaishaal Shankar, Aaron Gokaslan, Mike A. Merrill, Tatsunori Hashimoto, Yejin Choi, Jenia Jitsev, Reinhard Heckel, Maheswaran Sathiamoorthy, Alexandros G. Dimakis, and Ludwig Schmidt. Openthoughts: Data recipes for reasoning models, 2025. URL https://arxiv.org/abs/2506.04178.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. DeepSeek-R1: Incentivizing reasoning capability in

LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. doi: 10.48550/arXiv.2501.12948. URL https://arxiv.org/abs/2501.12948.

Bingxiang He, Zekai Qu, Zeyuan Liu, Yinghao Chen, Yuxin Zuo, Cheng Qian, Kaiyan Zhang, Weize Chen, Chaojun Xiao, Ganqu Cui, Ning Ding, and Zhiyuan Liu. JustRL: Scaling a 1.5b LLM with a simple RL recipe, 2025. URL https://arxiv.org/abs/2512.16649.

Haonan He, Haodi Lei, Yun Luo, Haoran Zhang, Shunkai Zhang, Yizhuo Li, Shengji Tang, Zhilin Wang, Runzhe Zhan, Lei Bai, Ganqu Cui, Fangchen Yu, Yafu Li, Peng Ye, Ning Ding, and Yu Cheng. SimpleOPD: Simple tokenizer-agnostic on-policy distillation for long-context reasoning, 2026a. URL https://arxiv.org/abs/2608.14277.

Yinghui He, Simran Kaur, Adithya Bhaskar, Yongjin Yang, Jiarui Liu, Narutatsu Ri, Liam Fowl, Abhishek Panigrahi, Danqi Chen, and Sanjeev Arora. Self-distillation zero: Self-revision turns binary rewards into dense supervision, 2026b. URL https://arxiv.org/abs/2604.12002.

Byeongho Heo, Jaehui Hwang, Sangdoo Yun, and Dongyoon Han. On-policy delta distillation, 2026. URL https://arxiv.org/abs/2607.15161.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. doi: 10.48550/arXiv.1503.02531. URL https://arxiv.org/abs/1503.02531.

Namgyu Ho, Laura Schmid, and Se-Young Yun. Large language models are reasoning teachers. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 14852–14882, 2023.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alex Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In *Findings of the association for computational linguistics: ACL 2023*, pp. 8003–8017, 2023.

Li Jiang, Haoran Xu, Yichuan Ding, and Amy Zhang. Trajectory-refined distillation, 2026. URL https://arxiv.org/abs/2606.08432.

Yoon Kim and Alexander M Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pp. 1317–1327, 2016.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, M. C., Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y. Charles, H. S. Che, Guanduo Chen, Guangyu Chen, Guanzheng Chen, Huarong Chen, Jia Chen, Jianlong Chen, Jun Chen, Kexin Chen, Peng Chen, Ruijue Chen, Wentao Chen, Xin Chen, Yang Chen, Yanru Chen, Yifei Chen, Yingjiang Chen, Yuankun Chen, Yujie Chen, Yutian Chen, Zhirong Chen, Dazhi Cheng, Yean Cheng, Jialei Cui, Jingbing Cui, Anqi Dai, Jiaqi Deng, Hao Ding, Rui Ding, Shaofeng Ding, Mengfan Dong, Mengnan Dong, Yuhao Dong, Yuxin Dong, Angang Du, Chenzhuang Du, Dikang Du, Jusen Du, Yulun Du, Yu Fan, Jing Feng, Qiulin Feng, Yichen Feng, Kelin Fu, Qiang Fu, Fuxuan Gao, Hongcheng Gao, Jingyue Gao, Tong Gao, Weijia Gao, Shangyi Geng, Jie Gong, Linhu Gong, Shengao Gong, Xiaochen Gong, Qizheng Gu, Yicheng Gu, Shuhao Guan, Haiqing Guo, Shiqi Guo, Xiang Guo, Zhengyan Guo, Beixi Hao, Wenxin Hao, Xiaoru Hao, Dailan He, Haotian He, Lehan He, Qi He, Weiran He, Xinran He, Xinyi He, Yibo He, Yunjia He, Chao Hong, Tiange Hong, Hao Hu, Jiaxi Hu, Ruikun Hu, Weiming Hu, Yangyang Hu, Zhenxing Hu, Liang Hua, Jinbin Huang, Ke Huang, Ruiyuan Huang, Siying Huang, Weixiao Huang, Yan Huang, Zhengjie Huang, Zhiqi Huang, Yulong Hui, Chaobo Jia, Yutong Jiang, Zhejun Jiang, Zuoyou Jiang, Wenyi Jin, Xinyi Jin, Yu Jing, Huanjun Kong, Guokun Lai, Aidi Li, Cheng Li, Chengyuan Li, Cong Li, Fang Li, Guanyu Li, Haoyang Li, Jia Li, Junxiong Li, Lei Li, Letian Li, Lincan Li, Weihong Li, Wentao Li, Xintong Li, Yang Li, Yishen Li, Yiwei Li, Yuxiao Li, Zhaowei Li, Zhaoxi Li, Zheming Li, Zhengxiao Li, Zhiyuan Li, Jiawei Lin, Xiaohan Lin, Yibo Lin, Zichao Lin, Ziyan Lin, Bill Liu, Boxiao Liu, Chuan Liu, Liang Liu, Shaowei Liu, Shudong Liu, Shuran Liu, Tianwei Liu, Weizhou Liu, Yangyang Liu, Yanming Liu, Yibo Liu, Yipeng Liu, Zhengying Liu, Zhiheng Liu, Enzhe Lu, Haoyu Lu, Linqiang Lu, Tingzhan Lu, Zhiyuan Lu, Aotian Luo, G. Luo, Junyu Luo, Yifan Luo, B. Lyu, Wenzhou Lyu, Shaoguang Mao, Yuan Mei, Xin Men, Minqing Ni, Yixuan Niu, Siyuan Pan, Shujun

