RECURSIVE SELF-IMPROVEMENT VIA ON-POLICY DISTILLATION FOR REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy distillation (OPD) trains a student model by having it generate trajectories, then matching its next-token predictions with an external teacher's next-token predictions. This provides dense, token-level supervision to the student. On-policy self-distillation (OPSD) eliminates the need for the external teacher. Specifically, a second frozen copy of the student model, now given the ground truth in its context, serves as the teacher. The student model only receives the problem and learns to mimic the privileged teacher model, while the teacher remains frozen throughout training. Previous work showed that freezing the teacher is useful for training stability, but we argue that this can prevent the teacher from incorporating the improvements learned by the student during training. Our primary contribution is to address this limitation with a recursive framework built around two complementary components. First, we let the privileged teacher co-evolve with the student so that revision learned in one round can guide the next, a process we refer to as **Dynamic Co-Evolution (DCE)**. Second, because stronger revision can also make responses too verbose and self-critical, we additionally train on shorter, verified rewrites of the model's own on-policy responses. We call this complementary objective **Self-Refined Concise Learning (SRCL)**. Overall, our comprehensive evaluations show that DCE+SRCL outperforms OPSD across multiple model scales and four competition-level mathematics benchmarks. Specifically, on Qwen3-8B, DCE+SRCL reaches 65.97% Average@12, outperforming OPSD by 35.62 percentage points while reducing mean output length by 7.80% relative to DCE alone.

1 INTRODUCTION

Post-training has played a central role in recent advances in LLM reasoning. A prominent approach is reinforcement learning with verifiable rewards (RLVR), which uses an automatic checker to assign an outcome reward to a completed solution (Shao et al., 2024; Yu et al., 2025). Because correctness can be verified without annotating every intermediate step, RLVR scales without step-level supervision and can elicit longer derivations, intermediate verification, backtracking, and self-correction (Guo et al., 2025). However, outcome rewards provide only coarse, sequence-level supervision: they indicate whether the final answer is correct, but not where the reasoning went wrong or how it should be revised. Process supervision provides more local feedback, but generally requires step-level annotations or a separately trained verifier (Lightman et al., 2024; Zhang et al., 2025b).

Between sparse outcome rewards and intensive process supervision, on-policy distillation (OPD) offers a third route: dense token-level targets on trajectories sampled from the model being trained (Agarwal et al., 2024; Lu & Lab, 2025). This model serves as the student, while a separate, stronger teacher provides next-token predictions for each generated prefix. However, standard OPD still requires a capable external teacher whose guidance is not conditioned on a ground-truth solution for the current problem. The student may therefore inherit the teacher's errors as well as its strengths.

On-policy self-distillation (OPSD) (Zhao et al., 2026) instead initializes two roles from the same base checkpoint. The student sees only the problem and generates an on-policy response; a copy of the base checkpoint also receives the ground-truth solution and serves as the privileged teacher. At each student-response prefix, the teacher supplies next-token predictions conditioned on the problem,

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.

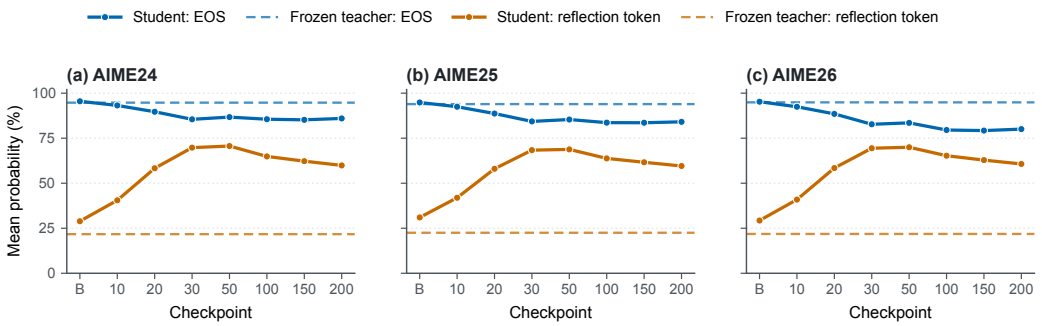


Figure 1: **Frozen teacher favors termination over reflection.** On fixed incorrect trajectories Y^-, the evolving student (solid) and gold-conditioned frozen teacher (dashed) score identical prefixes. Blue: $p(\text{EOS} \mid Y^-)$; orange: $p(r_t \mid Y^-_{<t})$ at observed reflection positions t.

solution, and prefix. OPSD transfers this guidance by minimizing the divergence between student and teacher predictions, but keeps the teacher frozen throughout training.

