When Do We Need On-Policy Distillation? Distilling on Offline Student Rollouts Is Often Better

Anonymous authors
Paper under double-blind review

Abstract

On-policy distillation (OPD) has become increasingly popular for transferring teacher capabilities to student models. In this work, we ask: Is on-policy sampling always beneficial for distilling arbitrary teacher–student pairs? We show that a simple alternative, Semi-OPD, which distills from offline rollouts generated by the initial student, can often outperform OPD in both accuracy and training efficiency. Across 17 teacher–student pairs ranging from 1.5B to 235B parameters, Semi-OPD outperforms OPD in 14 cases, with up to +13.6% accuracy and 11.4× training speedup. We further find that the choice between OPD and Semi-OPD depends on the alignment between the initial teacher and student, quantified by an output-token overlap ratio: OPD is beneficial only when the two are highly aligned with high overlap ratios. Our further investigation suggests that effective distillation requires on-policyness w.r.t. both the student and the teacher. For misaligned pairs, student rollouts can become increasingly off-policy w.r.t. the teacher as context length grows, weakening the distillation signal. In contrast, Semi-OPD is often more stable, as it distills on shorter contexts while covering full trajectories and exposing the student to more teacher-preferred tokens. Beyond proposing Semi-OPD as an efficient alternative, our work motivates rethinking when to use OPD and studying stronger OPD variants using meaningful teacher–student pairs.

1 Introduction

On-policy distillation (OPD) has emerged as an effective paradigm for transferring the capabilities of a teacher language model (LM) to a student LM (Agarwal et al., 2024; Gu et al., 2024; Lu & Lab, 2025; Xiaomi, 2026; Xu et al., 2026a). It mitigates the training–inference distribution mismatch, i.e., exposure bias, in traditional knowledge distillation by performing token-level distillation on student rollouts obtained via on-policy sampling.

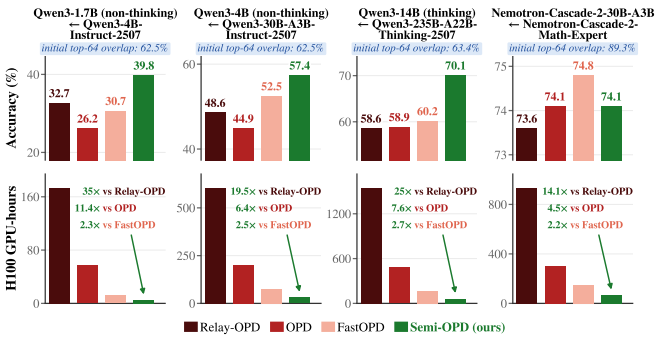


Figure 1: **Accuracy & training cost of different distillation methods.** Our Semi-OPD achieves much improved/comparable accuracy for teacher–student pairs with low/high overlap ratios, respectively, while notably reducing training time by up to 11.4× over OPD. Accuracy is reported as avg@32 over AIME 24/25/26 and HMMT Feb 26.

Despite the promise, recent works reveal several artifacts of OPD: Luo et al. (2026); Yang et al. (2026a); Fu et al. (2026a); Yu et al. (2026) find that on-policy rollouts tend to grow longer and fall into self-correction loops, exhausting the generation budget without emitting an EOS token; Zhang et al. (2026); Ziheng et al. (2026); Liu et al. (2026) find that supervision on later tokens in student rollouts is often unhelpful and that truncating to a short prefix can suffice; and Xu et al. (2026b); Jiang et al. (2026) similarly finds that once the student commits to an incorrect reasoning direction, subsequent

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

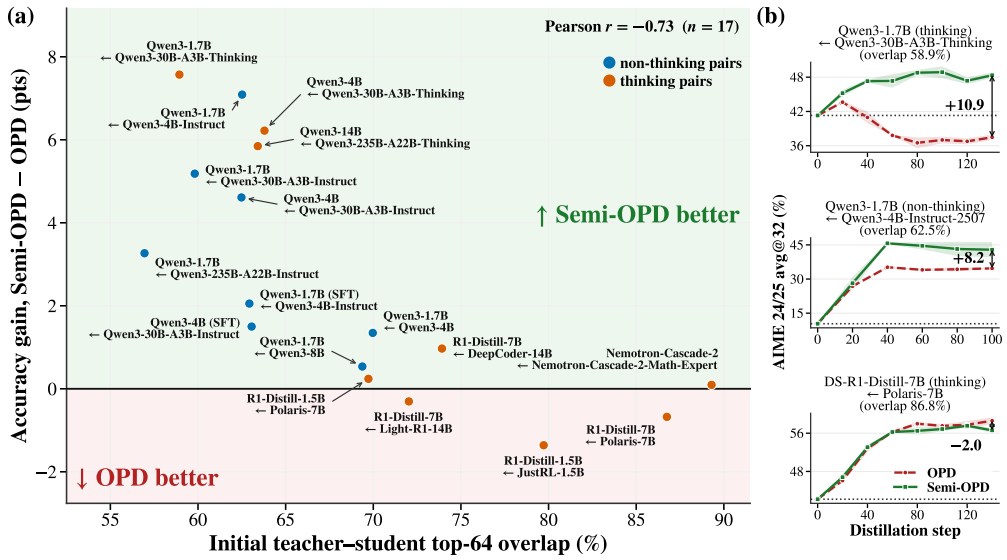


Figure 2: **Semi-OPD outperforms standard OPD when the initial teacher–student overlap is low. (a)** Accuracy gain vs. initial top-64 overlap across 17 pairs ranging from 1.5B to 235B, spanning thinking and non-thinking modes and multiple model families, with a Pearson correlation of $r = -0.73$. Accuracy gain is the area between the Semi-OPD and OPD accuracy curves divided by their shared training horizon, averaged over AIME 24/25/26 and HMMT Feb 26. **(b)** Training curves for three pairs of increasing overlap (AIME 24/25 avg@32, mean of 3 seeds). At low overlap, Semi-OPD outperforms OPD which can even degrade; at high overlap, Semi-OPD can slightly lag behind.

misdirected continuations elicit unreliable supervision and waste compute. These artifacts lead to our central question:

Is on-policy sampling always beneficial for distilling arbitrary teacher–student pairs?

To answer this, we propose a simple and efficient alternative called semi-on-policy distillation (Semi-OPD), where rollouts are sampled once from the *initial* student before any training and reused throughout distillation, thus making both rollout and teacher logit generation offline. Although the source of rollouts is fixed, the objective is identical to OPD's, and the student's log-probabilities, and hence the per-token advantage, are recomputed using the current parameters at every step. We evaluate across 17 teacher–student pairs spanning 1.5B to 235B parameters and three model families in both thinking and non-thinking modes, covering the most commonly used student–teacher pairs in the OPD literature. We find that Semi-OPD outperforms OPD on 14 of them, with up to +13.6% accuracy and 11.4× training speedup as shown in Fig. 1, and achieves comparable performance (within a 0.1%–2.6% gap) on the remaining three. This phenomenon motivates our further investigation into how to choose between Semi-OPD and OPD and why Semi-OPD is often better.

How to choose between Semi-OPD and OPD. We find that the gain of Semi-OPD over OPD is strongly correlated with the alignment between the initial teacher and student, which can be quantified by the teacher–student top-k overlap ratio over output tokens, as shown in Fig. 2a. The accuracy evolution curves in Fig. 2b further show that (1) when this overlap ratio is low, continually regenerating rollouts from the evolving student can shift training toward increasingly teacher-misaligned trajectories, making strict on-policyness counterproductive; (2) OPD is only beneficial when the teacher and student are aligned with a high overlap ratio.

Why OPD often underperforms Semi-OPD. Our empirical study suggests that effective distillation requires "on-policyness" w.r.t. both the student and the teacher, where the former mitigates exposure bias and the latter determines the effectiveness of the distillation signal. OPD improves the former but may reduce the latter. Specifically, for misaligned pairs, student rollouts from OPD can become increasingly off-policy w.r.t. the teacher as context length grows. This cannot be addressed by simple truncation, as we empirically show the importance of covering supervision signals from different reasoning stages. In contrast, Semi-OPD on initial student rollouts usually distills over

shorter contexts while covering full trajectories, making it more stable. In other words, Semi-OPD trades some student on-policyness for teacher on-policyness, which is also echoed by our observation that Semi-OPD exposes the student to more tokens preferred by the teacher.

We aim to inspire the community to rethink when to use OPD and to study stronger OPD variants under meaningful settings. The key takeaways of our work are summarized below:

- OPD is only useful for aligned teacher–student pairs with high overlap in output tokens; for misaligned pairs, Semi-OPD is a better alternative in both accuracy and training efficiency.
- Effective distillation requires both the on-policyness of training rollout w.r.t. the student and closeness to the teacher; OPD prioritizes the former but may sacrifice the latter.
- Given the target domain and initial teacher–student pair, the choice between OPD and Semi-OPD should be guided by the measured overlap ratio, rather than blindly choosing OPD.
- For future research on OPD, teacher–student pairs should be carefully selected to ensure a meaningful experimental setting.

2 An Efficient OPD Alternative: Semi-On-Policy Distillation

2.1 Preliminaries of OPD

Formally, given a prompt $x \sim \mathcal{D}$, a student LLM π_θ generates a response $y = (y_1, \ldots, y_T)$ autoregressively, visiting prefix states $s_t = (x, y_{<t})$. The underlying sequence-level objective is the reverse $\mathrm{KL}(\pi_\theta(\cdot \mid x) \,\|\, \pi_T(\cdot \mid x))$ with respect to a fixed teacher π_T. In practice, a common approximation uses a local token-level objective that evaluates the teacher–student log-probability ratio only on tokens sampled by the student, avoiding a full-vocabulary comparison at every position (Lu & Lab, 2025; Team et al., 2026). This yields the sampled-token objective:

$$\mathcal{L}_{OPD}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\, y\sim\pi_\theta(\cdot|x)} \left[\sum_{t=1}^{T} -A_t \cdot \log \pi_\theta(y_t \mid s_t) \right], \tag{1}$$

where the advantage $A_t = \log \pi_T(y_t \mid s_t) - \log \pi_\theta(y_t \mid s_t)$ is treated as a constant in the gradient.

As discussed in Sec. 1, recent work has identified several undesirable behaviors in OPD, such as repeated self-correction (Luo et al., 2026; Yang et al., 2026a; Fu et al., 2026a), ineffective supervision on later tokens (Zhang et al., 2026; Ziheng et al., 2026), and unreliable supervision once the student commits to an incorrect direction (Xu et al., 2026b; Jiang et al., 2026). To study the necessity of on-policy sampling for distillation, we propose a simple alternative, Semi-OPD, in Sec. 2.2.