Peng, Zhangyang Qi, Ruoyu Qin, ZeChao Qin, Zeyu Qin, Haiquan Qiu, Jianxin Qiu, Jiezhong Qiu, Bowen Qu, Yuhao Qu, Zeyu Shang, Youbo Shao, Han Shen, Jincheng Shi, Juanfeng Shi, Lidong Shi, Shengyuan Shi, Wingchun Siu, Pengwei Song, Xiaoxi Song, Jianlin Su, Yunfeng Su, Zhaochen Su, Lin Sui, Jingsong Sun, Junyao Sun, Shaoning Sun, Shuzhe Sun, Tongyu Sun, Yujun Sun, Yunpeng Tai, Chuning Tang, Heyi Tang, Sirui Tang, Zecheng Tang, Chaoran Tian, Rongpeng Tian, Yu Tian, Wei Tu, Chensi Wang, Chuang Wang, Chunjie Wang, Dinglu Wang, Feng Wang, Hailong Wang, Haiming Wang, Hao Wang, Hao Wang, Huaqing Wang, Hui Wang, Jiayi Wang, Jinglong Wang, Jinhong Wang, Jiuzheng Wang, Linian Wang, Shaobo Wang, Shenzhi Wang, Shuyi Wang, Si Wang, Siyuan Wang, Tianfu Wang, Wenjue Wang, Xingran Wang, Xinmei Wang, Xinyuan Wang, Xusheng Wang, Yalin Wang, Yangkun Wang, Yao Wang, Yaoyu Wang, Yejie Wang, Yiqin Wang, Yucheng Wang, Yuzhi Wang, Zhaoji Wang, Zhaowei Wang, Zhengtao Wang, Zhenhao Wang, Zhongsheng Wang, Zifan Wang, Chu Wei, Ming Wei, Shouxin Wei, Zichen Wen, Fan Wu, Haoning Wu, Rucong Wu, Wenhao Wu, Xiaoxue Wu, Yingcong Wu, Yongqi Wu, Yuxin Wu, Zijian Wu, Xinglang Xian, Chenxuan Xiang, Yuye Xiang, Bocheng Xiao, Chenjun Xiao, Xin Xiao, Jin Xie, Xiaotong Xie, Yifeng Xie, Zhe Xie, Bowei Xing, Yiming Xiong, Baosheng Xu, Boyu Xu, Jiale Xu, Jianfan Xu, Jing Xu, Jinjing Xu, L. H. Xu, Qingtao Xu, Shuyao Xu, Suting Xu, Tiantian Xu, Tianxiang Xu, Weixin Xu, Xinran Xu, Yangchuan Xu, Ye Xu, Yueni Xu, Ziyao Xu, Haonan Xue, Junjie Yan, Yaoyao Yan, Fan Yang, Guangyao Yang, Hao Yang, Junwei Yang, Ruoyu Yang, Wenjie Yang, Xiaofei Yang, Xinyu Yang, Yi Yang, Yiling Yang, Ying Yang, Yuchen Yang, Zhen Yang, Zhilin Yang, Zian Yang, Zuhao Yang, Haotian Yao, Dan Ye, Haoran Ye, Wenjie Ye, Zhanbo Ye, Bohong Yin, Haoxiang Yin, Xietong Yin, Chengzhen Yu, Haozhen Yu, Longhui Yu, Shengnan Yu, Shuying Yu, Tianxiang Yu, Enming Yuan, Mengjie Yuan, Tongtian Yue, Wei Yue, Yang Yue, Dunyuan Zha, Haobing Zhan, B. H. Zhang, Dehao Zhang, Fei Zhang, Hao Zhang, Haoyuan Zhang, Huanyu Zhang, Jiapei Zhang, Jiaxuan Zhang, Jin Zhang, Kaiyi Zhang, Miaozhen Zhang, Puqi Zhang, Qinglei Zhang, Rong Zhang, Rui Zhang, Shaoshuai Zhang, Shiyi Zhang, Xiaobin Zhang, Xiaoyun Zhang, Y. Zhang, Yangkun Zhang, Ye Zhang, Yichi Zhang, Yikun Zhang, Yizhi Zhang, Yongting Zhang, Yu Zhang, Yutao Zhang, Yutong Zhang, Zheng Zhang, Zijing Zhang, Bin Zhao, Chenguang Zhao, Feifan Zhao, Jinglun Zhao, Jinxiang Zhao, Shuai Zhao, Wenshuo Zhao, Xiangyu Zhao, Xuanle Zhao, Yikai Zhao, Zijia Zhao, Haozhi Zheng, Huabin Zheng, Ruihan Zheng, Shaojie Zheng, Tengyang Zheng, Haofeng Zhong, Lei Zhong, Longguang Zhong, M. Zhou, Qiankang Zhou, Runjie Zhou, Ruozhang Zhou, Xinyu Zhou, Yiqiao Zhou, Zaida Zhou, Jinguo Zhu, Liya Zhu, Xinhao Zhu, Yangjunfeng Zhu, Yuxuan Zhu, Zhen Zhu, Chen Zhuang, Weiyu Zhuang, and Xinxing Zu. Kimi k3: Open frontier intelligence, 2026. URL https://arxiv.org/abs/2607.24653.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huanang Gao, Wenkai Yang, Zhiyuan Liu, and Ning Ding. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe, 2026. URL https://arxiv.org/abs/2604.13016.