Does access to the ground-truth solution provide informative guidance at every student-generated prefix? We probe this question on fixed incorrect Qwen3-8B trajectories from AIME 2024, AIME 2025, and AIME 2026 (Mathematical Association of America, 2026), comparing evolving student checkpoints with the frozen, gold-conditioned teacher on the same prefixes. Despite seeing the solution, the frozen teacher assigns an average probability of 94.5% to EOS at incorrect response endpoints but only 22.0% to observed reflection cues such as `Wait`, a common reflection marker in reasoning models (Wang et al., 2025) (figure 1). This indicates limited guidance on how to recover from incorrect reasoning. Meanwhile, the student becomes substantially more likely to reflect during training, creating a growing mismatch with its fixed supervisor.

This mismatch exposes a limitation of freezing the privileged teacher. Revision behaviors such as self-verification, backtracking, and error correction can strengthen during post-training (Guo et al., 2025; Zhu et al., 2025), but a teacher fixed at the initial checkpoint cannot acquire these emerging capabilities. Even when conditioned on the ground-truth solution, its guidance may become increasingly misaligned with later student trajectories. This observation motivates a privileged teacher that evolves with the student.

We therefore introduce **Dynamic Co-Evolution** (DCE), in which the privileged teacher evolves alongside the student. After each update, the resulting checkpoint initializes both the next student and a detached, gold-conditioned privileged teacher. Revision behavior acquired in one round can thus improve the next round's supervision. Stronger checking and backtracking, however, can also make reasoning unnecessarily long. We therefore pair DCE with **Self-Refined Concise Learning (SRCL)**, which trains on shorter, answer-verified rewrites of the same on-policy responses. DCE improves revision, while SRCL retains that capability without unnecessary token cost.

Our contributions are as follows:

- We introduce **Dynamic Co-Evolution (DCE)**, a recursive on-policy self-distillation framework in which each updated checkpoint initializes both the next student and a detached, gold-conditioned privileged teacher. Across four competition-level mathematical reasoning benchmarks, DCE improves Average@12 accuracy over OPSD by 35.41, 37.15, and 12.98 percentage points at 8B, 4B, and 1.7B, respectively.
- We introduce **Self-Refined Concise Learning (SRCL)** to control the reasoning cost that can accompany stronger revision. SRCL trains on accepted, shorter self-refinements of the same on-policy responses. Under our lowest tested 8K evaluation budget, DCE+SRCL achieves 35.07% Average@12 accuracy with 7,500 mean output tokens per response, outperforming the OPSD test-time-scaling control by 6.60 percentage points while generating approximately 692 fewer tokens per response.
- We show that the privileged teacher learns to provide stronger revision guidance as it evolves with the student. Matched-budget controls show that longer generation alone does

not explain the gains. Updating the teacher every round also outperforms frozen, EMA, and periodic alternatives. Fixed-trace probes corroborate the teacher's improved revision behavior across three AIME cohorts. Its endpoint EOS probability falls from 90.4% to 41.3%, while its probability on observed reflection tokens rises from 32.8% to 77.4%.

2 Related Work

2.1 Reflection and Verification in Reasoning Post-Training

Outcome-supervised post-training can elicit reflection-like behaviors without step-level labels. GRPO removes the learned critic used by PPO and estimates advantages from relative rewards within a sampled group, making outcome-based training practical for mathematical reasoning (Shao et al., 2024). DAPO introduces clip-higher, dynamic sampling, and token-level policy loss to improve the stability and efficiency of this recipe at scale (Yu et al., 2025). DeepSeek-R1 further demonstrates that outcome-based training alone can produce rechecking, backtracking, and "aha moments" without explicit reflection supervision (Guo et al., 2025). Related work extends RL post-training beyond closed-form math to content moderation (Firooz et al., 2025), open-ended environments without fixed-answer verification (Yin et al., 2026a), and non-verifiable preference tasks (Yin & Shi, 2026). Outcome-level rewards still score completed responses as a whole rather than the local transition where the model identifies and repairs an error.