2.2 Semi-OPD: Formulation

Let π_0 denote the initial student and π_θ the student being trained. OPD samples each training response from the current student, $y \sim \pi_\theta(\cdot \mid x)$, so the training data changes as θ is updated. Semi-OPD instead samples responses once from the initial student before training begins,

$$\mathcal{D}_0 = \{(x_i, y_i)\}_{i=1}^{N}, \qquad y_i \sim \pi_0(\cdot \mid x_i). \tag{2}$$

Semi-OPD reuses this fixed set throughout training to minimize the same objective as OPD:

$$\mathcal{L}_{\mathrm{Semi-OPD}}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}_0} \Big[\sum_{t=1}^{T} -A_t \cdot \log \pi_\theta(y_t \mid s_t) \Big]. \tag{3}$$

Although the rollouts are fixed, the student's log-probabilities, and hence the advantage A_t, are computed using the current parameters θ at each step, so the supervision signal continues to evolve.

Training efficiency of Semi-OPD. Semi-OPD decouples data collection from optimization. We first generate all student rollouts from π_0; after these responses are generated, we run the teacher only once to compute the teacher log-probabilities; finally, we train the student on the fixed dataset with fixed teacher log-probabilities, without performing either online rollout generation or teacher forward passes. In addition, rollout generation, teacher scoring, and student optimization are executed as separate phases, each of which can use the full GPU cluster rather than partitioning resources among concurrent student generation, teacher serving, and training.

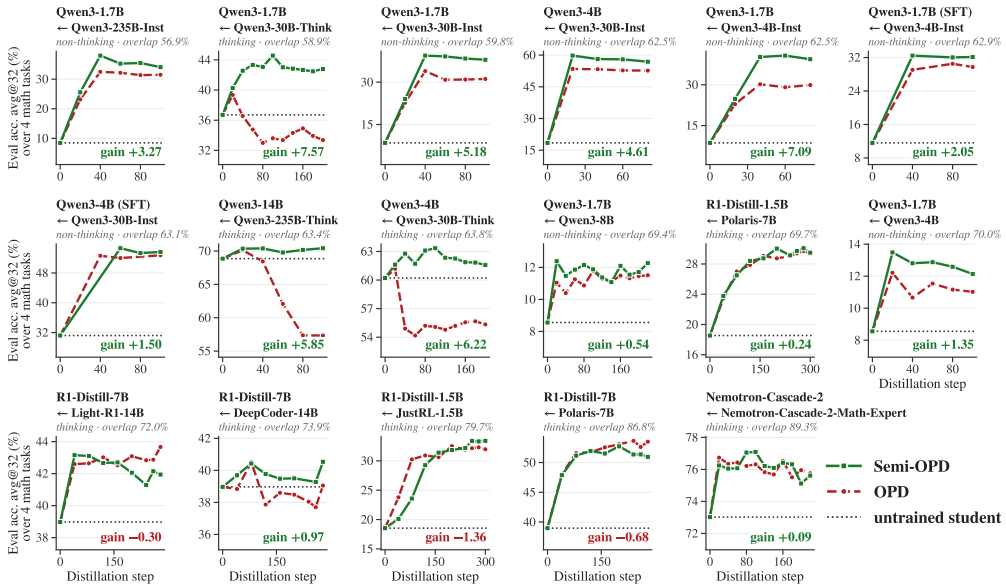


Figure 3: **Semi-OPD vs. OPD across 17 model pairs.** Each panel represents one student–teacher pair, with its title containing the pair's initial top-64 overlap. Accuracy is measured as the mean avg@32 over AIME 2024/2025/2026 and HMMT February 2026. The number in each panel is the gain plotted in Fig. 2a, defined as the area between the two curves divided by the training horizon.

Comparison with Lightning OPD. A prior work, Lightning OPD (Wu et al., 2026a), starts from a base student model and (1) performs supervised fine-tuning (SFT) on teacher rollouts and (2) performs distillation on offline student rollouts. Because it always starts from a base model, which cannot generate high-quality rollouts, it needs to retain step (1). Our Semi-OPD essentially drops step (1) because we start from instruction-tuned student models, which aligns with real-world use cases. We show in Sec. 2.3/ 3 that Semi-OPD outperforms Lightning OPD and that SFT on teacher rollouts has negative effects. In addition, we leverage our method to study when OPD should be used and the underlying reasons behind its failures, providing insights into best practices for OPD.

Top-K overlap ratio for measuring initial teacher–student alignment. To measure the alignment between teacher–student pairs and cover diverse combinations in our experiments, we follow Li et al. (2026) and compare the student's and teacher's top-K token sets at each response position of the initial student's rollouts. Specifically, let $S_K(t) = \mathrm{TopK}(\pi_0(\cdot \mid s_t))$ and $T_K(t) = \mathrm{TopK}(\pi_T(\cdot \mid s_t))$ denote the student's and teacher's top-K token sets, respectively. We define the top-K overlap ratio at position t as

$$\mathrm{Overlap@K}(t) = \frac{|S_K(t) \cap T_K(t)|}{K}.$$

Because both sets contain K tokens, this quantity is 1 when the two distributions have identical top-K supports and 0 when their supports are disjoint. We report the token-weighted mean over all response tokens. Unless otherwise noted, we use $K = 64$.

2.3 Benchmarking Semi-OPD vs. OPD and its Variants

Experiment setup. We conduct experiments on 17 student–teacher pairs (full model-pair names are provided in Tab. 3 in the appendix), spanning the Qwen3 (Yang et al., 2025), DeepSeek-R1-Distill (DeepSeek-AI, 2025), and Nemotron-Cascade-2 (Yang et al., 2026b) families. Qwen3 students (1.7B, 4B, and 14B) are distilled from Instruct and Thinking teachers with up to 235B-A22B parameters, using either original or SFT-warm-started initializations; DeepSeek-R1-Distill students (1.5B and 7B) are distilled from RL-trained descendants of the same checkpoints, including JustRL (He et al., 2025), Polaris (An et al., 2025), Light-R1 (Liang Wen, 2025), and DeepCoder (Luo et al., 2025); and the Nemotron-Cascade-2 pair consists of the student checkpoint before OPD and its math-domain teacher used for the OPD step in Yang et al. (2026b). The pairs cover both non-thinking and thinking modes, with training and evaluation matched within each pair. All models are distilled on DAPO-Math (Yu et al., 2025). Mathematical capability is measured by avg@32 accuracy

Method	AIME24	AIME25	AIME26	HMMT Feb26	Avg
Qwen3-1.7B (Non-think) ← Qwen3-4B-Instruct-2507				Initial overlap ratio: 62.5%	
Teacher	60.4	46.0	52.2	31.2	47.5
Student, untrained	12.6	9.6	7.4	6.3	9.0
Standard OPD[§]	35.8	25.5	23.3	20.1	26.2
Speculative KD (Xu et al., 2024)[§]	33.1	30.7	28.9	20.1	28.2
FastOPD (Zhang et al., 2026)[§]	42.3	30.4	26.4	23.6	30.7
Lightning OPD (Wu et al., 2026a)	41.0	28.8	27.5	22.7	30.0
Relay-OPD (Xu et al., 2026b)[§]	42.7	32.8	30.5	24.7	32.7
Semi-OPD (ours)	**50.3**	**38.9**	**41.8**	**28.0**	**39.8**
Qwen3-4B (Non-think) ← Qwen3-30B-A3B-Instruct-2507				Initial overlap ratio: 62.5%	
Teacher	72.9	62.0	72.8	40.6	62.1
Student, untrained	23.6	18.0	16.8	14.9	18.3
Standard OPD	55.1	45.3	49.1	30.0	44.9
FastOPD (Zhang et al., 2026)	63.5	55.5	56.2	34.9	52.5
Lightning OPD (Wu et al., 2026a)	64.0	56.8	54.3	33.5	52.2
Relay-OPD (Xu et al., 2026b)	61.1	48.1	52.5	32.8	48.6
Semi-OPD (ours)	**66.4**	**63.1**	**62.9**	**37.4**	**57.4**
Nemotron-Cascade-2-30B-A3B ← Nemotron-Cascade-2-Math-Expert				Initial overlap ratio: 89.3%	
Teacher	82.8	80.0	76.4	58.2	74.4
Student, untrained	77.9	72.4	68.8	49.3	67.1
Standard OPD	83.0	80.1	75.8	57.5	74.1
FastOPD (Zhang et al., 2026)	83.0	**80.7**	**77.1**	**58.4**	**74.8**
Lightning OPD (Wu et al., 2026a)	**83.2**	79.1	75.7	56.8	73.7
Relay-OPD (Xu et al., 2026b)	**83.2**	80.1	73.3	57.7	73.6
Semi-OPD (ours)	83.0	79.7	77.0	56.8	74.1

Table 1: Semi-OPD outperforms other OPD variants when the initial student–teacher overlap ratio is low, and remains comparable when the ratio is high. Results are reported as avg@32 at temperature 1.0, averaged over the checkpoints at steps 60/80/100. [§]Reported from (Xu et al., 2026b).

over AIME 2024–2026 and HMMT February 2026, using evaluation rollouts capped at 32k tokens (40k for Nemotron-Cascade-2), while cross-domain generalization is assessed on HumanEval and HumanEval+ (Chen et al., 2021). More details are provided in Appendix A.

Semi-OPD vs. OPD across 17 model pairs. Fig. 3 visualizes the average accuracy evolution during training achieved by Semi-OPD and OPD across all 17 teacher–student pairs. We order the pairs by the overlap ratio of the initial teacher–student pair, which is annotated above each plot. We also report the average accuracy gain of Semi-OPD over OPD, defined as the area between the Semi-OPD and OPD learning curves divided by their shared training horizon.

We observe that Semi-OPD outperforms OPD on 14 of 17 pairs with improvements of up to $\sim$13% over OPD at the last step, and the largest gains are generally achieved on pairs with lower initial overlap. On the remaining three pairs, Semi-OPD still achieves comparable performance, with gaps of 0.1–2.6%. The difference is especially pronounced for Qwen3 thinking-mode pairs: OPD undergoes a substantial performance collapse during training, whereas Semi-OPD continues to improve.

Comparison with OPD variants in accuracy and training efficiency. In addition to standard OPD, we also compare Semi-OPD with recent OPD variants across three model pairs with different model sizes, overlap ratios, and reasoning modes in Tab. 1. We observe that (1) on the Qwen3-1.7B (non-thinking) and Qwen3-4B (non-thinking) students, which have low teacher–student overlap ratios, Semi-OPD outperforms all other baselines in accuracy while also being the most efficient, e.g., achieving +7.1% and +8.8% higher accuracy with 35$\times$ and 19.5$\times$ less training time than Relay-OPD on the two pairs, respectively, according to Fig. 1 and Tab. 4. (2) On the Nemotron-Cascade-2 pair, which has a high overlap ratio of approximately 89%, our Semi-OPD remains competitive with other OPD baselines and falls only slightly behind the best baseline by 0.7% accuracy, while reducing the training time by 2.2x as compared to FastOPD and 4.5x compared to OPD.

