BOOTSTRAPPED ON-POLICY DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy distillation (OPD) improves language models by distilling teacher predictions on student-generated trajectories, while on-policy self-distillation (OPSD) removes the need for an external teacher by using a reference-conditioned copy of the model as the teacher. Despite its great early improvements, we find that OPSD does not remain consistently effective under continued training and can eventually suffer substantial performance degradation. We study this training behavior and link the degradation to a growing mismatch between the evolving reference-free student and the frozen privileged teacher, together with increasing student uncertainty on its evolving on-policy states. This growing mismatch motivates us to first learn from privileged reference supervision and then transfer the acquired knowledge to a fresh student. We introduce Bootstrapped On-Policy Distillation (BOPD), a two-stage framework that first uses OPSD to internalize task-relevant information from reference-conditioned supervision into an intermediate model, and then distills this reference-free model into a fresh copy of the initial model. Experiments across 4B and 8B models show that BOPD improves mathematical reasoning beyond both the initial models and the constructed OPSD teachers, while also outperforming the constructed teachers on code generation. These results suggest that privileged reference supervision can be more effectively exploited by first transferring its task-relevant signal into a task-adapted model and then distilling from that model without reference conditioning.

1 INTRODUCTION

On-policy distillation (OPD) has emerged as an effective approach to language model post-training (Agarwal et al., 2024; Gu et al., 2024). Rather than learning from fixed demonstrations, the student model generates its own trajectories and receives dense token-level supervision from a teacher model, enabling it to learn directly from the teacher on its on-policy state distribution (Agarwal et al., 2024; Zhang et al., 2026b; Yang et al., 2025; Lu, 2025). This on-policy supervision reduces the mismatch between training-time and inference-time state distributions while providing denser token-level feedback than sparse outcome-level rewards, thereby enabling substantial capability improvements during post-training.

Recent work has sought to broaden the applicability of OPD by removing the need for an external teacher model. On-policy self-distillation (OPSD) (Zhao et al., 2026a) achieves this by using the model itself as the teacher: the student generates trajectories conditioned only on the problem, while a frozen copy of the same model additionally conditions on the reference solution to provide token-level supervision. Building on this formulation, β-OPSD (Xu et al., 2026) derives OPSD from a KL-regularized reinforcement learning objective and introduces a tunable interpolation between the reference policy and the privileged teacher as the target distribution, while DualOPSD (Chen et al., 2026) jointly optimizes the teacher and student to adapt the supervision throughout training.

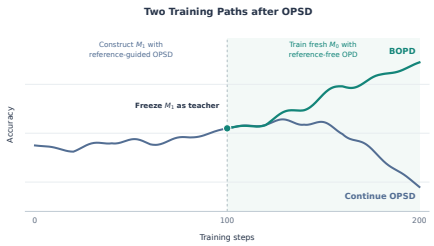

Figure 1: Performance of OPSD and BOPD during the training process

Despite these advances, we observe a striking phenomenon under prolonged OPSD training: OPSD improves model performance during the early stage of training, but these gains progressively di-

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.

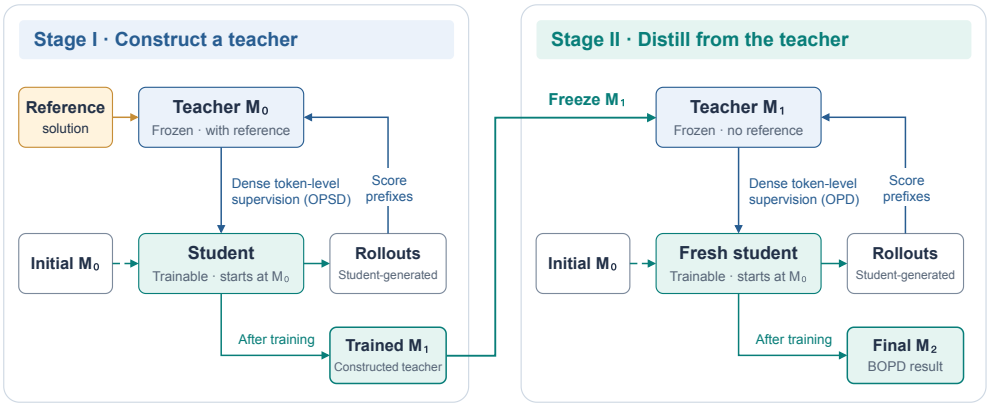

Figure 2: BOPD overview. In Stage I, a frozen copy of M_0 receives a reference solution and supervises a student on its own rollouts, producing M_1. In Stage II, frozen M_1 supervises a fresh student initialized from M_0; neither Stage II model receives a reference. Dashed arrows denote student initialization; solid arrows show rollout flow, supervision, and model transfer.

minish and can eventually reverse as optimization continues (Figure 1). During the early stage of training, the privileged reference provides useful corrective supervision, but as the student policy evolves, the teacher's predictions may increasingly depend on information that is unavailable to the reference-free student, leading to a growing teacher–student mismatch (Zhao et al., 2026b; Zhang et al., 2026a). This suggests a fundamental limitation of OPSD: although privileged reference information makes the teacher more informative, it can also make the resulting supervision increasingly incompatible with the reference-free student as training proceeds. In Section 3, we provide empirical analyses supporting this interpretation and, following the principles in Li et al. (2026), we show that the privilege-conditioned model does not constitute a well-matched teacher for the evolving student. These observations raise a broader question: can a student construct a capable and well-matched teacher for itself without relying on an external teacher model?

In this paper, we introduce Bootstrapped On-Policy Distillation (BOPD), a two-stage framework that first constructs a teacher by updating the initial model with OPSD, and then distills the resulting teacher into a fresh copy of the initial model without privileged reference conditioning. BOPD therefore separates teacher construction from teacher use: the first stage exploits privileged reference information to internalize task-relevant reasoning signals into the model, while the second stage reuses the resulting model as a reference-free teacher for standard on-policy distillation. Through Stage I, π_{B} absorbs part of the reasoning signal provided by the privileged reference into its own policy. When used in Stage II, π_{B} no longer requires access to the reference solution, so the teacher and student are conditioned on the same observable information. This design preserves the task-specific knowledge acquired through OPSD while reducing the privileged-context mismatch that arises when the teacher relies on information unavailable to the student.