A closer look suggests that these reflection-like behaviors may not be newly learned through training. R1-Zero reproductions find similar behaviors already present in some base models and attribute part of the GRPO effect to a length bias toward longer, often incorrect, responses (Liu et al., 2025). Across ten base models, response length and verification behavior do not reliably emerge together (Zeng et al., 2025). Activation-space analysis further reveals a latent, though rare, capacity for reflection that exists before any RLVR (Zhu et al., 2025). Together, these results suggest that outcome-based post-training amplifies a pre-existing behavioral prior rather than teaching the model how to recover from a particular wrong prefix.

Even when reflection does appear in generation, it does not reliably correct errors. Without external feedback, prompting a model to revise its own answer can reduce accuracy (Huang et al., 2024), and hidden-state probes reveal correctness signals that the model's generation does not always exploit (Zhang et al., 2025a; Lee et al., 2025). The gap between latent awareness and effective revision motivates a different form of supervision: DCE provides dense, token-level guidance on the model's own incorrect reasoning, directly training the transition from error recognition to successful revision.

2.2 On-Policy and Privileged Self-Distillation

OPD trains on student-sampled responses while a teacher supplies next-token supervision on those same responses (Agarwal et al., 2024; Lu & Lab, 2025). OPSD instead uses a frozen copy of the initial checkpoint as a privileged teacher: the student sees only the problem, while the teacher also receives a ground-truth solution (Zhao et al., 2026). RLSD and RLCSD combine self-distillation with reinforcement learning (Yang et al., 2026; Pan et al., 2026).

Let p_θ denote the model with parameters θ. The student sees a problem x and generates $y \sim p_\theta(\cdot \mid x)$. The privileged teacher also receives a ground-truth solution g and instruction τ when scoring each prefix $y_{<t}$. Standard OPSD presents g as user-side reference material in $\mathcal{C}_t^{\mathrm{ref}}(x, g, \tau, y_{<t})$. Let q and p denote the teacher and student next-token distributions and $\Delta(q, p)$ their divergence. Our main experiments use Forward KL, $\Delta(q, p) = D_{\mathrm{KL}}(q\|p) = \sum_v q(v) \log \frac{q(v)}{p(v)}$; alternatives are evaluated in section A.4. Standard OPSD minimizes

$$\mathcal{L}_{\mathrm{fixed}}(\theta) = \mathbb{E}_{(x,g),y}\left[\frac{1}{|y|}\sum_{t=1}^{|y|}\Delta\big(p_{\theta_0}\big(\cdot \mid \mathcal{C}_t^{\mathrm{ref}}(x, g, \tau, y_{<t})\big), p_\theta(\cdot \mid x, y_{<t})\big)\right], \tag{1}$$

where the teacher remains fixed at θ_0 and receives no gradients, even as the student and its generated responses change throughout training.