The overall accuracy–training-efficiency trade-off is shown in Fig. 1, where Semi-OPD achieves up to an 11.4$\times$ speedup over standard OPD and a 2.7$\times$ speedup over the fastest baseline, FastOPD. In contrast, another OPD variant, Relay-OPD, which also aims to make student rollouts closer to the teacher, requires interleaved teacher and student generation, resulting in lower efficiency.

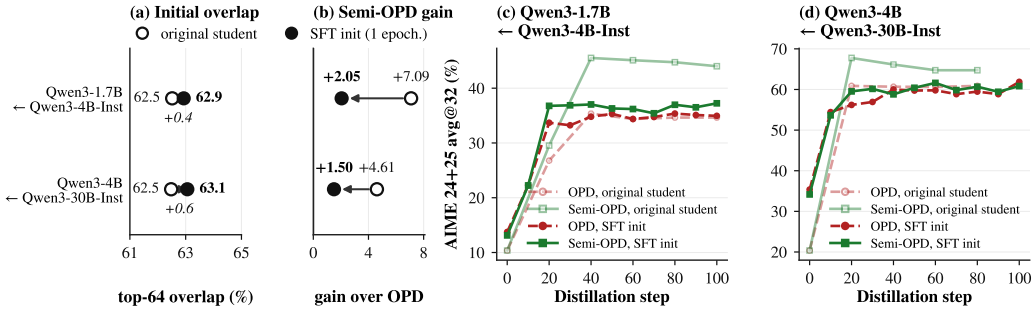


Figure 4: **Teacher-rollout SFT improves overlap ratios but reduces final performance after Semi-OPD. (a)** SFT on teacher rollouts slightly increases the initial top-64 overlap. **(b)** SFT on teacher rollouts followed by Semi-OPD still outperforms OPD but underperforms Semi-OPD alone. **(c, d)** SFT on teacher rollouts negatively impacts final performance when combined with Semi-OPD.

3 How to Choose Between OPD and Semi-OPD?

As observed in Sec. 2.3 and Fig. 3, the relative performance of OPD and Semi-OPD is highly correlated with the teacher–student overlap ratio. It can therefore serve as an indicative metric for determining when to use OPD or Semi-OPD. We characterize this correlation more formally here.

OPD is needed for teacher–student pairs with high overlap ratios; otherwise, Semi-OPD is better and faster. We compute the correlation between Semi-OPD's gain over OPD, measured as the area between their learning curves normalized by their shared training horizon, and the overlap ratio of each teacher–student pair in Fig. 2. The two quantities are strongly negatively correlated (Pearson's $r = -0.73$ across 17 pairs): pairs with lower initial overlap ratios tend to benefit substantially more from Semi-OPD, while OPD is only needed for pairs with high overlap ratios.

This pattern implies that, when the student differs substantially from the teacher, repeatedly sampling from the evolving student policy can cause online rollouts to degrade (several examples are shown in Appendix D) and weaken the supervision signal; reusing rollouts from the initial student can provide a more stable distillation signal. We dive into the underlying reason in Sec. 4.

Key insights. The above phenomenon provides useful implications for the practical use and research of OPD: (1) Given a target teacher–student pair and target-domain prompts, determine whether to use OPD or Semi-OPD based on their overlap ratio, rather than blindly using OPD; (2) For future studies of improved OPD schemes in LM post-training pipelines, carefully select the teacher–student pair used as the experimental setting; otherwise, Semi-OPD can serve as a strong baseline.

Does SFT on teacher rollouts close the gap? We also experiment with two pairs in which the students undergo one epoch of SFT on teacher-generated rollouts, followed by OPD or Semi-OPD. This matches the setup of Lightning OPD (Wu et al., 2026a). As shown in Fig. 4, we observe that (1) SFT on teacher rollouts slightly increases teacher–student overlap ratios, while Semi-OPD still outperforms OPD on both pairs, albeit by a smaller margin; and (2) Our Semi-OPD without the additional teacher-rollout SFT stage can outperform Semi-OPD with this stage, indicating that SFT on teacher rollouts is unnecessary. We also note that, while Wu et al. (2026a) frames fixed rollouts primarily as an efficient approximation to OPD, our results show that freezing the rollout cache can improve distillation when the initial teacher–student overlap is low.

4 Why Does OPD Often Underperform Semi-OPD?

We further investigate why OPD underperforms Semi-OPD in low-overlap cases.

Our hypothesis. We hypothesize that effective distillation requires training data (i.e., rollouts) to satisfy two properties: (1) on-policyness w.r.t. the student to avoid exposure bias, and (2) high-quality supervision from the teacher, which is often determined by on-policyness w.r.t. the teacher distribution, i.e., how well the rollouts align with the teacher's own inference-time distribution. OPD prioritizes the former while struggling with the latter, especially for low-overlap teacher–student pairs; Semi-OPD trades off some student on-policyness for better supervision signals.

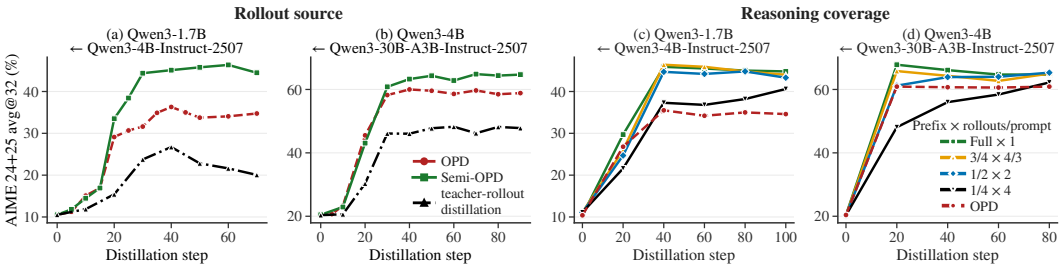


Figure 5: **Both rollout source and reasoning coverage matter for distillation. (a, b)** Semi-OPD outperforms teacher-rollout distillation (and OPD) on both pairs. **(c, d)** We compare Semi-OPD trained on the first $1/4$, $1/2$, $3/4$, or all of each rollout, while keeping the total number of training tokens fixed by scaling the number of rollouts per prompt. Distillation on all stages performs best.

More specifically, for low-overlap pairs, rollouts become increasingly longer during OPD and exhibit various artifacts (Luo et al., 2026; Yang et al., 2026a; Fu et al., 2026a). This makes later tokens in a rollout increasingly off-policy w.r.t. the teacher's distribution, making teacher supervision on them unhelpful or even harmful (Xu et al., 2026b; Jiang et al., 2026). Truncating the rollouts and training only on the prefix can mitigate this issue (Zhang et al., 2026; Ziheng et al., 2026), but at the cost of removing supervision on later tokens. As demonstrated later, covering supervision signals across all reasoning stages in full trajectories is important for final performance. In contrast, Semi-OPD trains on rollouts from the initial student, which are usually shorter than on-policy rollouts, thereby better maintaining on-policyness w.r.t. the teacher while still covering full trajectories with all reasoning stages. We support the above analysis through the following study.

Student on-policyness matters: Semi-OPD vs. teacher-rollout distillation. We study the importance of student on-policyness to Semi-OPD by replacing the initial-student rollout cache with teacher-generated rollouts while keeping all other settings unchanged. As shown in Fig. 5(a, b), Semi-OPD consistently outperforms teacher-rollout distillation. This indicates that the benefit of Semi-OPD does not come merely from using a fixed rollout cache. Rather, retaining student-generated trajectories to maintain student on-policyness is also important.

Training on longer rollouts can hurt OPD for low-overlap pairs. As shown in Fig. 6, we vary the rollout length (i.e., the rollout cap) used for OPD training and observe that increasing the rollout cap can hurt the low-overlap pair while benefiting the higher-overlap pair. We hypothesize that, for low-overlap pairs, training on long rollouts that are off-policy w.r.t. the teacher may tend to reinforce superficial reasoning patterns without sufficiently improving the underlying reasoning process, causing subsequent student rollouts to deteriorate (examples comparing rollouts are in Appendix D).

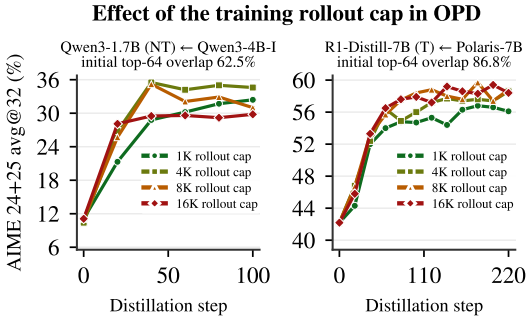


Figure 6: Longer rollouts hurt the low-overlap pair (left) but not the high-overlap pair (right).

Supervision signals covering all reasoning stages are essential. If long rollouts can be harmful, one simple solution is to truncate them, at the cost of dropping supervision from later reasoning stages. To isolate the value of reasoning-stage coverage, we perform an ablation by truncating the cached Semi-OPD traces to the first $1/4$, $1/2$, or $3/4$ of their length, while scaling the number of rollouts per prompt to keep the total training token budget fixed. As shown in Fig. 5(c, d), restricting supervision to the first quarter substantially slows convergence and reduces final accuracy, whereas training on full trajectories achieves the best peak accuracy. Thus, both covering all reasoning stages and maintaining on-policyness w.r.t. the teacher are important.

Semi-OPD exposes the student to more tokens preferred by the teacher. Semi-OPD exactly satisfies the above requirement by providing full trajectory coverage with shorter rollout lengths. To analyze the resulting distillation signal, we measure the percentage of training tokens with $p_T > p_S$,

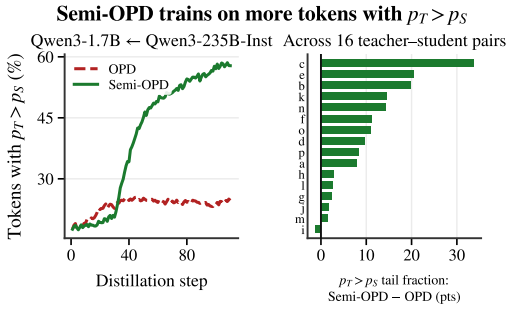


Figure 7: **The percentage of tokens with positive rewards**, i.e., fraction of training tokens with $p_T > p_S$ under Semi-OPD and OPD.