Tianqi Liu, Wei Xiong, Jie Ren, Lichang Chen, Junru Wu, Rishabh Joshi, Yang Gao, Jiaming Shen, Zhen Qin, Tianhe Yu, Daniel Sohn, Anastasiia Makarova, Jeremiah Liu, Yuan Liu, Bilal Piot, Abe Ittycheriah, Aviral Kumar, and Mohammad Saleh. RRM: Robust reward model training mitigates reward hacking. In *Proceedings of the 13th International Conference on Learning Representations*, 2025a. URL https://arxiv.org/abs/2409.13156.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding R1-Zero-like training: A critical perspective, 2025b. URL https://arxiv.org/abs/2503.20783.

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.

Feng Luo, Yu-Neng Chuang, Guanchu Wang, Zicheng Xu, Xiaotian Han, Tianyi Zhang, and Vladimir Braverman. Demystifying OPD: Length inflation and stabilization strategies for large language models, 2026. URL https://arxiv.org/abs/2604.08527.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. DeepScaleR: Surpassing O1-Preview with a 1.5B model by scaling RL. https://pretty-radio-b75

.notion.site/DeepScaleR-Surpassing-O1-Preview-with-a-1-5B-Model-by-Scaling-RL-19681902c1468005bed8ca303013a4e2, 2025. Notion Blog.

Wenhan Ma, Jianyu Wei, Liang Zhao, Hailin Zhang, Bangjun Xiao, Lei Li, Qibin Yang, Bofei Gao, Yudong Wang, Rang Li, Jinhao Dong, Zhifang Sui, and Fuli Luo. MOPD: Multi-teacher on-policy distillation for capability integration in LLM post-training, 2026. URL https://arxiv.org/abs/2606.30406.

Arindam Mitra, Luciano Del Corro, Shweti Mahajan, Andres Codas, Clarisse Simoes, Sahaj Agarwal, Xuxi Chen, Anastasia Razdaibiedina, Erik Jones, Kriti Aggarwal, Hamid Palangi, Guoqing Zheng, Corby Rosset, Hamed Khanpour, and Ahmed Awadallah. Orca 2: Teaching small language models how to reason, 2023. URL https://arxiv.org/abs/2311.11045.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and Ahmed Awadallah. Orca: Progressive learning from complex explanation traces of GPT-4, 2023. URL https://arxiv.org/abs/2306.02707.

Hao Wang, Kun Yuan, Wenlin Zhong, Minglei Zhang, Han Xiao, Ming Sun, and Honggang Qi. Cross-tokenizer on-policy distillation via byte-prefix marginalization, 2026a. URL https://arxiv.org/abs/2607.22334.

Rui Wang, Hongru Wang, Yi Chen, Boyang Xue, Tianqing Fang, Wenhao Yu, and Kam-Fai Wong. Demystifying on-policy distillation: Roles, pathologies, and regulations, 2026b. URL https://arxiv.org/abs/2607.13399.

Bangjun Xiao, Bingquan Xia, Bo Yang, Bofei Gao, Bowen Shen, Chen Zhang, Chenhong He, Chiheng Lou, Fuli Luo, Gang Wang, et al. MiMo-V2-Flash technical report. *arXiv preprint arXiv:2601.02780*, 2026.

Xingrun Xing, Haoqing Wang, Boyan Gao, Ziheng Li, and Yehui Tang. Trust region on-policy distillation, 2026. URL https://arxiv.org/abs/2606.01249.

Jianhao Elliott Yan, Jin Xu, Chiyu Song, Chenming Wu, Yafu Li, and Yue Zhang. Understanding in-context learning from repetitions. In *International Conference on Learning Representations*, volume 2024, pp. 49968–49989, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Shenzhi Yang, Guangcheng Zhu, Bowen Song, Haobo Wang, Mingxuan Xia, Xing Zheng, Yingfan Ma, Zhongqi Chen, Weiqiang Wang, Junbo Zhao, and Gang Chen. OPRD: On-policy representation distillation, 2026a. URL https://arxiv.org/abs/2606.06021.

Wenkai Yang, Weijie Liu, Ruobing Xie, Kai Yang, Saiyong Yang, and Yankai Lin. Learning beyond teacher: Generalized on-policy distillation with reward extrapolation, 2026b. URL https://arxiv.org/abs/2602.12125.