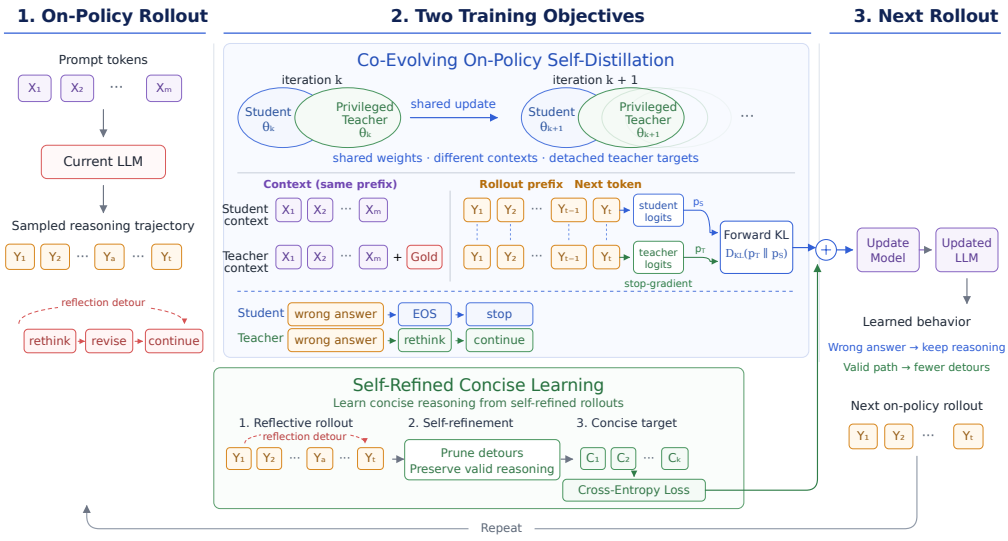


Figure 2: **Overview of recursive co-evolution.** At round k, the current model generates an on-policy rollout. DCE supplies gold-conditioned guidance along that rollout, while SRCL learns from accepted, shorter self-refinements. Their joint update produces θ_{k+1}, which initializes both roles in the next round and carries newly acquired revision behavior into future supervision.

2.3 Self-Refinement and Efficient Reasoning

Self-training turns a model's own generations into new training data. STaR iteratively trains on rationales that yield correct answers and uses answer-conditioned rationalization to recover additional examples, while ReST generates, filters, and reuses model samples as offline data (Zelikman et al., 2022; Gülçehre et al., 2023). In alignment, SAO generates its own prompts, responses, and preferences without external annotation (Yin et al., 2026b), whereas PIKA uses an external generator and reward model to construct synthetic training data (Yin et al., 2025). Reflexion stores verbal feedback to guide later attempts without updating parameters (Shinn et al., 2023), and subsequent work links verification, backtracking, and subgoal construction to successful self-improvement (Gandhi et al., 2025). SRCL adopts this filtered self-training view for a different purpose: the current checkpoint rewrites its own on-policy response without seeing the verified solution, and training keeps only rewrites that are shorter, naturally terminated, structurally valid, and answer-correct, so SRCL teaches concise successful solutions while DCE teaches how to revise.

Test-time scaling instead allocates more computation at inference. Repeated sampling, search, and compute-aware decoding trade inference compute for accuracy (Brown et al., 2024; Snell et al., 2025; Wu et al., 2025); input-adaptive methods allocate it by estimated difficulty (Damani et al., 2025); and budget forcing uses continuation cues to prevent early termination and extend a response to a prescribed budget (Muennighoff et al., 2025). SRCL changes the model during post-training so a single rollout preserves correct reasoning with fewer tokens, and our matched-budget controls test whether the gains come merely from longer generation.

3 Method

To address the limitations of a frozen privileged teacher while controlling the cost of increasingly long reasoning, our framework combines two complementary objectives. DCE distills gold-conditioned guidance along the current student's rollout, while SRCL learns from shorter, verified self-refinements of that response. Their joint update produces the checkpoint used as both the student and privileged teacher in the next round, closing the recursive loop in figure 2.

3.1 DYNAMIC CO-EVOLUTION

DCE alternates on-policy generation with dynamic teacher updates. Starting from θ_0, the current checkpoint samples a response $y^{(k)} \sim p_{\theta_k}(\cdot \mid x)$ at round k. Let $\widetilde{y}^{(k)} = (\widetilde{y}_1^{(k)}, \ldots, \widetilde{y}_m^{(k)})$ denote the response tokens used to compute the guidance loss. At each position t, both branches predict the next token in this response. The student conditions on the problem x and the preceding response tokens $\widetilde{y}_{<t}^{(k)}$, whereas the privileged branch opens a single assistant turn with the ground-truth solution g, then a transition instruction τ asking the model to solve the problem in its own way, and only then the same response prefix. We denote this input by $\mathcal{C}_t^{\mathrm{prev}}(x, g, \tau, \widetilde{y}_{<t}^{(k)})$, the concatenation of these privileged inputs up to position t. Placing g in the assistant turn lets the model condition on it as preceding reasoning rather than as external user-provided material. The exact templates and placement ablation appear in figure 7 and section 5.4; g shapes the teacher distribution without becoming a sequence target. We define

$$\begin{aligned} p_{k,t} &:= p_{\theta_k}\left(\cdot \mid x, \widetilde{y}_{<t}^{(k)}\right), \\ q_{k,t} &:= \mathrm{stopgrad}\left[p_{\theta_k}\left(\cdot \mid \mathcal{C}_t^{\mathrm{prev}}(x, g, \tau, \widetilde{y}_{<t}^{(k)})\right)\right]. \end{aligned} \tag{2}$$