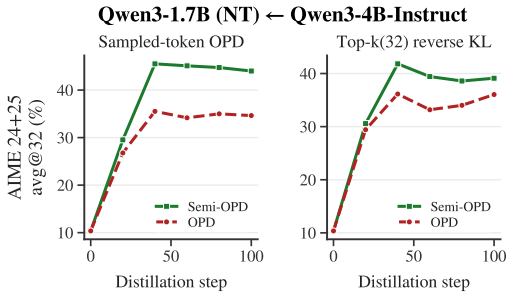


Figure 8: **Semi-OPD vs. OPD under two distillation objectives**, i.e., the sampled-token and top-k reverse KL objectives.

i.e., tokens that receive a positive reward due to higher probabilities assigned by the teacher. As shown in Fig. 7, Semi-OPD maintains a larger fraction of such tokens than OPD throughout training. This indicates that Semi-OPD exposes the student to more teacher-preferred tokens, i.e., more on-policy w.r.t. the teacher, whereas OPD more often produces signals that suppress the student's own sampled tokens. Importantly, Semi-OPD also outperforms OPD under top-k reverse KL, as shown in Fig. 8, indicating that its advantage is not specific to the sampled-token objective.

Together, these findings suggest that Semi-OPD benefits from preserving student-anchored rollouts that remain effective teacher supervision and broad coverage of the reasoning process. These advantages can outweigh the benefits of fully on-policy sampling under large teacher–student mismatch.

5 More Intriguing Observations and Analysis

Beyond the main comparison, we observe several interesting phenomena that provide a more complete picture of Semi-OPD versus OPD. More interesting observations are provided in the appendix.

OPD converges to a rollout-determined equilibrium, regardless of initialization. A natural question is whether OPD can further improve after Semi-OPD. We study this by initializing OPD from Semi-OPD or teacher-rollout distillation checkpoints as warm starts. As shown in Fig. 9, regardless of initialization, all runs eventually converge to the performance level reached by OPD initialized from the initial student model under the same rollout cap, while the warm start only affects the convergence trajectory. This suggests that repeated on-policy sampling drives distillation toward an equilibrium determined by the rollout data distribution rather than by the initial checkpoint. Characterizing this equilibrium remains an interesting direction for future work.

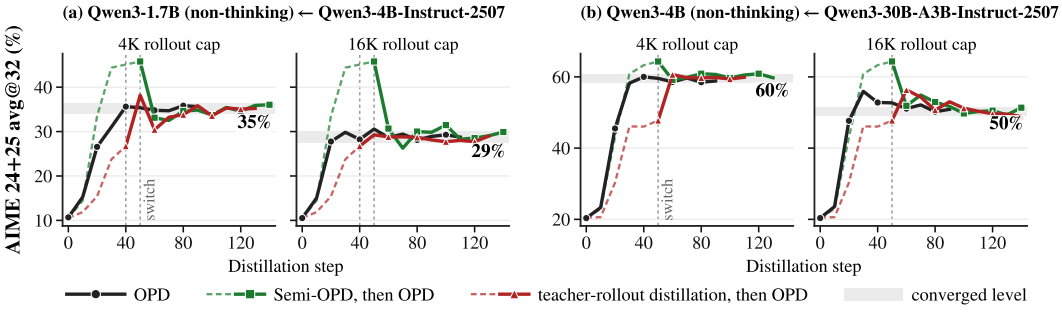


Figure 9: The OPD accuracy evolution curves when initialized from the initial student, a Semi-OPD checkpoint, or a teacher-rollout distillation checkpoint (dashed lines: before switching to OPD).

The gains transfer beyond the training domain. As shown in Fig. 10, models distilled only on math-heavy datasets (Yu et al., 2025) also improve on HumanEval and HumanEval+, with Semi-OPD remaining ahead of OPD. This suggests that the gains are not limited to fitting the math training distribution but transfer to broader reasoning tasks.

Semi-OPD requires adequately strong initial rollouts. Since Semi-OPD trains only on rollouts from π_0, the initial rollouts need to meet a minimum quality threshold for effective distillation. For

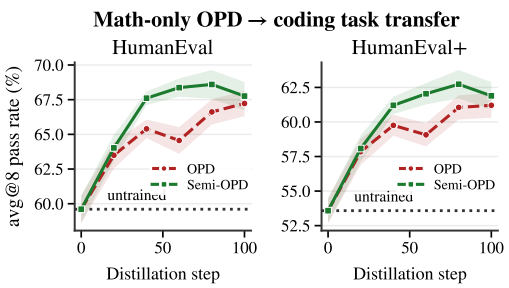


Figure 10: **Transfer to coding.** Code performance of models distilled on math only (Qwen3-1.7B ← Qwen3-4B-Instruct-2507).

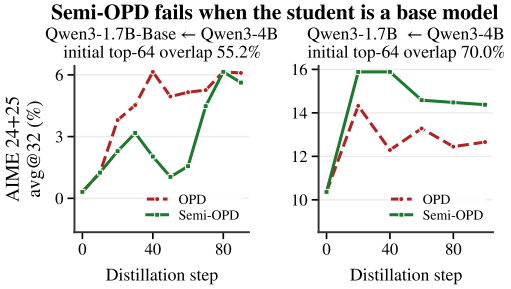


Figure 11: **Initial student competence.** Semi-OPD vs. OPD with the same teacher, from base (left) and instruction-tuned (right) students.

example, as shown in Fig. 11, with a base student, the initial rollouts are too weak and Semi-OPD underperforms OPD; with an instruction-tuned student, Semi-OPD regains its advantage. Thus, Semi-OPD is most suitable when the initial student already produces meaningful trajectories.

6 Related Work

On-Policy Distillation and Its Mechanics. OPD trains a student on trajectories sampled from its own policy, while a teacher provides dense supervision through reverse-KL or related objectives (Agarwal et al., 2024; Gu et al., 2024; Lu & Lab, 2025; Song & Zheng, 2026). By optimizing on the student's own visitation distribution, OPD mitigates the exposure bias of offline distillation (Hinton et al., 2015; Kim & Rush, 2016), an idea shared with interactive imitation learning methods such as DAgger (Ross et al., 2011). OPD has recently become a popular post-training stage for reasoning models (Xiaomi, 2026; Yang et al., 2025; Xu et al., 2026a; Team et al., 2026), motivating recent work to better understand its underlying mechanics (Jin et al., 2026; Fu et al., 2026a; Armandpour et al., 2026). For example, successful OPD is linked to teacher–student compatibility (Li et al., 2026), while even a single training example or 1% of tokens can recover much of full-data OPD (Fu et al., 2026b; Sheng et al., 2026). This raises a research question: which parts of on-policy distillation are necessary?

Failure Modes in OPD. Recent work has identified several failure modes of OPD: student rollouts can grow longer, fall into repeated self-correction, and fail to terminate (Luo et al., 2026; Yang et al., 2026a; Fu et al., 2026a; Yu et al., 2026), and later portions of a rollout can drift away from the teacher distribution, yielding increasingly unreliable supervision (Ziheng et al., 2026; Xu et al., 2026b; Jiang et al., 2026; Wang et al., 2026). Existing methods mitigate these issues by truncating student rollouts (Zhang et al., 2026; Ziheng et al., 2026), letting the teacher intervene during generation (Xu et al., 2024; 2026b), refining student trajectories with the teacher (Jiang et al., 2026), warm-starting OPD on teacher data (Li et al., 2026), or constraining updates to a trust region (Xing et al., 2026; Xie et al., 2026). Other work relaxes strict on-policyness for efficiency, using fixed rollouts from a teacher-SFT reference policy (Wu et al., 2026a;b) or policy-lagged student rollouts in asynchronous training (Kang et al., 2026). We study the simplest form of this relaxation: rollouts are sampled once from the initial student, without any teacher-SFT stage, and no refreshing. Beyond efficiency gain, we find that freezing rollouts can also improve accuracy. Moreover, rather than focusing on a few model pairs, we evaluate 17 diverse model pairs and show that this benefit is strongly correlated with the initial teacher–student top-k overlap, with lower-overlap pairs benefiting more.

7 Conclusion

This work revisits when on-policy sampling is truly necessary for LM distillation. By systematically comparing OPD with the proposed Semi-OPD across diverse teacher–student pairs, we show that continually regenerating student rollouts is often unnecessary and can even be harmful, especially when the teacher and student are poorly aligned. Our analyses highlight a key principle: effective distillation requires balancing on-policyness w.r.t. both the student and the teacher, with the initial teacher–student overlap serving as a practical indicator of when to use OPD or Semi-OPD. Beyond providing an efficient alternative to OPD, our findings motivate future work to rethink on-policy sampling as a design choice rather than a default, and to develop distillation methods that explicitly account for teacher–student alignment and rollout quality.

AI USE STATEMENT

We use AI for implementing some experiments that we instructed it to do, and for preparing summarizations of experiment results to help us analyze the results. In paper writing, we use AI for revising the grammar and plotting the figures in the paper. The main idea, research direction, and overall design of the work were developed by ourselves, and AI was used to assist with implementation, analysis, and grammar revision.

REPRODUCIBILITY STATEMENT

Semi-OPD has a simple implementation: student rollouts are generated once, scored by the teacher, and then reused throughout distillation. The main goal of this work is to provide a systematic empirical analysis of when on-policy distillation is beneficial, rather than to introduce a new method. We document the datasets, teacher–student model pairs, training configurations, and evaluation protocols used in all experiments in detail as shown in Appendix A. We also keep the experimental setup matched across methods whenever possible, so that the reported comparisons can be reproduced from the described settings.

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 *The twelfth international conference on learning representations*, 2024.

Chenxin An, Zhihui Xie, Xiaonan Li, Lei Li, Jun Zhang, Shansan Gong, Ming Zhong, Jingjing Xu, Xipeng Qiu, Mingxuan Wang, and Lingpeng Kong. Polaris: A post-training recipe for scaling reinforcement learning on advanced reasoning models, 2025. URL https://hkunlp.github.io/blog/2025/Polaris.

Mohammadreza Armandpour, Fatih Ilhan, David Harrison, Ajay Jaiswal, Duc NM Hoang, Fartash Faghri, Yizhe Zhang, Minsik Cho, and Mehrdad Farajtabar. Unmasking on-policy distillation: Where it helps, where it hurts, and why. *arXiv preprint arXiv:2605.10889*, 2026.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

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*, 2026a.

Zixuan Fu, Bingxiang He, Yuxin Zuo, Haohuan Huang, Jinqian Zhang, Ruhang Xiao, Cheng Qian, Qinyu Luo, Huan-ang Gao, Yudong Wang, et al. Rethinking on-policy distillation of large language models ii: One training example. *arXiv preprint arXiv:2609.04172*, 2026b.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. In *ICLR*, 2024.

Bingxiang He, Zekai Qu, Zeyuan Liu, Yinghao Chen, Yuxin Zuo, Cheng Qian, Kaiyan Zhang, Weize Chen, Chaojun Xiao, Ganqu Cui, et al. Justrl: Scaling a 1.5 b llm with a simple rl recipe. *arXiv preprint arXiv:2512.16649*, 2025.

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

Li Jiang, Haoran Xu, Yichuan Ding, and Amy Zhang. Trajectory-refined distillation. *arXiv preprint arXiv:2606.08432*, 2026.