Formally, the BOPD procedure can be summarized as

$$\pi_0 \xrightarrow[\text{reference available}]{\text{OPSD}} \pi_{\mathrm{B}} \quad \text{and then} \quad \pi_0 \xrightarrow[\text{reference unavailable}]{\text{OPD with teacher } \pi_{\mathrm{B}}} \pi_{\mathrm{final}}, \tag{1}$$

where π_{B} denotes the bootstrapped teacher. Our work makes three contributions:

- We analyze the training dynamics of OPSD and provide evidence that its early improvement followed by prolonged degradation is associated with a growing incompatibility between the privileged reference-conditioned teacher and the reference-free student.
- We introduce Bootstrapped On-Policy Distillation (BOPD), a two-stage framework that first uses OPSD to construct a task-adapted teacher from the initial model, and then distills this teacher into a fresh copy of the initial model without privileged reference conditioning.
- We validate BOPD across 4B and 8B models through extensive experiments and ablations. BOPD improves over both the initial model and the constructed OPSD teacher, while anal-

yses of prolonged OPSD training, teacher construction, and reference access provide empirical support for the proposed framework.

2 Related Work

Learning from Reference Solutions. SFT learns from reference solutions by fitting their target tokens. Recent work improves this learning signal through token weighting or selection: DFT revisits SFT from a reinforcement-learning perspective, while PriFT and ProFit use model probabilities to guide which supervision to emphasize (Wu et al., 2025; Wang et al., 2026; Liu et al., 2026). A complementary approach adapts the training data itself. Mind the Gap retains correct model-generated solutions and uses guided re-solving to replace incorrect ones before fine-tuning (Zhao et al., 2025). Hinted Decoding similarly constructs responses that better match the model's distribution (Zhang et al., 2026b). These methods use references to improve the targets or learning signal for supervised training. Reference data can also support teacher construction. PromptSD trains task-specific soft prompts on labeled examples with a frozen backbone, then uses the resulting teachers to supervise student-generated trajectories (Ma et al., 2026). BOPD shares the goal of turning reference data into useful teacher supervision, but constructs its teacher by updating model parameters through OPSD.

On-Policy (Self-)Distillation. Knowledge distillation transfers a teacher's predictive distribution to a student (Hinton et al., 2015). GKD and MiniLLM apply distributional supervision to student-generated trajectories, with different choices of distillation objective (Agarwal et al., 2024; Gu et al., 2024). SKD adjusts trajectory generation through teacher–student cooperation, while analyses of OPD highlight the importance of teacher capability and compatibility with student-visited states (Xu et al., 2025; Li et al., 2026). OPSD removes the need for a separate teacher by conditioning a self-teacher on reference solutions; SDFT uses demonstration-conditioned supervision for continual learning (Zhao et al., 2026a; Shenfeld et al., 2026). Subsequent work examines how privileged context affects supervision: DOPD routes token-level signals between privileged teacher and student policies, and Rethinking OPSD identifies failure modes of reference conditioning in thinking models (Yu et al., 2026; Kaur et al., 2026). LOPD makes privileged context learnable by composing retrieved experiences into latent tokens and jointly optimizing the context composer and student (Zhang et al., 2026a). These approaches motivate studying how to obtain effective self-supervision. Our focus is to use OPSD as a distinct teacher-construction stage, followed by OPD with a frozen teacher, no reference context, and a student reset to the initial model. This reset also connects BOPD to born-again distillation, which trains a new student from a teacher of the same architecture (Furlanello et al., 2018).

3 What Happens When OPSD Continues?

In this section, we study the training dynamics of OPSD beyond its initial improvement phase. OPSD uses a frozen, privilege-conditioned initial model as the teacher to provide guidance on student-generated trajectories. As training proceeds, the teacher remains fixed while the student policy and the distribution of student-generated prefixes continue to evolve, causing the teacher to be queried on increasingly different states. This raises three questions: whether the effectiveness of OPSD is sustained as training continues, whether the privileged teacher remains well matched to the evolving student, and how the student's on-policy state distribution changes over prolonged training.

To study the behavior of continued OPSD training, we train Qwen3-4B (Yang et al., 2025) with OPSD for 200 optimization steps and evaluate intermediate checkpoints throughout training. We measure task performance on AIME 2024, AIME 2025, and HMMT 2025, and analyze teacher–student distributions on 64 training prompts. For each checkpoint, we track teacher–student agreement using Overlap@16, overlap-token advantage@16, the absolute entropy gap, and the full-vocabulary KL divergence. We additionally compare student entropy on on-policy prefixes with entropy measured on a fixed set of prefixes generated by the initial model, allowing us to distinguish changes in the student distribution from changes in the states visited by the student during training. Details of these measurements are provided in Appendix E.

Continued OPSD does not remain consistently effective. As shown in Figure 3, task accuracy improves during the early and middle stages of training, but the gains eventually diminish and re-

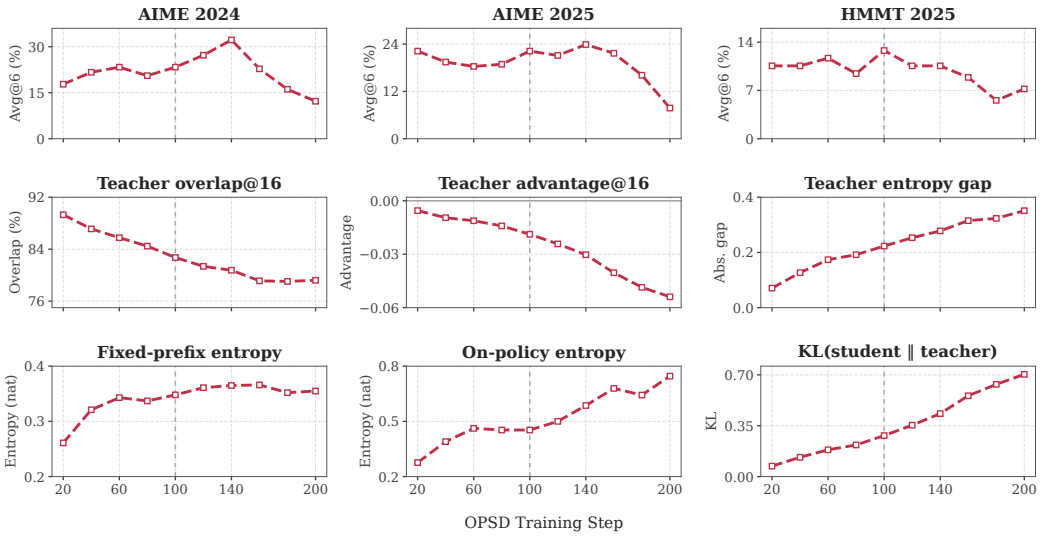

Figure 3: Continued OPSD training on Qwen3-4B. Top: Avg@6 on three mathematics benchmarks. Middle: overlap@16, overlap-token advantage@16, and absolute entropy gap between the OPSD student and its privileged teacher. Bottom: Fixed-prefix entropy, on-policy entropy and KL divergence between the student and teacher.

verse. The mean Avg@6 accuracy across the three mathematical benchmarks reaches 22.22% at step 140, but falls to 9.07% by step 200. At the same time, the student becomes increasingly mismatched with the privileged teacher. Teacher–student Overlap@16 decreases from 89.30% at step 20 to 79.21% at step 200, while the full-vocabulary KL divergence increases from 0.0719 to 0.7038. The overlap-token advantage also moves farther from zero, and the absolute entropy gap increases over training. Together, these trends show that the loss of effectiveness under continued OPSD is accompanied by growing distributional mismatch between the reference-free student and the frozen reference-conditioned teacher.