Here $p_{k,t}$ and $q_{k,t}$ are the student and ground-truth-conditioned teacher distributions from the current checkpoint θ_k; gradients are stopped through $q_{k,t}$. Standard OPSD instead places g in the user turn and keeps the teacher at θ_0. With $\Delta(q, p)$ denoting a divergence between next-token distributions, DCE minimizes

$$\mathcal{L}_G^{(k)} = \mathbb{E}_{(x,g),\, y^{(k)} \sim p_{\theta_k}(\cdot|x)}\left[\frac{1}{m}\sum_{t=1}^{m} \Delta(q_{k,t}, p_{k,t})\right]. \tag{3}$$

After the round's update, θ_{k+1} generates the next on-policy response and initializes both the student and privileged teacher. Revision behavior learned in one round can therefore become part of the next round's gold-conditioned supervision, producing recursive co-evolution. We use Forward KL for Δ in the main experiments and compare Reverse KL and JSD in section A.4.

3.2 SELF-REFINED CONCISE LEARNING

Reflection helps a model recover, but it can also introduce failed branches, repeated calculations, unnecessary revisions, and continued reasoning after a correct solution has been reached. SRCL converts the same on-policy rollout used by DCE into a clean SFT target. For a minibatch $\mathcal{B} = \{(x_i, g_i)\}$ of problems and ground-truth solutions, the current checkpoint rewrites each response $y_i^{(k)}$ as

$$c_i^{(k)} = R_{\theta_k}\left(x_i, y_i^{(k)}\right), \tag{4}$$

using a prompt that asks for a direct, self-contained solution with detours and reflection removed. A rewrite is accepted only when it terminates naturally, is shorter than its source, passes the structural checks, and matches the ground-truth answer. Let $A_i^{(k)} \in \{0, 1\}$ indicate that all four criteria hold; the exact rules are given in section L. Each accepted response becomes an SRCL target, learned with the standard token-level cross-entropy:

$$\mathcal{L}_{\mathrm{SRCL}}^{(k)} = -\frac{1}{N_k}\sum_{i \in \mathcal{B}} A_i^{(k)} \sum_{t=1}^{|c_i^{(k)}|} \log p_{\theta_k}\left(c_{i,t}^{(k)} \mid x_i, c_{i,<t}^{(k)}\right), \qquad N_k = \sum_{i \in \mathcal{B}} A_i^{(k)} |c_i^{(k)}|. \tag{5}$$

If no rewrite is accepted, the SRCL term is omitted and the batch is trained with DCE alone. The ground-truth solution is used only to filter candidates and is never provided during rewrite generation, preventing leakage into the learned rewriting behavior. The exact rewrite prompt, acceptance rules, and answer-verifier implementation are reported in sections L and M.

3.3 JOINT OPTIMIZATION

DCE and SRCL provide complementary supervision on the same on-policy response. DCE supplies token-level guidance along the current rollout, while SRCL learns from accepted concise targets.

Their joint objective is

$$\mathcal{L}^{(k)}_{\text{DCE+SRCL}} = \underbrace{\lambda_G \mathcal{L}^{(k)}_{G}}_{\text{DCE guidance}} + \underbrace{\lambda_S \mathcal{L}^{(k)}_{\text{SRCL}}}_{\text{SRCL target}}. \tag{6}$$

Here λ_G and λ_S are outer loss weights applied after the two token-level losses are computed: λ_G controls DCE guidance and λ_S controls the SRCL target loss. Their empirical sensitivity is reported jointly in section A.3. Each update produces the model that assumes both roles in the next round; algorithm 1 in section I states the full recursive procedure step by step.