Tianzhu Ye, Li Dong, Xun Wu, Shaohan Huang, and Furu Wei. On-policy context distillation for language models, 2026. URL https://arxiv.org/abs/2602.12275.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan 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, Mu Qiao, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in LLMs beyond the base model? In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 57654–57689. Curran Associates, Inc., 2025. doi: 10.52202/085713-1933. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/537d5aa768c2d534016a4d06f87bc8fb-Paper-Conference.pdf.

Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, Da Yin, Chendi Ge, Chenghua Huang, Chengxing Xie, et al. GLM-5: From vibe coding to agentic engineering. *arXiv preprint arXiv:2602.15763*, 2026.

Chuheng Zhang, Wei Shen, Li Zhao, Xuyun Zhang, Xiaolong Xu, Wanchun Dou, and Jiang Bian. Policy filtration for RLHF to mitigate noise in reward models. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025. URL https://arxiv.org/abs/2409.06957.

Hongbo Zhang, Yue Yang, Jianhao Yan, Guangsheng Bao, Yue Zhang, and Yue Zhang. Detecting RLVR training data via structural convergence of reasoning, 2026. URL https://arxiv.org/abs/2602.11792.

Rosie Zhao, Alexandru Meterez, Sham Kakade, Cengiz Pehlevan, Samy Jelassi, and Eran Malach. Echo chamber: RL post-training amplifies behaviors learned in pretraining, 2025. URL https://arxiv.org/abs/2504.07912.

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, 2026. URL https://arxiv.org/abs/2601.18734.

Siqi Zhu, Xuyan Ye, Hongyu Lu, Weiye Shi, and Ge Liu. The many faces of on-policy distillation: Pitfalls, mechanisms, and fixes, 2026. URL https://arxiv.org/abs/2605.11182.

Zilin Zhu, Chengxing Xie, Xin Lv, and slime Contributors. slime: An LLM post-training framework for RL scaling. https://github.com/THUDM/slime, 2025.

A Experimental Details

A.1 Training Settings

We conduct three main experiments to study these phenomena. The JustRL and DeepScaleR settings share the same student initialization and training data, while the Qwen3-4B setting uses a base Qwen3 student and OpenThoughts3 prompts. The differences between the settings are summarized in Table 4.

Table 4: **Training settings used in the main experiments.** The teacher is fixed throughout OPD training.

Setting	Teacher	Student initialization	Training data	Length limit	LR
JustRL	JustRL-1.5B	DeepSeek-R1-Distill-Qwen-1.5B	DeepMath-103K, level ≥ 6	16,384	10^{-5}
DeepScaleR	DeepScaleR-1.5B-Preview	DeepSeek-R1-Distill-Qwen-1.5B	DeepMath-103K, level ≥ 6	16,384	10^{-5}
Qwen3-4B	Qwen3-4B	Qwen3-1.7B-Base	OpenThoughts3 OPD 30K	8,192	10^{-5}

Each setting runs for 100 rollout iterations. At each iteration, the student samples 8 responses for each of 32 prompts at temperature 1.0 and top-p 1.0, producing 256 responses and two optimizer updates with global batch size 128. We use Adam with $\beta_1 = 0.9$, $\beta_2 = 0.98$, weight decay 0.1, no learning-rate warmup, PPO clipping $\epsilon = 0.2$, and an approximate reference-KL coefficient of 0.001 toward the fixed initialization S_{INIT}.

The loss mask covers response tokens only. Each sample is first averaged over its unmasked response tokens, and the resulting values are then averaged over the global batch, so that long and short responses contribute equally. All runs use eight NVIDIA H100 GPUs. Checkpoints are named by rollout count: the initialization S_{INIT} together with iterations 20, 40, 60, 80, and 100, where iteration 100 gives S_{OPD}.

A.2 DeepScaleR Teacher-Temperature Pilot

We ran two small temperature studies before fixing the DeepScaleR setting. First, Table 5 evaluates the frozen DeepScaleR-1.5B-Preview teacher itself. For every temperature, we draw one response per problem at top-p 1.0 with an 8,192-token limit and score it with the verl verifier. The macro average weights AMC23, AIME24, and AIME25 equally despite their different problem counts.

Table 5: **Standalone teacher accuracy depends on the generation-sampling temperature.** Accuracy is in percent; Macro gives equal weight to the three benchmarks, and bold marks each column maximum.

Temperature	AMC23	AIME24	AIME25	Macro
0.1	66.27	23.33	**30.00**	39.87
0.5	**67.47**	30.00	26.67	41.38
0.75	65.06	**36.67**	23.33	**41.69**
1	56.63	20.00	10.00	28.88
1.5	0.00	0.00	0.00	0.00

The training pilot in Figure 9 instead varies the temperature applied to the teacher logits when scoring student rollouts. Each temperature corresponds to one training run. At each displayed checkpoint, we evaluate the student with four responses per problem sampled at temperature 1.0, top-p 1.0, and a 16,384-token limit; we average within each problem and then give the three benchmarks equal weight. The $T = 1.5$ run has observations only through iteration 20.