To analyze whether the observed entropy increase comes from changes in the student itself or from the states it visits, we compare student entropy on on-policy prefixes with entropy on prefixes that are generated by the initial model. Entropy on the fixed prefixes quickly saturates, whereas entropy on the on-policy prefixes continues to increase, suggesting that the growing uncertainty is closely tied to the evolving states visited by the student.

Taken together, these results suggest that continued OPSD progressively shifts the student-generated prefix distribution while the privileged teacher remains fixed, leading the teacher to provide supervision on states where its predictions are increasingly mismatched with those of the evolving student. The observed degradation can be characterized as a growing mismatch between the privileged teacher and the reference-free student, which reduces the suitability of the teacher's supervision as training progresses. This motivates separating teacher construction from teacher use: rather than continuing OPSD indefinitely, we may stop at an intermediate checkpoint that has already internalized part of the task-relevant signal provided by the reference-conditioned teacher and can therefore serve as a task-adapted, reference-free teacher for a fresh student.

4 Bootstrapped On-Policy Distillation

4.1 Problem Setup and Overview

We consider a dataset $\mathcal{D} = \{(x_i, y_i^\star)\}_{i=1}^N$, where x_i is a problem and $y_i^\star$ is a reference solution. Let $\mathcal{D}_x$ denote the empirical prompt distribution, $\mathcal{V}$ the token vocabulary, and $\pi_0 = \pi_{\theta_0}$ the initial language model. Our goal is to improve π_0 using $\mathcal{D}$ without relying on an external teacher model.

BOPD consists of two stages (Equation 1). In Stage I, we use OPSD to transform the initial model into a task-adapted checkpoint. In Stage II, we freeze this checkpoint as a reference-free teacher and distill it into a fresh copy of the initial model. Both stages operate on student-generated trajectories. Algorithm 1 summarizes the full procedure.

4.2 Stage I: Bootstrapping a Teacher with OPSD

We initialize a trainable student π_θ and a frozen teacher π_{θ_0} from the same initial model π_0. For each $(x, y^\star) \sim \mathcal{D}$, the student generates $\hat{y} \sim \pi_\theta(\cdot \mid x)$. At each student-generated prefix $\hat{y}_{<t}$, the student and privileged teacher define

$$p_t(v) = \pi_\theta(v \mid x, \hat{y}_{<t}), \tag{2}$$

$$q_t^{\mathrm{ref}}(v) = \pi_{\theta_0}(v \mid x, y^\star, \hat{y}_{<t}), \qquad v \in \mathcal{V}. \tag{3}$$

We update only the student parameters by minimizing the trajectory-averaged forward KL divergence:

$$\mathcal{L}_{\mathrm{OPSD}}(\theta) = \mathbb{E}_{(x,y^\star)\sim\mathcal{D},\, \hat{y}\sim\pi_\theta(\cdot|x)} \left[\frac{1}{|\hat{y}|} \sum_{t=1}^{|\hat{y}|} D_{\mathrm{KL}}\big(q_t^{\mathrm{ref}} \big\| p_t\big) \right]. \tag{4}$$

After Stage I, we save the resulting checkpoint as the bootstrapped teacher $\pi_{\mathrm{B}} = \pi_{\theta_{\mathrm{B}}}$. This stage converts reference-conditioned supervision into a task-adapted model which can subsequently be used without access to the reference solution.

4.3 Stage II: On-Policy Distillation with the Bootstrapped Teacher

We freeze the bootstrapped teacher π_{B} and reinitialize the student from the exact base checkpoint, $\phi \leftarrow \theta_0$. The new student generates $\hat{y} \sim \pi_\phi(\cdot \mid x)$. At each prefix $\hat{y}_{<t}$, the student and teacher condition on the same observable context:

$$p_t(v) = \pi_\phi(v \mid x, \hat{y}_{<t}), \tag{5}$$

$$q_t^{\mathrm{B}}(v) = \pi_{\mathrm{B}}(v \mid x, \hat{y}_{<t}). \tag{6}$$

Unlike Stage I, the teacher no longer receives the reference solution and we minimize

$$\mathcal{L}_{\mathrm{BOPD}}(\phi) = \mathbb{E}_{x\sim\mathcal{D}_x,\, \hat{y}\sim\pi_\phi(\cdot|x)} \left[\frac{1}{|\hat{y}|} \sum_{t=1}^{|\hat{y}|} D_{\mathrm{KL}}\big(q_t^{\mathrm{B}} \big\| p_t\big) \right]. \tag{7}$$

Resetting the student separates teacher construction from knowledge transfer: Stage I produces a task-adapted teacher using privileged supervision, while Stage II transfers the resulting behavior to a fresh reference-free student on its own on-policy trajectories. The final model has the same architecture and inference cost as the initial model.

5 Experiments

5.1 Experimental Setup

Training data and models. We use Qwen3-4B and Qwen3-8B in non-thinking mode (Yang et al., 2025). Mathematical reasoning experiments use the OpenThoughts OPSD split (Guha et al., 2026). Code-generation experiments use problems from the TACO subset of the DeepCoder Preview Dataset (Li et al., 2023; Luo et al., 2025). Within each domain, OPSD and BOPD use the same training data. We train OPSD and each BOPD stage for 100 optimizer steps with an effective batch size of 32. We use LoRA (Hu et al., 2021) with rank 64 and scaling parameter 128 on the query, key, value, and output attention projections and the gate, up, and down MLP projections; code runs use 0.05 LoRA dropout. The learning rate is 5×10^{-6}, with gradient clipping at 0.1 and pointwise loss clipping at $\tau = 0.05$. Student rollouts contain at most 1,024 tokens, sampled with temperature 1.1, top-p 0.95, and top-k 20. We use bfloat16 and DeepSpeed ZeRO-2. Appendix A provides additional settings.