4 Experimental Setup

Models, data, and metrics. We evaluate Qwen3-1.7B, Qwen3-4B, Qwen3-8B, and Qwen3-14B (Qwen Team, 2025) as our primary model family, and Gemma-4-12B-IT in the cross-family study in section A.2. Both the problem-only student and gold-conditioned privileged teacher operate in non-thinking mode. All methods use the same 14,717 problems drawn from the OpenThoughts mathematical-reasoning data adopted by OPSD (Guha et al., 2025; Zhao et al., 2026). DCE constructs student and privileged views of each problem during training; these paired views are not additional independent problems. Complete configurations are reported in table 26. Evaluation uses all 30 problems from each of AIME 2024, AIME 2025, AIME 2026 (Mathematical Association of America, 2026), and HMMT February 2025 (Harvard–MIT Mathematics Tournament, 2026). We sample 12 responses per problem with a 32K generation cap, temperature 1.0, top-$p = 0.8$, top-$k = -1$, and repetition penalty 1.0.

Baselines and controls. We compare against four training baselines: **Base**, **SFT**, **GRPO**, and **OPSD**. **Base** is the original model evaluated in non-thinking mode. **SFT** trains on the problem–solution pairs (x, g) using autoregressive cross-entropy, while **GRPO** optimizes an outcome reward based on final-answer correctness. **OPSD** is our closest training baseline: it distills next-token predictions from a frozen, gold-conditioned privileged teacher initialized from the same base checkpoint. To separate the effect of improved training from that of simply allocating more inference tokens, we additionally include **OPSD-TTS** as an inference-time control. Following the extended-thinking intervention of Ghosal et al. (2025), when an OPSD response terminates before the target budget, a `Wait` cue resumes generation until the response reaches an exact 8K or 16K token budget. OPSD-TTS does not modify the model parameters and therefore tests whether longer generation alone can account for the gains of DCE+SRCL.

Training protocol. Across the Qwen3 DCE experiments, we train with AdamW (Loshchilov & Hutter, 2019) in bfloat16 using a learning rate of 5×10^{-6} and rank-128 LoRA (Hu et al., 2022). Each training example produces a single on-policy response. DCE computes privileged-teacher guidance over the retained response tokens, while SRCL greedily rewrites the same response and trains only on accepted targets that are shorter, reflection-free, and answer-correct. We set the DCE loss weight to $\lambda_G = 5 \times 10^4$ for all Qwen3 models. For SRCL, we use $\lambda_S = 12.5, 25, 35$, and 25 for Qwen3-14B, 8B, 4B, and 1.7B, respectively. Sensitivity to both coefficients is analyzed in section A.3. We evaluate checkpoints at training steps $\{10, 20, 30, 50, 100, 150, 200\}$ when available. Baseline-specific SFT and GRPO hyperparameters are reported in table 26, and checkpoint-level learning trajectories are provided in figure 3 and section B. The Gemma transfer configuration is reported separately in section A.2.

5 Results

5.1 Main Experimental Results

Table 1 summarizes the main results. DCE substantially improves over the training baselines across model scales. For Qwen3-8B, DCE+SRCL reaches 65.97% Average@12, compared with 30.35% for OPSD and 20.28% for GRPO. The gains remain substantial with Qwen3-4B, where DCE+SRCL reaches 61.88%, compared with 22.85% for OPSD and 18.47% for GRPO. DCE alone achieves similar accuracy with Qwen3-8B (65.76%) and reaches 60.00% with Qwen3-4B, indicating that the primary accuracy gains come from dynamic co-evolution.

SRCL improves the accuracy–length tradeoff at these scales. With Qwen3-8B, adding SRCL reduces mean output length from 19,046 to 17,561 tokens while maintaining comparable Average@12 accuracy (65.76% versus 65.97%). With Qwen3-4B, SRCL both improves accuracy from 60.00% to 61.88% and reduces mean output length from 19,360 to 17,265 tokens, a 10.82% reduction. This effect is not uniform at the smallest scale: with Qwen3-1.7B, DCE+SRCL improves Average@12 from 23.33% to 26.88%, but also increases mean output length from 12,600 to 19,504 tokens.