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. *arXiv preprint arXiv:2603.07079*, 2026.

Wonjun Kang, Kevin Galim, Seunghyuk Oh, Minjun Kang, Sanghyun Park, Donghoon Kim, Minjae Lee, Minseo Kim, Rishabh Tiwari, Yuchen Zeng, et al. Asyncopd: How stale can on-policy distillation be? *arXiv preprint arXiv:2606.24143*, 2026.

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.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huanang Gao, Wenkai Yang, Zhiyuan Liu, et al. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe. *arXiv preprint arXiv:2604.13016*, 2026.

Fenrui Xiao Xin He Qi An Zhenyu Duan Yimin Du Junchen Liu Lifu Tang Xiaowei Lv Haosheng Zou Yongchao Deng Shousheng Jia Xiangzheng Zhang Liang Wen, Yunke Cai. Light-r1: Curriculum sft, dpo and rl for long cot from scratch and beyond, 2025. URL https://github.com/Qihoo360/Light-R1.

Kaiyuan Liu, Ziyuan Zhuang, Yang Bai, Bing Wang, Rongxiang Weng, and Jieping Ye. Prefix teach, suffix fade: Local teachability collapse in strong-to-weak on-policy distillation. *arXiv preprint arXiv:2605.13643*, 2026.

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. *arXiv preprint arXiv:2604.08527*, 2026.

Michael Luo, Sijun Tan, Roy Huang, Ameen Patel, Alpay Ariyak, Qingyang Wu, Xiaoxiang Shi, Rachel Xin, Colin Cai, Maurice Weber, Ce Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepcoder: A fully open-source 14b coder at o3-mini level. https://pretty-radio-b75.notion.site/DeepCoder-A-Fully-Open-Source-14B-Coder-at-O3-mini-Level-1cf81902c14680b3bee5eb349a512a51, 2025. Notion Blog.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

Huanxin Sheng, Zhiling Ye, Haonan Wang, Jian Wang, Jinjie Gu, and Jian Kang. 1% of tokens can be enough: On gradient estimation in on-policy distillation. *arXiv preprint arXiv:2609.24432*, 2026.

Mingyang Song and Mao Zheng. A survey of on-policy distillation for large language models. *arXiv preprint arXiv:2604.00626*, 2026.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y Charles, et al. Kimi k3: Open frontier intelligence. *arXiv preprint arXiv:2607.24653*, 2026.

Rui Wang, Hongru Wang, Yi Chen, Boyang Xue, Tianqing Fang, Wenhao Yu, and Kam-Fai Wong. Demystifying on-policy distillation: Roles, pathologies, and regulations. *arXiv preprint arXiv:2607.13399*, 2026.

Yecheng Wu, Song Han, and Hai Cai. Lightning opd: Efficient post-training for large reasoning models with offline on-policy distillation. *arXiv preprint arXiv:2604.13010*, 2026a.

Yecheng Wu, Song Han, and Han Cai. Lightning opd 2.0: Mitigating style bias in cross-teacher on-policy distillation for large reasoning models. *arXiv preprint arXiv:2607.28449*, 2026b.

LLM-Core Xiaomi. Mimo-v2-flash technical report, 2026. URL https://arxiv.org/abs/2601.02780.

Zhengpeng Xie, Li Lyna Zhang, Zeke Xie, and Mao Yang. Trust region policy distillation. *arXiv preprint arXiv:2607.04751*, 2026.

Xingrun Xing, Haoqing Wang, Boyan Gao, Ziheng Li, and Yehui Tang. Trust region on-policy distillation. *arXiv preprint arXiv:2606.01249*, 2026.

Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, et al. Deepseek-v4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026a.

Haolei Xu, Xiaowen Xu, Haiwen Hong, Zixuan Ni, Hongxing Li, Yiwen Qiu, Weiming Lu, and Yongliang Shen. Pass the baton: Trajectory-relayed on-policy distillation. *arXiv preprint arXiv:2607.26057*, 2026b.

Wenda Xu, Rujun Han, Zifeng Wang, Long Le, Dhruv Madeka, Lei Li, William Yang Wang, Rishabh Agarwal, Chen-Yu Lee, and Tomas Pfister. Speculative knowledge distillation: Bridging the teacher-student gap through interleaved sampling. In *The Thirteenth International Conference on Learning Representations*, 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. *arXiv preprint arXiv:2505.09388*, 2025.

Yuxiao Yang, Tianrun Yu, Shangzhe Li, Kaixiang Zhao, Xuchao Zhang, Chetan Bansal, Huaxiu Yao, Taylor W Killian, and Weitong Zhang. When eos tokens disagree: Understanding length inflation in on-policy distillation. *arXiv preprint arXiv:2609.20511*, 2026a.

Zhuolin Yang, Zihan Liu, Yang Chen, Wenliang Dai, Boxin Wang, Sheng-Chieh Lin, Chankyu Lee, Yangyi Chen, Dongfu Jiang, Jiafan He, et al. Nemotron-cascade 2: Post-training llms with cascade rl and multi-domain on-policy distillation. *arXiv preprint arXiv:2603.19220*, 2026b.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, et al. Dapo: An open-source llm reinforcement learning system at scale, 2025. *URL https://arxiv. org/abs/2503.14476*, 2025.

Zichao Yu, Chengzhi Yu, Shengze Xu, Yujin Han, Bingqing Jiang, Xu Wang, and Difan Zou. Mismatch matters: On-policy distillation beyond token agreement. *arXiv preprint arXiv:2608.09836*, 2026.

Dongxu Zhang, Zhichao Yang, Sepehr Janghorbani, Jun Han, Andrew Ressler II, Qian Qian, Gregory D Lyng, Sanjit Singh Batra, and Robert E Tillman. Fast and effective on-policy distillation from reasoning prefixes. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 25553–25569, 2026.

Zhou Ziheng, Jiaqi Li, Huacong Tang, Ying Nian Wu, and Demetri Terzopoulos. Less is more: Early stopping rollout for on-policy distillation. *arXiv preprint arXiv:2605.27028*, 2026.

A Training and Evaluation Setup

Models. Table 3 lists all 17 teacher–student pairs used in our experiments. They cover three groups: (i) Qwen3 students (1.7B, 4B, 14B) to be distilled from Qwen3 Instruct-2507 and Thinking-2507 teachers of up to 235B-A22B parameters; (ii) DeepSeek-R1-Distill-Qwen students (1.5B, 7B) to be distilled from RL-trained models of the same checkpoints (JustRL, Polaris, Light-R1, DeepCoder); (iii) one Nemotron Cascade-2 math expert pair. Pairs (g) and (h) start from an SFT checkpoint of the student, after doing 1 epoch of SFT on the teacher generated rollouts. Reasoning mode (thinking T / non-thinking NT) is fixed and matched per pair. Both methods of a pair train and evaluate with the same chat template and thinking mode, and the evaluation mode always matches the training mode. The DeepSeek-R1-Distill chat templates always prefill `<think>`, so we treat those pairs as thinking-mode.

Training data and setup. All runs train on the prompts from the DAPO-Math-17K dataset (14,116 English prompts after deduplication), using 128 prompts per step and one rollout per prompt. One epoch is therefore around 110 steps. For Semi-OPD, we sample one response per prompt from the initial student π_0 before training, using the same sampling parameters and response cap as OPD. OPD reads the prompts in the same order. As a result, the two methods see the same prompts at every step and differ only in which model generated the response. Both methods use the same sampled-token reverse-KL objective. They also share the teacher, learning rate, batch size and evaluation protocol. Table 2 lists the hyperparameters. Additionally, it is worth noting that in our work, Fast OPD is implemented with a training rollout limit of 4K, following Xu et al. (2026b). For only the non-thinking pairs, we also adopt a 4K training rollout length for OPD in the comparisons shown in Fig. 3 and Fig. 2. In Table 1, however, OPD is trained with a 16K rollout limit for all pairs to clearly distinguish with Fast OPD. For exact rollout shape used during training can refer to Table 3 too. Note that this is for training rollout, for evaluation we all use 32k or 40k limit.

Table 2: Training and evaluation hyperparameters.

	Non-thinking pairs	Think/Long-CoT pairs
Optimization		
Optimizer	AdamW (β_1=0.9, β_2=0.999, wd 0.01)	
Learning rate	1×10^{-6}, constant (no warm-up)	
Gradient clipping	1.0	
Precision	bfloat16	
Prompts per step / rollouts per prompt	128 / 1	
Global batch size	128	
Rollouts		
Max prompt length	2,048	
Max response length (training)	4,096	16,384
Sampling temperature / top-p	1.0 / 1.0	
Loss		
Objective	Sampled-token reverse KL	
Evaluation		
Benchmarks	AIME24, AIME25, AIME26, HMMT Feb26	
Metric	avg@32	
Temperature / top-p	1.0 / 1.0	
Max response length	32k (40k for Nemotron model)	

Infrastructure. We train with NeMo-RL (DTensor backend) and generate rollouts and evaluation samples with vLLM. Runs use 1–8 nodes of 8× H100 GPUs, depending on the teacher size.

Table 3: All 17 teacher–student pairs, sorted by AUC gain of Semi-OPD over OPD. Model names link to the exact Hugging Face checkpoints used. Mode: T = thinking / long chain-of-thought, NT = non-thinking. Cap: maximum response length of training rollouts, shared by both Semi-OPD and OPD. Overlap: top-64 overlap between the initial student and the teacher, in %. Last-step: Semi-OPD minus OPD at the last step both methods evaluated. AUC: signed area between the two accuracy curves over the steps both methods evaluated, divided by the length of that span. Gains are avg@32 points on the four-benchmark mean. R1-Distill = DeepSeek-R1-Distill-Qwen; Cascade-2 = Nemotron-Cascade-2 (SFT): the linked student after one epoch of SFT on teacher rollouts. †Evaluated with a 40,960-token response limit; all other pairs use 32,768.