Algorithm 1 Bootstrapped On-Policy Distillation (BOPD)

Require: Initial parameters θ_0, reference data $\mathcal{D}$, stage budgets K_1, K_2

Stage I: Construct a teacher with reference-conditioned OPSD

Initialize student $\theta \leftarrow \theta_0$ and freeze teacher π_{θ_0}
for $k = 1, \ldots, K_1$ **do**
  Sample a minibatch of $(x, y^\star)$ from $\mathcal{D}$
  Generate $\hat{y} \sim \pi_\theta(\cdot \mid x)$
  Compute $q_t^{\text{ref}} = \pi_{\theta_0}(\cdot \mid x, y^\star, \hat{y}_{<t})$
  Update θ by minimizing Eq. 4
end for
Set $\theta_{\text{B}} \leftarrow \theta$ and freeze $\pi_{\text{B}} = \pi_{\theta_{\text{B}}}$

Stage II: Distill from the reference-free bootstrapped teacher

Reset student $\phi \leftarrow \theta_0$ and initialize a fresh optimizer
for $k = 1, \ldots, K_2$ **do**
  Sample a minibatch of prompts x from $\mathcal{D}_x$
  Generate $\hat{y} \sim \pi_\phi(\cdot \mid x)$
  Compute $q_t^{\text{B}} = \pi_{\text{B}}(\cdot \mid x, \hat{y}_{<t})$
  Update ϕ by minimizing Eq. 7
end for
return π_ϕ

Table 1: Mathematical reasoning results (%). Bold and underlining mark the best and second-best distinct scores, respectively, within each model size and metric.

		Avg@12 ↑				Pass@12 ↑			
Size	Method	AIME24	AIME25	HMMT25	Mean	AIME24	AIME25	HMMT25	Mean
4B	Initial model	22.50	19.72	<u>11.94</u>	18.06	56.67	40.00	<u>30.00</u>	42.22
	OPSD (Stage I)	<u>25.83</u>	<u>20.00</u>	**13.06**	<u>19.63</u>	<u>60.00</u>	<u>46.67</u>	**33.33**	<u>46.67</u>
	BOPD (ours)	**34.17**	**26.39**	11.39	**23.98**	**66.67**	**63.33**	**33.33**	**54.44**
8B	Initial model	25.56	19.44	<u>11.11</u>	18.70	43.33	33.33	<u>23.33</u>	33.33
	OPSD (Stage I)	<u>26.67</u>	<u>21.67</u>	**12.22**	<u>20.19</u>	<u>46.67</u>	<u>36.67</u>	**26.67**	<u>36.67</u>
	BOPD (ours)	**31.11**	**23.89**	<u>11.11</u>	**22.04**	**56.67**	**40.00**	<u>23.33</u>	**40.00**

Evaluation. We evaluate three competition mathematics benchmarks with 30 problems each: AIME 2024 and AIME 2025 (Mathematical Association of America, n.d.), and HMMT February 2025 (Harvard-MIT Mathematics Tournament, 2025). We use temperature 1.0, top-p 0.8, and a 38,912-token completion limit; top-k and min-p filtering are disabled. We report average response accuracy (Avg@K) and the fraction of problems solved at least once (Pass@K), using $K = 12$ for the main results and $K = 6$ for the ablation studies. Mean accuracy is the unweighted average across the three benchmarks. We evaluate code-trained models on HumanEval+ and MBPP+ (Liu et al., 2023) and the new problems in LiveCodeBench v5 and v6 (Jain et al., 2025). These code evaluations use non-thinking mode, greedy decoding, Pass@1, and a 4,096-token generation limit. To assess capability retention after mathematical training, we evaluate HumanEval+ Pass@1, prompt-level strict accuracy on IFEval (Zhou et al., 2023), and five-shot accuracy on MMLU (Hendrycks et al., 2021). Appendices A.1 and B provide further evaluation details.

5.2 Main Results

Mathematical reasoning. BOPD outperforms both the initial model and its constructed teacher at 4B and 8B (Table 1). At 4B, average accuracy increases from 18.06% to 23.98%, surpassing the OPSD teacher's 19.63%. At 8B, it rises from 18.70% to 22.04%, compared with 20.19% for the teacher. Pass@12 also improves at both sizes. These results show that a model can construct a teacher from reference data and use it to train a student that surpasses the teacher itself.

Code generation. Table 2 reports results on coding benchmarks for models trained on code. At 4B, BOPD achieves 46.54% mean Pass@1, compared with 44.42% for the initial model and 45.53%

Table 2: Code-generation results. Mean equally averages the four benchmarks before rounding. Bold and underlining mark the best and second-best distinct scores within each model size and metric.

Size	Method	HumanEval+	MBPP+	LCB v5-new	LCB v6-new	Mean
4B	Initial model	<u>74.39</u>	57.14	<u>21.56</u>	**24.57**	44.42
	OPSD (Stage I)	**75.61**	<u>60.32</u>	**22.75**	<u>23.43</u>	<u>45.53</u>
	BOPD (ours)	**75.61**	**63.23**	**22.75**	**24.57**	**46.54**
8B	Initial model	**81.71**	**65.87**	**22.75**	24.57	<u>48.73</u>
	OPSD (Stage I)	<u>79.27</u>	<u>62.96</u>	<u>22.16</u>	<u>25.14</u>	47.38
	BOPD (ours)	<u>79.27</u>	**65.87**	<u>22.16</u>	**28.00**	**48.82**

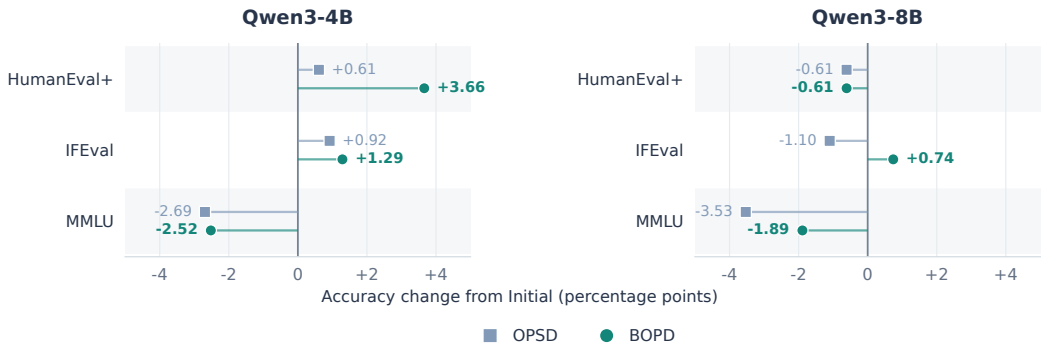

Figure 4: Capability retention after mathematical training. Points show accuracy changes relative to the initial model (zero line), in percentage points.

for OPSD. At 8B, BOPD reaches 48.82%, compared with 48.73% for the initial model and 47.38% for OPSD. BOPD outperforms its OPSD teacher in mean Pass@1 at both sizes.

Capability retention. Figure 4 evaluates capabilities outside mathematics after mathematical training. On HumanEval+, BOPD scores 78.05% versus 74.39% for the initial model at 4B, and 81.10% versus 81.71% at 8B. It raises IFEval prompt-level strict accuracy by 1.29 and 0.74 percentage points at 4B and 8B, respectively. On MMLU, BOPD improves over OPSD at both sizes; at 8B, it narrows the gap according to initial model from 3.53 to 1.89 percentage points. These results show BOPD could retain more out domain ability. Appendix B provides further evaluation details.