Longer inference alone does not explain the gains. Forcing OPSD to generate exactly 16K tokens yields only 30.76%, 22.99%, and 9.24% Average@12 for Qwen3-8B, Qwen3-4B, and Qwen3-1.7B, respectively. The matched training trajectories in figure 3 further separate the effect of SRCL from checkpoint selection: at step 100 with Qwen3-8B, DCE+SRCL reaches 65.97% with 17,561 mean tokens, compared with 64.93% and 19,036 tokens for DCE at the same step. With Qwen3-4B, DCE peaks earlier and declines after step 50, whereas DCE+SRCL remains near 61% through step 200. Complete checkpoint trajectories and budget controls are provided in sections A.5, B and E and table 24.

Table 1: **Main results.** Cells report Average@12 accuracy (**Acc.**, %) and mean generated tokens (**Tok.**). Unmarked rows use non-thinking, 32K decoding. Gray Base (thinking)‡ rows are reference-only and excluded from the main ranking; † marks inference-only 16K OPSD-TTS. Blue bold marks the best non-thinking accuracy within each model size; complete trajectories appear in figure 3 and section B.

Method	AIME24		AIME25		AIME26		HMMT25		Average	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
Qwen3–8B										
Base	28.89	4,706	21.39	3,189	17.78	4,631	10.00	3,161	19.51	3,922
SFT	28.33	4,662	21.94	3,583	18.06	5,075	11.67	3,030	20.00	4,088
GRPO	31.11	4,710	22.50	3,242	16.39	4,007	11.11	2,860	20.28	3,705
OPSD	46.11	5,678	28.61	6,458	29.72	6,371	16.94	5,529	30.35	6,009
OPSD-TTS†	46.39	16,384	28.89	16,384	31.11	16,384	16.67	16,384	30.76	16,384
DCE	**74.72**	16,869	68.61	18,562	**72.78**	17,550	**46.94**	23,203	65.76	19,046
+ SRCL	73.89	15,640	**71.94**	17,241	71.94	16,158	46.11	21,205	**65.97**	17,561
Base (thinking)‡	75.00	15,565	64.72	18,784	64.44	17,192	44.17	20,940	62.08	18,120
Qwen3–4B										
Base	21.39	4,754	20.56	3,514	17.22	4,142	11.11	2,809	17.57	3,805
SFT	23.89	4,522	20.83	3,388	19.17	4,083	10.83	2,880	18.68	3,718
GRPO	25.28	3,577	20.83	3,082	16.11	3,915	11.67	2,876	18.47	3,363
OPSD	30.83	9,419	22.50	7,970	24.17	8,228	13.89	7,782	22.85	8,350
OPSD-TTS†	30.28	16,384	22.78	16,384	25.56	16,384	13.33	16,384	22.99	16,384
DCE	68.06	17,777	58.89	19,644	69.17	17,944	43.89	22,073	60.00	19,360
+ SRCL	**74.17**	15,273	**59.44**	17,466	**69.72**	16,181	**44.17**	20,143	**61.88**	17,265
Base (thinking)‡	72.50	15,013	61.39	18,037	64.44	15,996	41.67	18,631	60.00	16,919
Qwen3–1.7B										
Base	14.44	3,464	7.78	2,633	8.33	3,892	6.11	2,350	9.17	3,085
SFT	15.00	3,600	8.33	2,584	11.11	3,957	6.11	2,568	10.14	3,177
GRPO	14.44	3,539	11.39	2,546	10.00	3,715	7.78	2,412	10.90	3,053
OPSD	16.11	6,556	7.78	5,050	9.72	6,164	7.78	3,751	10.35	5,380
OPSD-TTS†	14.72	16,384	7.50	16,384	8.89	16,384	5.83	16,384	9.24	16,384
DCE	31.39	13,709	24.72	11,848	20.83	12,943	16.39	11,898	23.33	12,600
+ SRCL	**33.61**	19,849	**28.33**	18,396	**28.89**	19,725	**16.67**	20,046	**26.88**	19,504
Base (thinking)‡	48.33	17,581	38.89	17,559	36.94	18,260	23.33	18,533	36.88	17,983