				Train Len	Init.	Gain	
#	Student	Teacher	Mode	Cap	Overlap	Last-step	AUC
n	Qwen3-1.7B	Qwen3-30B-A3B-Thinking	T	16K	58.9	**+9.4**	**+7.57**
a	Qwen3-1.7B	Qwen3-4B-Instruct-2507	NT	4K	62.5	**+9.5**	**+7.09**
p	Qwen3-4B	Qwen3-30B-A3B-Thinking	T	16K	63.8	**+6.2**	**+6.22**
o	Qwen3-14B	Qwen3-235B-A22B-Thinking	T	16K	63.4	**+13.1**	**+5.85**
b	Qwen3-1.7B	Qwen3-30B-A3B-Instruct-2507	NT	4K	59.8	**+6.8**	**+5.18**
f	Qwen3-4B	Qwen3-30B-A3B-Instruct-2507	NT	4K	62.5	**+4.2**	**+4.61**
c	Qwen3-1.7B	Qwen3-235B-A22B-Instruct-2507	NT	4K	56.9	**+2.6**	**+3.27**
g	Qwen3-1.7B (SFT)	Qwen3-4B-Instruct-2507	NT	4K	62.9	**+2.4**	**+2.05**
h	Qwen3-4B (SFT)	Qwen3-30B-A3B-Instruct-2507	NT	4K	63.1	+0.9	+1.50
d	Qwen3-1.7B	Qwen3-4B	NT	4K	70.0	+1.1	+1.35
m	R1-Distill-7B	DeepCoder-14B	T	16K	73.9	+1.5	+0.97
e	Qwen3-1.7B	Qwen3-8B	NT	4K	69.4	+0.8	+0.54
j	R1-Distill-1.5B	Polaris-7B	T	16K	69.7	+0.1	+0.24
q	Cascade-2-30B-A3B	Cascade-2-Math-Expert†	T	16K	89.3	−0.1	+0.09
l	R1-Distill-7B	Light-R1-14B	T	16K	72.0	−1.7	−0.30
k	R1-Distill-7B	Polaris-7B	T	16K	86.8	−2.6	−0.68
i	R1-Distill-1.5B	JustRL-1.5B	T	16K	79.7	+1.4	−1.36

B Training Cost Comparison

Semi-OPD is faster because rollout generation and teacher scoring are done once before training. Similar to Lightning OPD (Wu et al., 2026a), this removes online rollout generation and teacher inference from the training loop, so all GPUs can be used for rollout generation, teacher log-probability scoring, and student training. For a fixed student, the same rollout cache can also be reused across different teachers; evaluating a new teacher only requires computing its log-probabilities on the cached rollouts. This can further reduce the cost of comparing or selecting teachers in practice. In contrast, Relay-OPD requires interleaved student and teacher generation throughout training, leading to higher training cost.

Table 4: **Training cost.** H100 GPU-hours for 100 training steps, excluding evaluation. Semi-OPD's cost is the sum of rollout generation, teacher scoring, and student training. FastOPD is OPD with rollouts truncated to 4K tokens. Speedup is the method's GPU-hours divided by Semi-OPD's.

Method	GPU-hours	Semi-OPD speedup
Qwen3-1.7B (non-thinking) ← Qwen3-4B-Instruct-2507[†]		
Relay-OPD	173.3	34.7×
OPD	56.8	11.4×
FastOPD	11.7	2.3×
Semi-OPD	**5.0**	–
Qwen3-4B (non-thinking) ← Qwen3-30B-A3B-Instruct-2507[†]		
Relay-OPD	604.4	19.5×
OPD	199.1	6.4×
FastOPD	77.3	2.5×
Semi-OPD	**31.0**	–
Qwen3-14B (thinking) ← Qwen3-235B-A22B-Thinking-2507		
Relay-OPD	1546.5	24.6×
OPD	480.0	7.6×
FastOPD	169.8	2.7×
Semi-OPD	**63.0**	–
Nemotron-Cascade-2-30B-A3B ← Nemotron-Cascade-2-Math-Expert		
Relay-OPD	933.5	14.1×
OPD	301.4	4.5×
FastOPD	147.5	2.2×
Semi-OPD	**66.4**	–

[†]Semi-OPD uses 4K training rollouts on these pairs and OPD uses 16K; on the other two pairs both use 16K. Training rollout caps are listed in Table 3.

C LENGTH AND SELF-CORRECTION INFLATION

Fig. 12 tracks two properties of the rollouts each method trains on: rollout length (mean generated tokens per rollout) and self-correction word density. Under OPD, the rollouts are regenerated by the current student at every step. Under Semi-OPD, they are frozen rollouts sampled once from the initial student π_0.

Measuring self-correction. We measure self-correction with a lexical word density. We count case-insensitive, whole-word occurrences of a fixed set of self-correction words, {*wait, but, hmm, actually, hold, however, yet, oh, alternatively, no, ah, oops, well*}, in the generated completion (prompt excluded). At each training step, we pool all 128 training rollouts and report $1000 \times \sum_i h_i / \sum_i c_i$, where h_i is the number of word hits and c_i the number of characters in rollout i. The teacher reference (dotted line) uses the same definition on 128 teacher samples (T=1.0, same rollout cap) on the final training step's prompts; rollout length for the teacher is measured on the same samples. Because the vocabulary includes common discourse words such as *but* and *no*, the metric is a proxy for reflective, hedging style rather than a count of verified corrections. We show more qualitative examples of rollout comparisons between OPD and Semi-OPD in Appendix D.

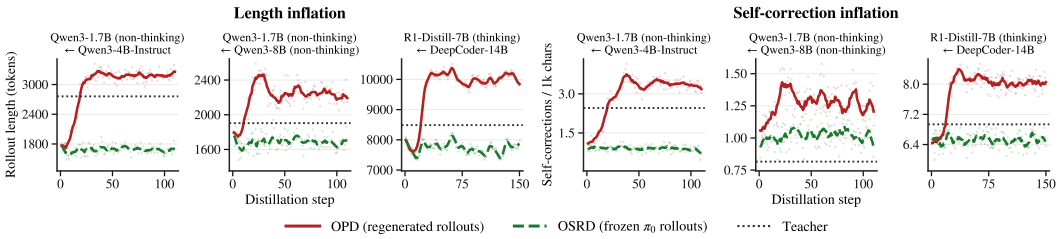


Figure 12: **On-policy rollouts inflate during OPD.** Rollout length (left three panels) and self-correction word density (right three panels) of the training rollouts. OPD rollouts grow longer and use more self-correction words, eventually exceeding the teacher (dotted), while the frozen rollouts sampled once from the initial student π_0 do not.

D QUALITATIVE EXAMPLES: OPD VERSUS SEMI-OPD TRAINING ROLLOUTS

Example 1. Qwen3-1.7B ← Qwen3-30B-A3B-Thinking (thinking). OPD rollout at training step 201; Semi-OPD rollout from π_0; generation cap 16,384 tokens.

Problem. What is the hundreds digit of the sum below?
$1 + 12 + 123 + 1234 + 12345 + 123456 + 1234567 + 12345678 + 123456789$

Semi-OPD (frozen π_0 rollout): 5,688 tokens, 36 reflection words, reaches an answer

[. . . 1,101 tokens omitted . . .]

Now, take mod 1000: 205.

Wait, so the total sum mod 1000 is 205? So the last three digits of the sum are 205. Therefore, the hundreds digit is 2. Is that correct?

[. . . 2,535 tokens omitted . . .]

So total sum is 137,174,205.

Now, the last three digits are 205, so the hundreds digit is 2. So that confirms it.

[. . . 1,380 tokens omitted . . .]

Therefore, the hundreds digit of the sum is 2.

Final Answer

$\boxed{2}$

[. . . 550 tokens omitted . . .]

OPD (step 201): 16,384 tokens, 80 reflection words, hits the cap

[. . . 6,594 tokens omitted . . .]

Sum of hundreds digits: 0+0+1+2+3+4+5+6+7 = 28

Add carryover from tens: 28 + 4 = 32

Hundreds digit of sum: 32 mod 10 = 2

[. . . 2,658 tokens omitted . . .]

Now, the hundreds place of the sum is the second digit from the right, so 32 mod 100 is 32, so the hundreds digit is 3

[. . . 2,458 tokens omitted . . .]

Ah! Here's the mistake: when I said the sum of the hundreds digits is 28, plus carryover from tens (4), that's the total sum for the hundreds place, which is 32. **But** the hundreds place of the sum is the digit we get from 32, which is the tens digit of 32, so 2, and the carryover is 3 (from 32 to the thousands place).

[. . . 3,097 tokens omitted . . .]

Wait, no. 32 is the sum of the digits in the hundreds place (plus carryover), which is the value of the hundreds place in the total sum. For instance, if the hundreds place sum is 32, it means that the total sum has a 3 in the hundreds place and 2 in the thousands place? **No**, the place values are:

- If you have 32 as the sum for the hundreds place, it means that you have 3*100 + 2*1, so the hundreds digit is 3, and the tens digit is 2.

[. . . 1,140 tokens omitted . . .]

For example, if the sum of the hundreds digits plus carryover is 32, then the sum is 32*100 + (remainder), so the hundreds digit is 2 (the 32's tens place).

Wait, no: if the sum of the hundreds digits plus carryover is 32, then the total value contributed by the hundreds place is 32*100. **But** in reality, when you add numbers, the hundreds place is the 100s place, so the hundreds digit is the digit in the 100s place of the

[generation cap reached; no final answer]

Example 2. DeepSeek-R1-Distill-Qwen-7B ← DeepCoder-14B (thinking). OPD rollout at training step 128; Semi-OPD rollout from π_0; generation cap 16,384 tokens.

Problem. Find the positive solution to
$\frac{1}{x^2-10x-29} + \frac{1}{x^2-10x-45} - \frac{2}{x^2-10x-69} = 0.$

Semi-OPD (frozen π_0 rollout): 2,228 tokens, 10 reflection words, reaches an answer

[. . . 176 tokens omitted . . .]

Let me let $y = x^2 - 10x$. Then, each denominator becomes $y - 29$, $y - 45$, and $y - 69$. So, substituting that in, the equation becomes:

$\frac{1}{y-29} + \frac{1}{y-45} - \frac{2}{y-69} = 0.$

[. . . 769 tokens omitted . . .]

So, the equation reduces to:

-64y +2496=0.

[. . . 718 tokens omitted . . .]

But the question asks for positive solution, so x=13.

[. . . 438 tokens omitted . . .]

OPD (step 128): 9,016 tokens, 117 reflection words, reaches an answer

[. . . 124 tokens omitted . . .]

Let's set $y = x^2 - 10x$. That seems like a smart substitution because each denominator can be written in terms of y.

[. . . 1,916 tokens omitted . . .]

Wait, perhaps it's a perfect square? Let me check if 257,664 is a square.

Well, 508^2=258,064.

507^2=257,049.

Subtract 257,664 -257,049=615.

Not a perfect square. **Hmm**. . . **Wait**, perhaps 615 can be written as... Not sure.

[. . . 735 tokens omitted . . .]

Wait, perhaps the initial approach is getting me stuck because the square root is messy. Maybe I made a mistake in earlier calculation? Let me go back.

Wait, hold on. Let me recap:

[. . . 139 tokens omitted . . .]

Wait, hold on, perhaps I made a mistake in expanding the denominator?

Wait, (y -29)(y -45)= y^2 -45y -29y +29×45= y^2 -74y +1305?

Wait, 29×45: 20×45=900, 9×45=405, total 1305, that's correct.

Wait, could I make a mistake in cross-multiplying earlier? Let me double-check that.

[. . . 5,172 tokens omitted . . .]

Therefore, okay, the logic is hold.