5.3 Mechanism Analysis

Prior analyses of on-policy distillation suggest that an effective teacher should provide useful supervision while remaining sufficiently compatible with the student on the states visited during training. We therefore examine whether the bootstrapped teacher in BOPD exhibits a more favorable teacher–student relationship than the privileged teacher in OPSD. Following Li et al. (2026), we track checkpoint accuracy together with Overlap@16, overlap-token advantage@16, and the absolute entropy gap during 4B training (Figure 5). The distributional metrics are computed on the same 64 fixed training prompts used in Section 3, with each checkpoint generating its own on-policy trajectories.

Figure 5 shows a clear difference in candidate-set agreement. Under continued OPSD, Overlap@16 decreases from 0.8930 at step 20 to 0.8272 at step 100, whereas under BOPD it increases from 0.8638 to 0.9005. Thus, the BOPD student increasingly shares its teacher's likely top-16 token candidates, while the OPSD student progressively moves away from those of its privileged teacher. This trend accompanies stronger BOPD checkpoint accuracy on AIME24 and AIME25 after step 40.

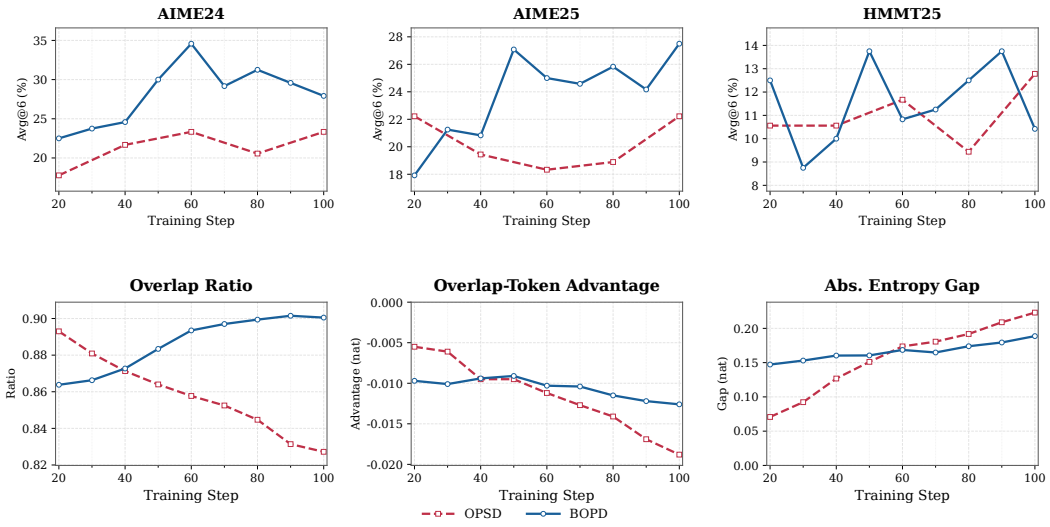

Figure 5: Accuracy and teacher–student alignment during 4B mathematical training. The top row shows checkpoint accuracy on each benchmark; the OPSD scores are also shown in Figure 3. The bottom row reports alignment metrics on regenerated trajectories for the same 64 fixed training prompts. Curves begin at step 20; steps are relative to each method's training stage.

The other metrics show a similar contrast. Under BOPD, the overlap-token advantage remains relatively close to zero with modest fluctuations, reaching -0.0126 at step 100. Under OPSD, by contrast, it moves steadily away from zero, from -0.0055 to -0.0188. A similar pattern appears in the entropy gap: BOPD shows only a moderate increase from 0.1472 to 0.1887 nats, whereas OPSD starts with a much smaller gap of 0.0707 nats but increases substantially to 0.2230 nats. Together with the opposing Overlap@16 trends, these results indicate that the BOPD teacher remains better matched to the evolving student, whereas the privileged OPSD teacher becomes increasingly mismatched as training proceeds. This growing mismatch may limit the quality of supervision provided by the OPSD teacher at later stages of training.

Taken together, these dynamics are consistent with the bootstrapped model serving as a better-matched teacher. They support the role of teacher bootstrapping in converting reference-conditioned supervision into a more compatible reference-free teacher.

5.4 Ablation Studies

We ask how to construct a suitable teacher for OPD and whether BOPD makes better use of its training steps than continued OPSD (Figure 6). With the same 200-step budget, BOPD's 100 OPSD steps followed by 100 OPD steps outperform 200 steps of OPSD at both model sizes. We also construct a teacher with solution SFT instead of OPSD. Its accuracy is below the initial model at both sizes. Subsequent OPD improves on this SFT teacher, but remains below the initial model and BOPD. Finally, we give the OPSD-trained teacher the reference solution again during Stage II. Compared with reference-free BOPD, this reduces average accuracy from 23.15% to 12.22% at 4B and from 22.04% to 16.48% at 8B. The trained teacher does not need reference access to guide Stage II in this setting. One possible explanation is that the added reference changes its targets in ways the reference-free student cannot readily follow. Appendix D reports detailed scores and control settings.

Further distillation. We ask whether the BOPD student M_2 can support further self-improvement (Table 3). First, we freeze M_2 as a teacher and distill it into a fresh M_0 student. The resulting M_3 reaches 19.44% average accuracy, below M_2's 23.15%: a stronger student does not automatically transfer its gains when used as a teacher. Next, we test whether M_2 itself can improve through another OPSD stage. We initialize the student and frozen, reference-conditioned teacher from M_2, obtaining M_4; its accuracy instead falls to 3.33%. We then complete a second BOPD cycle from

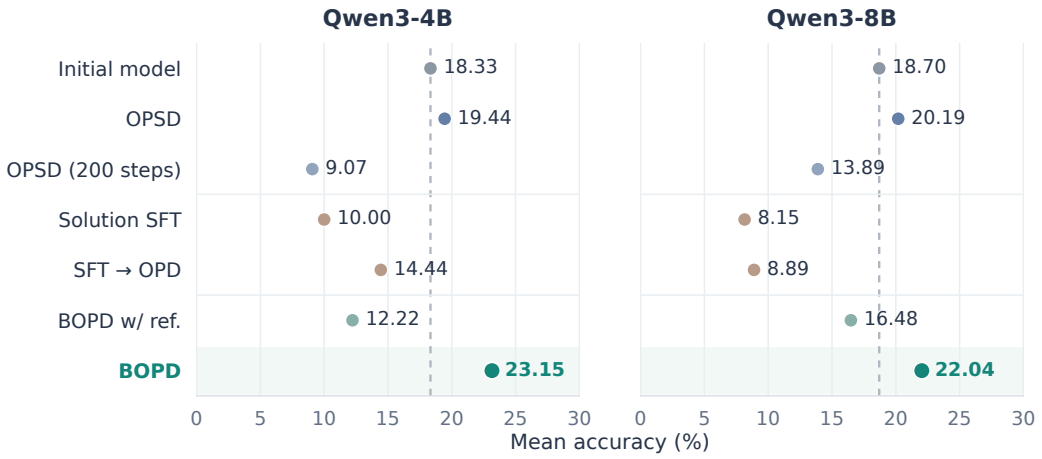

Figure 6: Mean mathematical accuracy across AIME24, AIME25, and HMMT25. Each gray dashed line marks the corresponding initial-model accuracy.