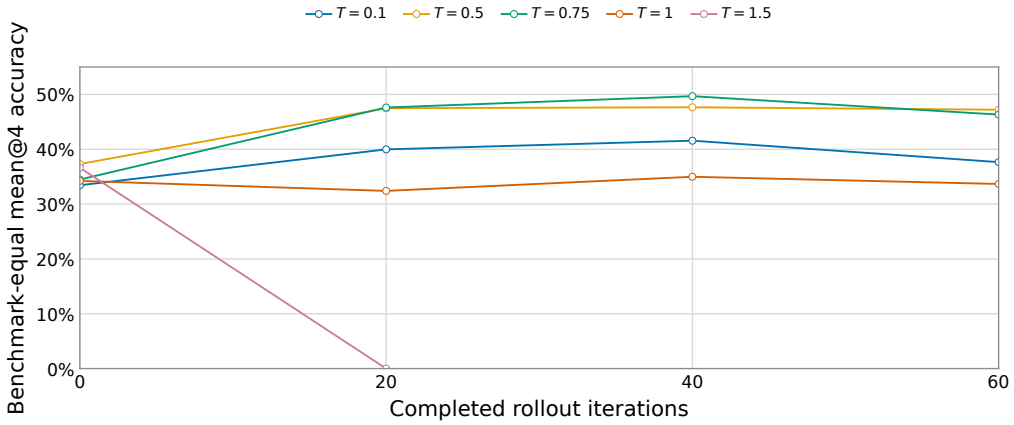


Figure 9: **Intermediate teacher-scoring temperatures give the strongest single-run student trajectories in the pilot.** Benchmark-equal student mean@4 accuracy is plotted against completed rollout iterations for the five teacher-scoring temperatures.

These pilots support only a working choice, not an optimum. In the standalone teacher study, $T = 0.75$ has macro accuracy 41.69%, just 0.31 percentage points above $T = 0.5$; weighting all 143 problems equally reverses the ordering (50.35% versus 51.05%). The teacher study uses one sample per problem, and every training curve is a single run with unequal coverage at $T = 1.5$. Together with the strongest observed training point at $T = 0.75$, these results motivated adopting 0.75 as the working temperature, but do not establish that it is optimal.

A.3 Evaluation Protocol

We use two evaluation protocols, and they differ only in how many responses are drawn per problem. Both sample at top-p 1.0, use the same generation length limit as training, and serialize prompts with the template described in Appendix A.4. The sampling temperature follows the model being evaluated rather than the protocol. In the JustRL setting and the DeepScaleR setting, S_{INIT} and S_{OPD} are both sampled at $T = 1.0$, matching the rollout temperature used in training.

Routine evaluation. Endpoint and per-checkpoint accuracies are reported as mean@4: we draw four responses per problem, and calculate average accuracy among all the responses. This protocol is used for the accuracy columns in Section 5.2 and for the accuracy traces over training.

Large-pool evaluation. The coverage and pass@k analyses in Section 3 require a much larger budget. For each model and each of the 90 AIME24–26 problems we draw $n = 256$ responses. This pool supports pass@k for $k \in \{1, 2, 4, 8, 16, 32, 64, 128, 256\}$ and the per-problem success rate $\hat{p}_m(q)$, defined as the fraction of verifier-correct responses among the 256 samples.

Conditional extension for the coverage audit. For the audit in Appendix B we extend sampling only where it can change the conclusion. For every problem covered by S_{OPD} but not by S_{INIT} at $n = 256$, we draw a further 768 responses from S_{INIT}, giving a conditional pool of 1,024 responses for that problem. The prompt, temperature, top-p, and length limit are unchanged; only the number of samples increases.

The audit protocol is deliberately one-sided. Both halves of the audit act in a single direction, and we state this explicitly rather than leave it to be inferred. The additional sampling is spent only on problems that S_{OPD} covers and S_{INIT} does not; the reverse case—covered by S_{INIT} but not by S_{OPD}—never receives a larger S_{OPD} pool. Manual review is applied to the same candidate list, where it reinstates verifier misses among S_{INIT} responses and rejects invalid S_{OPD} derivations. Relative to the unaudited 256-response pools, the procedure can therefore only raise S_{INIT} coverage and only lower S_{OPD} coverage, which is the direction that favors the conclusion of Section 3.

We chose this design because a symmetric audit at equal budget would spend most of it on problems whose status is not in question, and because the claim under test is one-sided: a single surviving S_{OPD}-only problem would refute it, whereas additional S_{INIT}-only problems would not change it. The cost is that the audited partition in Table 6 is not a symmetric measurement of the two models. It should be read as an upper bound on S_{INIT} coverage and a lower bound on S_{OPD} coverage under this protocol, so the empty S_{OPD}-only column reports that no candidate survived a search directed against it, not that the two supports were compared on equal terms. The audited S_{INIT} and S_{OPD} totals are therefore not on the same footing, and their difference overstates the true gap.

A second consequence is that the candidate list differs between the two teacher settings, so the two rows of Table 6 audit the same S_{INIT} checkpoint under different budgets. We resolve this by reporting a single S_{INIT} set: the union of the problems it is found to solve under either audit. This is sound because S_{INIT} is one checkpoint drawing on one shared pool of 256 responses per problem, so a solution recovered in one audit is evidence about the checkpoint and not about the teacher it was audited alongside. No counterpart exists on the S_{OPD} side, where the two rows are different models. Pooling in this way gives S_{INIT} the combined budget of both audits and so widens the asymmetry above; it cannot create an S_{OPD}-only problem, since enlarging S_{INIT} can only remove entries from that column.