So steps: substituted y=x^2 -10x, made the equation only in y, solved for y=39, found x=13 and x=-3, recognized which is required: positive.

Hence, the positive solution is 13.

[. . . 587 tokens omitted . . .]

Example 3. Qwen3-1.7B ← Qwen3-30B-A3B-Instruct-2507 (non-thinking). OPD rollout at training step 90; Semi-OPD rollout from π_0; generation cap 4,096 tokens.

Problem. Let $f(x) = \frac{x+1}{x-1}$. Then for $x^2 \neq 1$, find the expression for $f(-x)$. The original answer is in $\frac{k}{m}$ format, please give the value of $k + m$.

Semi-OPD (frozen π_0 rollout): 308 tokens, 0 reflection words, reaches an answer

[. . . 69 tokens omitted . . .]

Step 1: Substitute $-x$ into $f(x)$

$f(-x) = \frac{(-x)+1}{(-x)-1}$

Simplify numerator and denominator:

$f(-x) = \frac{-x+1}{-x-1}$

We can factor out a negative sign from both the numerator and denominator:

$f(-x) = \frac{-(x-1)}{-(x+1)} = \frac{x-1}{x+1}$

Step 2: Identify k and m

We have:

$f(-x) = \frac{x-1}{x+1}$

This is already in the form $\frac{k}{m}$, where:

$k = -1, \quad m = 1$

Step 3: Compute $k + m$

$k + m = -1 + 1 = \boxed{0}$

Final Answer:

$\boxed{0}$

OPD (step 90): 4,096 tokens, 87 reflection words, hits the cap

[. . . 239 tokens omitted . . .]

Step 3: Check if it's in the form $\frac{k}{m}$

This is not a constant, it's a rational function in x, but we are to assume the form is $\frac{k}{m}$, which is a constant. **Wait** — perhaps we misread?

[. . . 156 tokens omitted . . .]

Wait — but $f(-x)$ is not a constant function. Only when x is a constant — but x is a variable.

Unless — maybe the question is a trick?

Wait — let's recheck.

[. . . 151 tokens omitted . . .]

Wait — maybe the question is not asking for $f(-x)$, but rather the expression for $f(-x)$ in a simplified form that becomes a constant?

Wait — but $f(-x) = \frac{x-1}{x+1}$ is not constant.

[. . . 510 tokens omitted . . .]

Wait — **maybe the original question has a typo?**

[. . . 531 tokens omitted . . .]

Wait — press on.

Wait — **let's suppose that the** original answer is $\frac{1}{2}$

[. . . 466 tokens omitted . . .]

Wait — **finally!** Let's suppose that the question is misworded.

[. . . 1,658 tokens omitted . . .]

Wait — perhaps it was supposed to be the **value** of $f(-x)$ at a specific x? **But** no.

Wait — maybe the question is asking for the expression of $f(-x)$ in terms of $f(x)$?

[. . . 100 tokens omitted . . .]

[generation cap reached; no final answer]

D.1 Training Dynamics Across All Pairs

We provide more training dynamics discussion in this section. Figures 13 to 17 track five properties of the training rollouts for 16 of the 17 pairs (all except the Nemotron-Cascade-2 pair).

Rollout length. As shown in Fig. 13, OPD's rollouts lengthen as training proceeds, while Semi-OPD's stay at the length of the initial student's rollouts by construction. Over the last 10 shared steps, OPD's rollouts are 7–113% longer than Semi-OPD's in 15 of 16 pairs. The growth is largest for the low-overlap non-thinking Qwen3 pairs, where OPD's rollouts often exceed the teacher's own length. The only exception is R1-Distill-1.5B ← JustRL-1.5B, whose OPD rollouts get shorter.

Positively supervised tokens. As shown in Fig. 14, Semi-OPD keeps a larger share of positively supervised tokens ($p_T > p_S$) than OPD in 15 of 16 pairs. The exception is again R1-Distill-1.5B ← JustRL-1.5B.

Student and teacher entropy. As shown in Figs. 15 and 16, on thinking pairs OPD drives both the student's and the teacher's next-token entropy on the training tokens well below Semi-OPD's in 7 of 8 pairs (the exception is R1-Distill-7B ← DeepCoder-14B), i.e., OPD increasingly trains on tokens that both models already predict with high confidence. On non-thinking pairs, student entropy stays similar between the two methods, while teacher entropy is usually higher under OPD, indicating that OPD's rollouts drift toward regions where the teacher is less certain.

Teacher–student overlap. As shown in Fig. 17, the top-64 overlap between the current student and the teacher on the training tokens generally rises under both methods as the student moves toward the teacher, but the ordering differs across different model pairs.

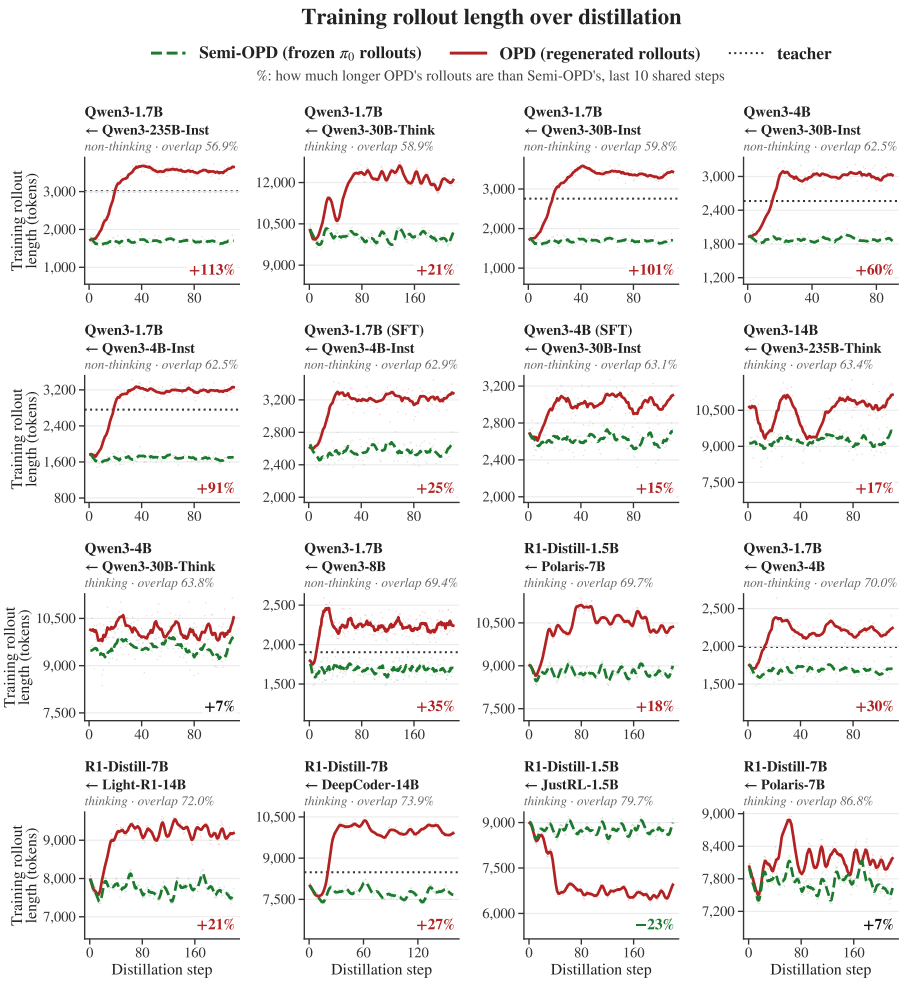


Figure 13: **Training rollout length.** Mean generated tokens per training rollout. The number in each panel is how much longer OPD's rollouts are than Semi-OPD's over the last 10 shared steps; the dotted line is the teacher's length, where available.

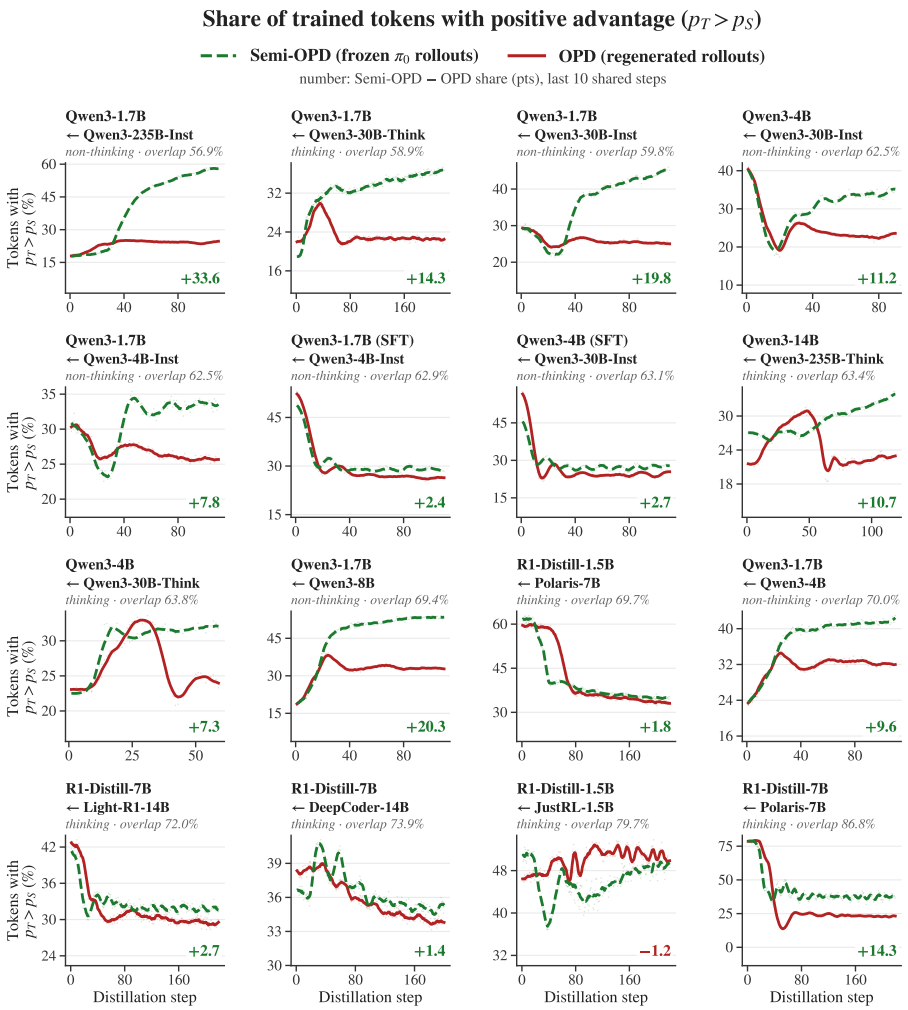


Figure 14: **Positively supervised tokens.** Share of training tokens with $p_T > p_S$. The number in each panel is Semi-OPD minus OPD, in points, over the last 10 shared steps.