Table 3: Further distillation at 4B. Each trained model in the first column is obtained by initializing from the model in the Student column and learning from the model in the Teacher column. "+ ref." means the teacher also sees the reference solution.

Model	Student	Teacher	AIME24	AIME25	HMMT25	Avg@6	Pass@6
Initial M_0	—	—	24.44	21.11	9.44	18.33	33.33
OPSD M_1	M_0	M_0 + ref.	25.00	19.44	13.89	19.44	38.89
BOPD M_2	M_0	M_1	32.78	25.56	11.11	23.15	44.44
M_3	M_0	M_2	29.44	20.00	8.89	19.44	37.78
M_4	M_2	M_2 + ref.	3.89	4.44	1.67	3.33	15.56
M_5	M_2	M_4	8.89	6.11	3.33	6.11	21.11
M_6	M_0	M_4	8.89	5.00	1.67	5.19	17.78

M_2 by using M_4 as a reference-free teacher for a new M_2 student. This yields M_5 with 6.11% accuracy. Finally, we test whether M_4 can teach a fresh M_0 student instead, obtaining M_6 with 5.19%. Neither student taught by M_4 recovers M_2's performance. Appendix F reports the protocol and detailed scores.

6 Conclusion

We study the training dynamics of on-policy self-distillation and find that its early gains are not sustained under continued training. As the student evolves, the privileged reference-conditioned teacher becomes increasingly mismatched with the reference-free student on its on-policy states. Motivated by this observation, we introduce Bootstrapped On-Policy Distillation (BOPD), which uses OPSD to construct a task-adapted intermediate model and then reuses this model as a reference-free teacher for a fresh student.

Across 4B and 8B models, BOPD improves mathematical reasoning beyond both the initial model and the constructed OPSD teacher, and also outperforms the constructed teacher on code generation. Ablations show that simply extending OPSD training, constructing the teacher with solution SFT, or restoring reference access during the second stage does not reproduce these gains under our setting. Training-dynamics analyses further show that BOPD maintains a more favorable teacher–student relationship than continued OPSD. At the same time, our further-distillation experiments show that these gains do not compound indefinitely. Overall, our results suggest that OPSD can be used effectively as a teacher-construction mechanism, while separating privileged teacher construction from subsequent reference-free distillation provides a practical route to self-improvement without an external teacher model.

AI USE STATEMENT

Generative AI tools assisted with refining the paper narrative and and drafting code for analysis and figures. The authors reviewed the AI-assisted outputs, independently checked all reported experimental results, and take responsibility for the paper's content, claims, and artifacts.

REFERENCES

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

Yutong Chen, Guangfu Guo, Zhichao Xu, and Kunpeng Liu. DualOPSD: Adaptive privileged teachers for on-policy self-distillation. *arXiv preprint arXiv:2608.26019*, 2026. URL https://arxiv.org/abs/2608.26019.

Tommaso Furlanello, Zachary C. Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born again neural networks. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1607–1616, 2018.

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

Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, Ashima Suvarna, Benjamin Feuer, Leon Liangyu Chen, Zaid Khan, Eric Frankel, et al. OpenThoughts: Data recipes for reasoning models. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2506.04178.

Harvard-MIT Mathematics Tournament. February 2025 tournament problem archive, 2025. URL https://www.hmmt.org/www/archive/282.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://arxiv.org/abs/2009.03300.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021. URL https://arxiv.org/abs/2106.09685.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2403.07974.

Simran Kaur, Narutatsu Ri, Yinghui He, Liam Fowl, and Sanjeev Arora. Rethinking on-policy self-distillation for thinking models. *arXiv preprint arXiv:2607.05184*, 2026. URL https://arxiv.org/abs/2607.05184.

Rongao Li, Jie Fu, Bo-Wen Zhang, Tao Huang, Zhihong Sun, Chen Lyu, Guang Liu, Zhi Jin, and Ge Li. TACO: Topics in algorithmic COde generation dataset. *arXiv preprint arXiv:2312.14852*, 2023. URL https://arxiv.org/abs/2312.14852.

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. *arXiv preprint arXiv:2604.13016*, 2026. URL https://arxiv.org/abs/2604.13016.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. Is your code generated by ChatGPT really correct? rigorous evaluation of large language models for code generation. In *Advances in Neural Information Processing Systems*, 2023. URL https://arxiv.org/abs/2305.01210.

Tao Liu, Taiqiang Wu, Runming Yang, Shaoning Sun, Junjie Wang, and Yujiu Yang. ProFit: Leveraging high-value signals in SFT via probability-guided token selection. *arXiv preprint arXiv:2601.09195*, 2026. URL https://arxiv.org/abs/2601.09195.

Kevin Lu. On-policy distillation. Thinking Machines Lab blog, October 2025. URL https://thinkingmachines.ai/blog/on-policy-distillation/.

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. Agentica Project and Together AI blog, 2025. URL https://www.together.ai/blog/deepcoder.

Yingzi Ma, Zichen Zhu, Ming Jiang, and Chaowei Xiao. One student, many teachers: Multi-task on-policy distillation via soft-prompt privileged context. *arXiv preprint arXiv:2607.18293*, 2026. URL https://arxiv.org/abs/2607.18293.

Mathematical Association of America. American invitational mathematics examination. MAA Invitational Competitions, n.d. URL https://maa.org/maa-invitational-competitions/.

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning. *arXiv preprint arXiv:2601.19897*, 2026. URL https://arxiv.org/abs/2601.19897.

Ke Wang, Shuangqi Li, Mathieu Salzmann, and Pascal Frossard. PriFT: Prior-support guided supervised fine-tuning. *arXiv preprint arXiv:2606.09396*, 2026. URL https://arxiv.org/abs/2606.09396.

Yongliang Wu, Yizhou Zhou, Zhou Ziheng, Yingzhe Peng, Xinyu Ye, Xinting Hu, Wenbo Zhu, Lu Qi, Ming-Hsuan Yang, and Xu Yang. On the generalization of SFT: A reinforcement learning perspective with reward rectification. *arXiv preprint arXiv:2508.05629*, 2025. URL https://arxiv.org/abs/2508.05629.

Jiawei Xu, Minghui Liu, Juzheng Zhang, Tom Goldstein, and Furong Huang. β-OPSD: Deriving with policy optimization, training with self-distillation. *arXiv preprint arXiv:2607.28582*, 2026. URL https://arxiv.org/abs/2607.28582.