Estimating PASS@k. Reporting PASS@k from a single group of k responses is high variance. For a problem with c correct responses among n, we instead use the standard unbiased estimator of the probability that a group of k responses drawn without replacement from the pool contains at least one correct response (Chen et al., 2021),

$$\text{PASS@}k = \mathbb{E}_q \left[1 - \frac{\binom{n-c_q}{k}}{\binom{n}{k}} \right], \tag{5}$$

evaluated at $n = 256$ and averaged over problems with equal weight. We compute the ratio as $\prod_{i=0}^{k-1}(n - c_q - i)/(n - i)$ to avoid large binomial coefficients. At $k = n$ the estimator reduces to an indicator of whether the pool contains any correct response, so PASS@256 equals the observed coverage of that pool.

Correctness. Correctness is always recomputed from the response text; we do not reuse the reward labels stored during rollout, which are produced by a different scorer and under a different generation setting. All three benchmarks have non-negative integer answers, and the verifier extracts a predicted integer with the first rule that applies: the content of the last `\boxed{...}` holding an integer; otherwise the last integer introduced by an explicit answer cue such as `final answer is`; otherwise the response as a whole, if it consists of a single integer. A response with no extractable integer counts as incorrect. Prediction and reference are compared after integer normalization, so the verifier accepts no symbolic equivalence beyond leading-zero and whitespace differences.

The second and third rules make the verifier permissive: a response can be scored correct without ever emitting a boxed answer. This matters for the coverage audit, where accepting an accidental hit from S_{INIT} would weaken the claim it supports. We therefore inspect every S_{INIT} recovery found in the extended pools by hand rather than trusting the verifier alone, and report the outcome per problem in Appendix B.

A.4 PROMPT TEMPLATES

Training and evaluation examples contain one user message and no system message. Each training example already includes the instruction `Please reason step by step, and put your final answer within \boxed {}.` after the problem statement, separated by a newline in DeepMath and a space in OpenThoughts3. We include the same instruction for mathematics evaluation and serialize the message with the student's chat template. Routine evaluation uses the same template as training. Below, {QUESTION} denotes the problem statement without this instruction.

JustRL and DeepScaleR settings. Both use the DeepSeek-R1-Distill-Qwen-1.5B student template:

JustRL / DeepScaleR Prompt | DeepSeek-R1-Distill-Qwen-1.5B

```
<|begin_of_sentence|><|User|>{QUESTION}
Please reason step by step, and put your final answer within \boxed{}.<|Assistant|><think>
```

The assistant prefix ends with `<think>` followed by a newline, leaving the reasoning block open for generation.

Qwen3-4B setting. We use the Qwen3-1.7B student template with thinking mode disabled. Specifically, we prefill the assistant turn with an empty reasoning block, `<think>` followed by two newlines, `</think>`, and two more newlines. Generation therefore starts after the closed block. The serialized prompt is:

Qwen3-4B Setting Prompt | Qwen3-1.7B | Thinking Disabled

```
<|im_start|>user
{QUESTION} Please reason step by step, and put your final answer within \boxed{}.<|im_end|>
<|im_start|>assistant
<think>

</think>
```

Disabling thinking mode changes this assistant prefix; the user instruction to reason step by step remains in the prompt.

Teacher scoring. After the student generates a response, we feed the full token sequence—the serialized prompt and the student-generated response—into the teacher in a single forward pass to obtain the next-token distributions at the response positions for distillation. The teacher generates no new tokens (`max_new_tokens=0`). We use the student's original token sequence directly, without reapplying a chat template or adding any instructions.

B Observed Solvable Sets

We first construct candidate solvable sets from problems with at least one verifier-correct response in the original 256-response pools. We then refine these sets through targeted sampling and manual review: invalid derivations are excluded, and valid solutions missed by the automatic verifier are reinstated. Table 6 reports the resulting audited partition of all 90 AIME24–26 problems under each teacher. The four columns form a disjoint partition: covered by both S_{INIT} and S_{OPD}, covered only by one of them, or covered by neither. Problem numbers are listed explicitly within each dataset.

The audit combines additional sampling with review of the existing outputs. We take every problem that is covered by S_{OPD} but not by S_{INIT} among 256 responses and extend the S_{INIT} pool to 1,024 responses. This recovers seven cases: JustRL AIME25-025 and AIME26-010, and DeepScaleR AIME25-014, AIME25-023, AIME25-025, AIME26-010, and AIME26-022. Manual review also checks the validity of S_{OPD} derivations among the candidate problems, including cases already recovered by additional sampling. The JustRL AIME25-020 and DeepScaleR AIME25-014 and AIME25-025 derivations are invalid despite reaching the correct final answer. The latter two problems are therefore assigned to the S_{INIT}-only set.

For JustRL AIME26-022 and AIME26-027, the automatic verifier records no S_{INIT} hit in the 1,024-response pools. Manual review identifies valid S_{INIT} solutions in the existing outputs and corrects these verifier misses; it does not use a separate batch of generations. These cases are assigned to the shared set. Table 7 distinguishes recoveries by additional sampling from corrections to automatic verification.

Table 6: **Audited observed-solvable-set partitions.** Each cell gives the number of problems followed by their three-digit suffixes.