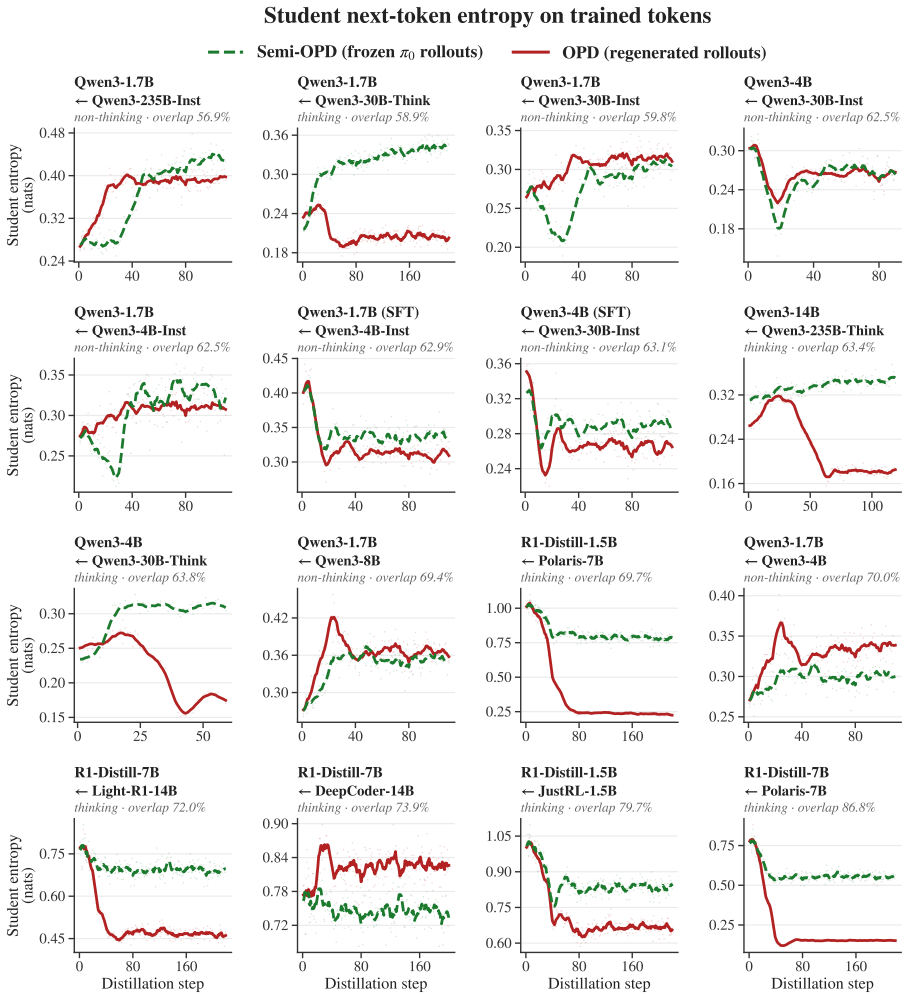


Figure 15: **Student entropy.** Mean next-token entropy of the student (nats) on the training tokens.

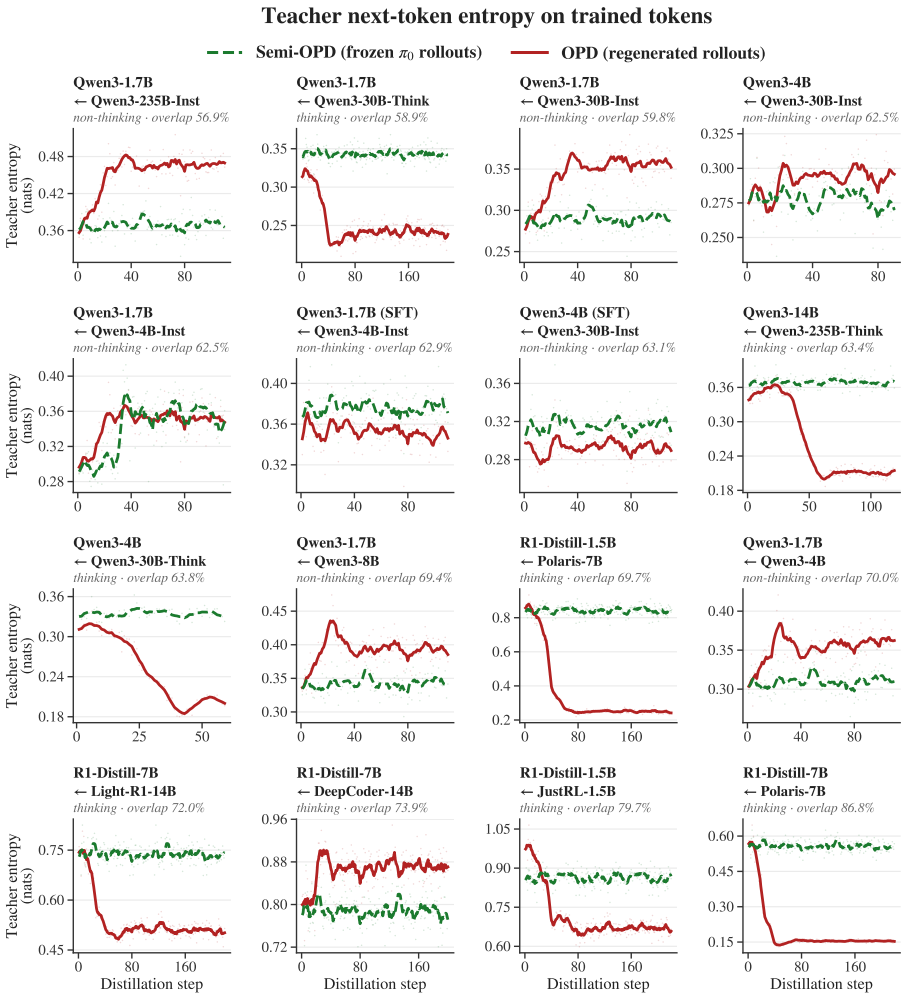


Figure 16: **Teacher entropy.** Mean next-token entropy of the teacher (nats) on the same training tokens, i.e. how uncertain the teacher is on the rollouts it supervises.

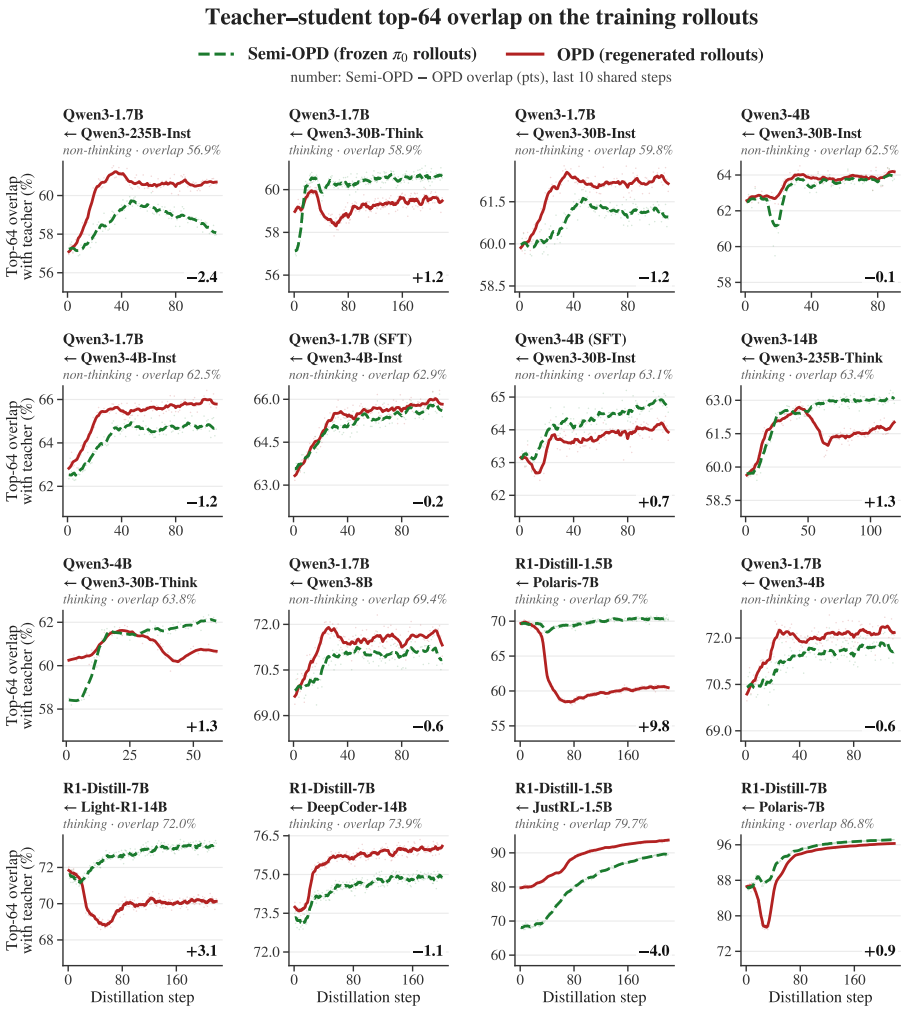


Figure 17: **Teacher–student overlap on the training rollouts.** Top-64 overlap between the current student and the teacher, computed on each method's own training tokens with the same metric as the initial overlap in Section 2. The number in each panel is Semi-OPD minus OPD, in points, over the last 10 shared steps. For the two SFT-initialized pairs, overlap was measured only at initialization and at step 110.

E SWITCHING FROM SEMI-OPD TO OPD

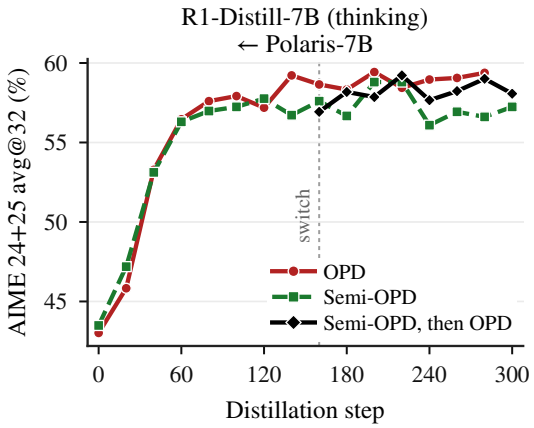


Figure 18: Switching from Semi-OPD to OPD on a high-overlap pair (R1-Distill-7B student, Polaris-7B teacher, 86.8% initial top-64 overlap). Starting from the step-160 Semi-OPD checkpoint, continuing with online OPD recovers most of the gap to OPD from scratch. Continuing with Semi-OPD plateaus. In contrast, on low-overlap pairs, switching from Semi-OPD to OPD drops performance to the level of OPD trained from scratch (Fig. 9).