Wenda Xu, Rujun Han, Zifeng Wang, Long T. 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 *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2410.11325.

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. URL https://arxiv.org/abs/2505.09388.

Xinlei Yu, Gen Li, Qingyi Si, Guibin Zhang, Yuqi Xu, Congcong Wang, Shuai Dong, Kaiwen Tuo, Xiangyu Zeng, Kaituo Feng, Qunzhong Wang, Yang Shi, Xiaobin Hu, Xiangyu Yue, Jiaqi Wang, and Shuicheng Yan. DOPD: Dual on-policy distillation. *arXiv preprint arXiv:2606.30626*, 2026. URL https://arxiv.org/abs/2606.30626.

Guibin Zhang, Jiayang Lyu, Ran Sun, Xinlei Yu, Haoyu Zhao, Qibing Ren, and Shuicheng Yan. Latent on-policy self-distillation. *arXiv preprint arXiv:2608.13040*, 2026a. URL https://arxiv.org/abs/2608.13040.

Miaosen Zhang, Yishan Liu, Shuxia Lin, Xu Yang, Qi Dai, Chong Luo, Weihao Jiang, Peng Hou, Anxiang Zeng, Xin Geng, and Baining Guo. Towards on-policy SFT: Distribution discriminant theory and its applications in LLM training. *arXiv preprint arXiv:2602.12222*, 2026b. URL https://arxiv.org/abs/2602.12222.

Shiwan Zhao, Xuyang Zhao, Jiaming Zhou, Aobo Kong, Qicheng Li, and Yong Qin. Mind the gap: Data rewriting for stable off-policy supervised fine-tuning. *arXiv preprint arXiv:2509.15157*, 2025. URL https://arxiv.org/abs/2509.15157.

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. *arXiv preprint arXiv:2601.18734*, 2026a. URL https://arxiv.org/abs/2601.18734.

Xuyang Zhao, Liting Zhang, Zichen Xu, Zhihu Wang, Caiyue Xu, Shiwan Zhao, and Qicheng Li. Is more privileged information better? from solution traces to problem-solving structure in self-distilled reasoning. *arXiv preprint arXiv:2608.01589*, 2026b. URL https://arxiv.org/abs/2608.01589.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023. URL https://arxiv.org/abs/2311.07911.

A Implementation and Evaluation Details

The source dataset contains 29,434 mathematical problem–solution pairs. The 4B and 8B controls reuse the same 3,200 examples across stages. Each stage generates 3,200 student rollouts, for 6,400 across BOPD's two stages.

Table 4 records the settings shared by the main training runs. We use LoRA on the attention projections `q_proj`, `k_proj`, `v_proj`, and `o_proj`, and the feed-forward projections `gate_proj`, `up_proj`, and `down_proj`. We report the step-100 checkpoint.

Table 4: Recorded training settings for the main mathematical experiments.

Setting	Value
Training examples available	29,434
Optimizer steps per stage	100
GPUs / batch / accumulation (4B)	4 / 4 / 2
GPUs / batch / accumulation (8B)	2 / 2 / 8
Effective batch size	32
Student rollouts per stage	3,200
LoRA rank / scaling parameter	64 / 128
Learning rate / gradient clipping	5×10^{-6} / 0.1
Distillation objective	Clipped forward KL
Pointwise clipping threshold	0.05
Loss parameters	$\beta = 0$, $\lambda = 1$
Rollout temperature / top-p / top-k	1.1 / 0.95 / 20
Maximum rollout output / sequence length	1,024 / 20,000 tokens
Precision / distributed training	BF16 / DeepSpeed ZeRO-2
Attention implementation	FlashAttention 2
Rollout engine	Colocated vLLM, tensor parallelism 1
vLLM memory fraction (4B / 8B)	0.4 / 0.35
Thinking mode (student / teacher)	Off / off
Teacher updates	None; teacher frozen within each stage
Stage II student initialization	Exact initial checkpoint

A.1 Evaluation Metrics and Protocols

For benchmark b, let $c_{i,j} \in \{0, 1\}$ indicate whether response j to question i is correct. With $N_b = 30$ and K responses per problem, we compute

$$\text{Avg@}\, K_b = \frac{100}{N_b K} \sum_{i=1}^{N_b} \sum_{j=1}^{K} c_{i,j}, \tag{8}$$

$$\text{Pass@}\, K_b = \frac{100}{N_b} \sum_{i=1}^{N_b} \mathbb{I}\left[\sum_{j=1}^{K} c_{i,j} > 0\right]. \tag{9}$$

Table 1 uses $K = 12$ at 4B and 8B; the ablation tables use $K = 6$. We average each metric over the three benchmarks for the table mean. Displayed values are rounded to two decimal places; the mean uses the underlying scores. The main mathematical evaluation sets `top_k` $= -1$, `min_p` $= 0$, and `presence_penalty` $= 0$, in addition to temperature 1.0, `top_p` $= 0.8$, and the 38,912-token output limit. Thinking mode is disabled for every evaluated model in the main tables.

Code-trained models use the same fixed 3,200 TACO problems for OPSD and BOPD. HumanEval+ and MBPP+ use the official EvalPlus tests; LiveCodeBench v5-new and v6-new contain the problems newly added relative to the preceding release. All four code benchmarks use greedy decoding, one response per problem, a 4,096-token generation limit, an 8,192-token context limit, and seed 0.

Table 5: IFEval accuracy (%) after mathematical training. P and I denote prompt and instruction accuracy. Trunc. counts responses reaching the output limit; they remain included in scores.

Size	Configuration	P strict	I strict	P loose	I loose	Trunc.
4B	Initial model	80.41	86.21	83.73	88.61	8
	OPSD (Stage I)	81.33	87.05	84.29	89.21	12
	Stage II OPSD	81.70	87.05	84.84	89.21	17
	BOPD	81.70	86.81	85.21	89.33	11
	OPSD (2× steps)	79.30	85.73	82.81	88.25	9
8B	Initial model	82.62	88.37	86.14	90.77	11
	OPSD (Stage I)	81.52	87.65	85.77	90.53	25
	Stage II OPSD	82.26	87.65	85.40	89.81	20
	BOPD	83.36	88.49	85.58	90.05	13
	OPSD (2× steps)	81.70	87.65	85.03	89.81	20

B Retention Evaluation Details

We evaluate all 541 IFEval prompts and all 164 HumanEval problems. Both evaluations use a 4,096-token output limit; IFEval additionally uses a 16,384-token context limit. Between 8 and 25 IFEval responses per configuration reach the output limit.

IFEval uses BF16, LoRA, greedy decoding, one response per prompt, and seed 0. The official Google scorer is pinned to commit `4700efb9afa54286b0e04473ba80a13e8461e25f`. All responses, including truncated ones, enter the reported scores. For code, we use EvalPlus 0.3.1 with HumanEvalPlus v0.1.10 and the full set of enhanced tests, greedy decoding, one response per problem, and seed 0.