Teacher	Data	Both	S_{INIT} only	S_{OPD} only	Neither
JustRL	AIME24	**24**: 001–007, 009, 010, 013–022, 025–029	**1**: 023	**0**: –	**5**: 008, 011, 012, 024, 030
JustRL	AIME25	**18**: 001–006, 008, 009, 012, 016, 017, 019, 021, 022, 024, 025, 027, 029	**6**: 010, 013, 014, 023, 028, 030	**0**: –	**6**: 007, 011, 015, 018, 020, 026
JustRL	AIME26	**23**: 001–008, 010, 012–014, 016, 018–027	**1**: 011	**0**: –	**6**: 009, 015, 017, 028–030
DeepScaleR	AIME24	**23**: 001–007, 009, 010, 013–022, 025–028	**2**: 023, 029	**0**: –	**5**: 008, 011, 012, 024, 030
DeepScaleR	AIME25	**18**: 001–006, 008, 009, 012, 016, 017, 019, 021–024, 027, 029	**6**: 010, 013, 014, 025, 028, 030	**0**: –	**6**: 007, 011, 015, 018, 020, 026
DeepScaleR	AIME26	**22**: 001–008, 010, 012–014, 016, 018–026	**2**: 011, 027	**0**: –	**6**: 009, 015, 017, 028–030

Table 7: **Resolution of the raw S_{OPD}-only cases.** The third column records how a correct S_{INIT} response was obtained. "Extended sampling" means the verifier accepted a response drawn while extending the pool to 1,024. "Manual review" means a correct response was already present and had been missed by the verifier, so no additional generation was needed. "None" means neither route produced one.

Teacher	Problem	S_{INIT} solution found by	S_{OPD} derivation	Resolution
JustRL	AIME25-020	None	Invalid	Removed as an accidental verifier hit
JustRL	AIME25-025	Extended sampling	Valid	Shared
JustRL	AIME26-010	Extended sampling	Valid	Shared
JustRL	AIME26-022	Manual review	Valid	Shared
JustRL	AIME26-027	Manual review	Valid	Shared
DeepScaleR	AIME25-014	Extended sampling	Invalid	S_{INIT} only
DeepScaleR	AIME25-023	Extended sampling	Valid	Shared
DeepScaleR	AIME25-025	Extended sampling	Invalid	S_{INIT} only
DeepScaleR	AIME26-010	Extended sampling	Not needed	Shared
DeepScaleR	AIME26-022	Extended sampling	Valid (two reviews)	Shared

These sets describe the responses observed under the stated sampling and manual-audit protocol. They do not establish equality of model support or zero probability for any response under S_{INIT}.

C Min-10NN Distance

Min-10NN measures how tightly responses to the same problem concentrate in surface form. For two decoded responses a and b, we use character-level normalized Levenshtein distance

$$d(a,b) = \begin{cases} 0, & |a| = |b| = 0, \\ \mathrm{Lev}(a,b)/\max\{|a|,|b|\}, & \text{otherwise.} \end{cases} \tag{6}$$

Let $R = \{r_i\}_{i=1}^{16}$ be 16 responses to one problem. For each response, we compute its distance to the nearest other response,

$$\delta_i = \min_{j \neq i} d(r_i, r_j). \tag{7}$$

After sorting these values as $\delta_{(1)} \leq \cdots \leq \delta_{(16)}$, Min-10NN is

$$\text{Min-10NN}(R) = \frac{1}{10} \sum_{i=1}^{10} \delta_{(i)}. \tag{8}$$

A smaller value means that the ten most tightly clustered responses are more similar. When 32 responses are available, we sample 16 without replacement, average Min-10NN over 100 repetitions, and then average across problems. The metric captures character-level concentration only; it does not measure semantic equivalence or response quality.

D Response Case Study

Figure 10 gives the examples of the excerpts condensed in Figure 1b. All four responses come from the Qwen3-4B setting before any OPD update: (a) is a free generation of the teacher, and (b)–(d) are sampled from S_{INIT}, with (c) and (d) drawn from the responses the teacher ranks as most preferred under the decile partition of Section 5.1.

(a) Teacher: coherent solution
correct; 842 tokens
"Number of **grand prize** outcomes: 1. … Number of **prize** outcomes ($\geq$ 2 matches): 115.
… $P(\text{Grand Prize} \mid \text{Prize}) = 1/115$. Thus, $m + n = 1 + 115 = 116$."

(b) S_{INIT}: coherent solution
correct; 771 tokens
"Jen has exactly 2 of the chosen 4 numbers:
$C(4,2)C(6,2) = 90$. … Exactly 3: 24; all 4: 1. So, the total number of ways to win a prize is $90 + 24 + 1 = 115$.
… $P(\text{grand prize} \mid \text{prize}) = 1/115$. … $\boxed{116}$."

(c) S_{INIT}, teacher-preferred: repeated reconsideration
incorrect; 7,311 tokens
"$17{,}499{,}840/661{,}680 = 26.5$. **Wait, this is incorrect.** The calculation is not correct. Instead, we should compute the number of valid sequences. Let's go back. … But this is not correct. **Wait, for the last word** to be GY, … **Wait, this is not correct.**"

(d) S_{INIT}, teacher-preferred: mechanical repetition
correct; 8,192 tokens; truncated
"**Final Answer:** $\boxed{116}$ … **Answer:** $\boxed{116}$ … **Final Answer:** $\boxed{116}$ … **Answer:** $\boxed{116}$ …"
The same answer line repeats 365 times until the generation length limit.