Table 6: Code pass@1 (%) after mathematical training, with all 164 problems included. Trunc. counts output-limit truncations.

Size	Configuration	HumanEval	HumanEval+	Trunc.
4B	Initial model	83.54	74.39	2
	OPSD (Stage I)	82.32	75.00	1
	Stage II OPSD	84.15	78.66	1
	BOPD	85.98	78.05	0
	OPSD (2× steps)	82.93	76.22	2
8B	Initial model	84.76	81.71	0
	OPSD (Stage I)	84.76	81.10	1
	Stage II OPSD	84.76	81.10	1
	BOPD	83.54	81.10	1
	OPSD (2× steps)	85.98	81.10	0

C Mechanism Analysis Details

For the alignment curves in Figures 3 and 5, we use 64 fixed problems from the mathematical training subset. We evaluate OPSD checkpoints at steps 10, 20, . . ., 200 and BOPD checkpoints at steps 10, 20, . . ., 100; Figure 5 shows both methods through step 100. Each checkpoint generates fresh on-policy trajectories. We score the full-vocabulary distributions and aggregate over generated token positions. The entropy gap averages $|H(q_t) - H(p_t)|$ per position; it is not the absolute difference between mean entropies. All entropies use natural logarithms. Figure 5 plots checkpoint means without smoothing. Each curve covers 100 steps within its own stage.

D Ablation Details

OPSD (200 steps) continues the Stage I run and evaluates that model directly. These comparisons match nominal optimizer steps, not total FLOPs or wall time. Solution SFT uses reference-only

cross-entropy for 100 steps. The model run uses a learning rate of 5×10^{-6}. SFT→OPD freezes this teacher and trains a fresh initial-model student for another 100 steps. The SFT optimizer differs from OPSD, so this comparison evaluates the tested recipes rather than an optimally tuned SFT teacher. The Stage II reference ablation fixes the OPSD teacher, student initialization, loss, and step budget; only the teacher's reference access changes.

Table 7: Ablation scores (%). Each benchmark cell lists Avg@6 / Pass@6. Mean lists the corresponding unweighted benchmark means. Dashes mark benchmark-level Pass@6 scores unavailable for the separate 4B 200-step run.

Size	Configuration	AIME24	AIME25	HMMT25	Mean
4B	OPSD (2× steps)	12.22 / –	7.78 / –	7.22 / –	9.07 / 22.22
	Solution SFT	10.56 / 20.00	12.78 / 23.33	6.67 / 16.67	10.00 / 20.00
	SFT→OPD	20.56 / 43.33	15.00 / 36.67	7.78 / 23.33	14.44 / 34.44
	Stage II OPSD	16.67 / 40.00	14.44 / 36.67	5.56 / 16.67	12.22 / 31.11
8B	OPSD (2× steps)	20.00 / 36.67	16.67 / 43.33	5.00 / 16.67	13.89 / 32.22
	Solution SFT	14.44 / 36.67	7.22 / 23.33	2.78 / 10.00	8.15 / 23.33
	SFT→OPD	12.78 / 36.67	11.67 / 33.33	2.22 / 10.00	8.89 / 26.67
	Stage II OPSD	22.78 / 46.67	17.78 / 40.00	8.89 / 16.67	16.48 / 34.44

E Mechanism Analysis Details

Checkpoints and prompts. We select the same 64 problems from the mathematical training subset (seed 20260922) for every diagnostic run. We evaluate OPSD checkpoints at steps 10, 20, . . ., 200 and BOPD checkpoints at steps 10, 20, . . ., 100. Figure 3 displays OPSD every 20 steps; Figure 5 displays both methods through step 100. All rollouts use non-thinking mode. The corresponding frozen teacher is M_0 with access to each problem's reference solution for OPSD, and the reference-free OPSD step-100 model for BOPD.

On-policy teacher–student diagnostics. For each checkpoint, we generate fresh responses to the 64 prompts with that checkpoint and replay each generated prefix through both the student and its frozen teacher. At every generated token position we compute their full-vocabulary next-token distributions. We calculate top-16 overlap and overlap-token advantage as defined in Section 3, full-vocabulary $D_{\mathrm{KL}}(p_t \| q_t)$, and the absolute entropy gap $|H(p_t) - H(q_t)|$. Each plotted value is the mean over all evaluated token positions, so longer responses contribute more positions. The gap is averaged after taking the absolute value at each position; it is not the absolute difference between two mean entropies. We use natural logarithms for entropy and KL. Although a teacher's weights are frozen, its predictions can change across checkpoints because the student-generated prefixes change.

Fixed-prefix and on-policy student entropy. To produce the bottom row of Figure 3, we also generate a set of reference-free M_0 trajectories for the same 64 prompts and hold their prefixes fixed. At OPSD step s, we evaluate the student distribution $p_s(\cdot \mid u)$ at every position u in this fixed set and average its full-vocabulary entropy. For the on-policy curve, we instead average $H(p_s(\cdot \mid u))$ over positions in that checkpoint's own generated trajectories. Thus, the fixed-prefix curve holds the evaluated contexts constant across checkpoints, whereas the on-policy curve also reflects changes in the contexts visited by the student. The two curves need not contain the same number of token positions and are not paired position by position. The bottom-right KL curve uses the on-policy prefixes and the reference-conditioned OPSD teacher, as above.

Accuracy evaluation. The top row of Figure 3 evaluates AIME 2024, AIME 2025, and HMMT February 2025, which are separate from the 64 training prompts. It uses six responses per problem and reports Avg@6 under the mathematics decoding protocol of Section 5. We plot checkpoint values without smoothing. Steps on the OPSD and BOPD curves in Figure 5 are relative to their respective stages, not a shared cumulative training axis.

F FURTHER-DISTILLATION DETAILS

The M_3 and M_4 runs in Table 3 branch independently from M_2; M_4 is not derived from M_3. The M_5 and M_6 students are independent runs supervised by the same frozen M_4 teacher.

Each new stage runs for 100 steps with a fresh optimizer. Student and teacher weights are loaded independently, without stacking adapters. The corrected runs share a fixed 3,200-example list and the main distillation hyperparameters. The original M_2 run sampled from the full training dataset; an identical training-example set for that earlier run cannot be verified.

We reuse the existing M_3 checkpoint and reevaluate it alongside M_4–M_6. This evaluation yields 19.44% average accuracy and 37.78% Pass@6; the earlier evaluation of the same checkpoint yielded 17.22% and 34.44%, respectively. These are two evaluations of one trained model, not independent training runs. Table 3 uses the corrected batch's reevaluation consistently; the M_0, M_1, and M_2 rows retain their original main-table evaluations.