Figure 10: **Illustrative response excerpts from the Qwen3-4B setting.** The teacher and a typical S_{INIT} response give coherent solutions, whereas the S_{INIT} responses the teacher most prefers repeatedly restart their reasoning or mechanically repeat an answer until the generation length limit. All four excerpts predate any OPD update. Ellipses mark omitted text. These examples illustrate behaviors and do not estimate their prevalence.

E High-Advantage Responses from the Initial Student

We present three responses sampled from Qwen3-1.7B-Base before any OPD update and scored by the Qwen3-4B teacher. All three belong to the highest-advantage group, A10, in the analysis of Section 5.1 and illustrate excessive length and mechanical repetition already present under S_{INIT}. All three are incorrect and contain no parseable final answer.

Reading the complete responses. Each response consists almost entirely of a repeated string. We specify its complete content using exact repetition counts: $+$ denotes concatenation without an added separator, and $\text{repeat}(s, n)$ denotes n consecutive copies of s. Quotation marks delimit strings and are not part of the response; `\n` denotes a newline. Unicode characters are written as `\uXXXX` escapes. No text is omitted from these representations. The original UTF-8 prompts, responses, token IDs, and provenance are retained in the accompanying `data/appendix/sinit_examples/` directory. Prompt mathematics below is typeset for readability. Every prompt ends with: "Please reason step by step, and put your final answer within `\boxed{}`."

E.1 Length-Cap Repetition

Prompt (AIME24, `aime24_001`). Every morning Aya goes for a 9-kilometer-long walk and stops at a coffee shop afterwards. When she walks at a constant speed of s kilometers per hour, the walk takes her 4 hours, including t minutes spent in the coffee shop. When she walks $s + 2$ kilometers per hour, the walk takes her 2 hours and 24 minutes, including t minutes spent in the coffee shop. Suppose Aya walks at $s + \frac{1}{2}$ kilometers per hour. Find the number of minutes the walk takes her, including the t minutes spent in the coffee shop.

Response (sample 65).

```
"\uFFFD\uFFFD\u6001 T 10T 50 10 1 10 10"
+ repeat("T 10", 2043)
```

The response mechanically repeats `T 10` until it reaches the 8,192-token generation cap. The two initial `\uFFFD` characters are present in the stored response.

E.2 Severe Repetition without Reaching the Length Cap

Prompt (AIME25, `aime25_028`). Let the sequence of rationals $x_1, x_2, \ldots$ be defined such that $x_1 = \frac{25}{11}$ and $x_{k+1} = \frac{1}{3}\left(x_k + \frac{1}{x_k} - 1\right)$. x_{2025} can be expressed as $\frac{m}{n}$ for relatively prime positive integers m and n. Find the remainder when $m + n$ is divided by 1000.

Response (sample 110).

```
repeat(".slfwith\n", 2154)+ ".slfwith"
```

The response consists of repeated `.slfwith` lines and ends after 6,465 recorded tokens, showing that severe repetition can occur without reaching the generation cap.

E.3 Severe Repetition at the Length Cap

Prompt (AIME25, `aime25_013`). Alex divides a disk into four quadrants with two perpendicular diameters intersecting at the center of the disk. He draws 25 more line segments through the disk, drawing each segment by selecting two points at random on the perimeter of the disk in different quadrants and connecting those two points. Find the expected number of regions into which these 27 line segments divide the disk.

Response (sample 58).

```
"\u6CD5 newVal89\u6CD5 newVal8\u6CD5 newVal8"
+ repeat("\u6CD5 newVal", 4091)
```

The response repeats the same fragment until it reaches the 8,192-token generation cap, exhibiting both excessive length and severe repetition.

Together, these cases illustrate how excessively long and repetitive responses can receive high relative teacher advantage already under S_{INIT}.

F Training Details for the Candidate-Pool Interventions

Section 5.2 compares three Qwen3-4B teacher settings. The baseline and masked settings initialize the student from Qwen3-1.7B-Base. The warmup setting instead initializes it from the Qwen3-1.7B-SFT checkpoint released by Li et al. (2026) and uses that model as the fixed reference. All three use OpenThoughts3 prompts, temperature 1.0, an 8,192-token generation length limit, and the optimization settings in Appendix A.1.

Loss masking for truncated or failed responses. The baseline trains on all sampled responses, including those that reach the generation length limit. In the masked setting, we set the loss mask to zero for truncated or failed responses; completed responses retain nonzero loss masks and contribute to training. Truncated or failed responses remain in reward and advantage normalization but contribute no policy, reference-KL, or entropy gradient. The loss is first averaged over unmasked response tokens within each sample and then over samples in the global batch.

Initialization from a released SFT checkpoint. The warmup setting directly loads the Qwen3-1.7B-SFT checkpoint released by Li et al. (2026); we do not perform an additional SFT stage ourselves. It changes only the student initialization and fixed reference from Qwen3-1.7B-Base to Qwen3-1.7B-SFT; the teacher, data, optimizer, sampling protocol, and evaluation schedule remain fixed. Its higher initial accuracy is part of the intervention: it tests whether supplying a better initial candidate pool changes what OPD can amplify.