5.2 Does the Privileged Teacher Need to Co-Evolve?

To isolate the effect of teacher refresh, we compare DCE with an otherwise matched variant whose gold-conditioned teacher remains frozen, both with and without SRCL. Co-evolution produces a large and consistent advantage across model sizes: DCE reaches 65.76% at 8B and 60.00% at 4B,

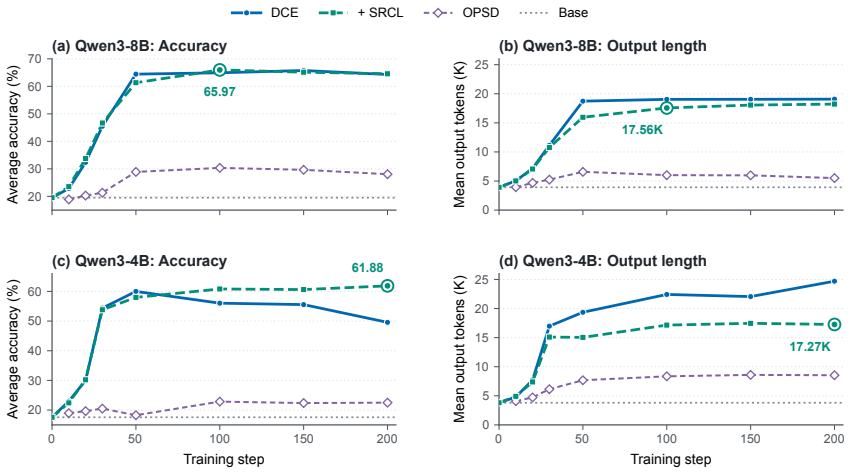


Figure 3: **Learning dynamics across model scales.** Qwen3-8B (top) and Qwen3-4B (bottom) under non-thinking, 32K, Average@12 evaluation. Each row reports Average accuracy (left) and mean generated tokens (right). DCE+SRCL peaks at step 100 for 8B and remains stable through step 200 for 4B; DCE peaks at steps 150 and 50, respectively.

compared with 47.99% and 25.28% for its frozen-teacher counterpart, and with SRCL the gaps widen to 65.97% versus 41.04% at 8B and 61.88% versus 27.29% at 4B (figure 5).

The frozen runs are shorter largely because they terminate prematurely, not because they reach a better accuracy–length frontier. EMA and periodic refresh substantially narrow the gap but remain below fully dynamic updates, which allow newly learned revision behavior to enter the privileged signal at every round. The accuracy–length summary, complete schedules, trajectories, and task-level results are reported in section C, figures 5 and 6, and table 13.

5.3 What Happens During Recursive Self-Improvement?

We score fixed incorrect trajectories from AIME24, AIME25, and AIME26 at every checkpoint. For both the student and its gold-conditioned teacher, we measure $p(\mathrm{EOS})$ after the full response and the probability of the observed reflection cue, such as `Wait`, at the earlier prefix where revision begins. Keeping the prefixes fixed isolates changes in next-token preference; formal definitions and cohort sizes are given in section F.

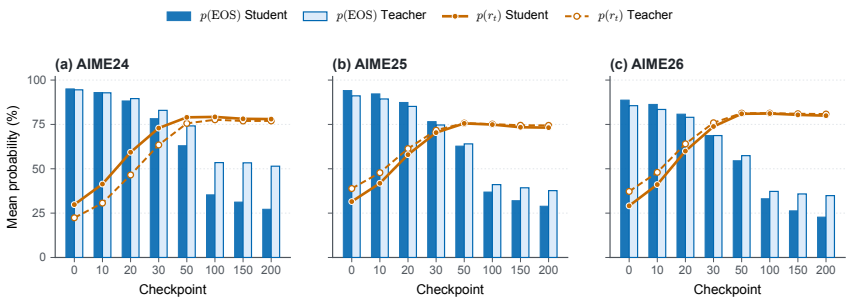


Figure 4: **Revision behavior during recursive self-improvement.** On fixed incorrect trajectories, endpoint $p(\mathrm{EOS})$ decreases while the probability of the observed reflection cue increases. Dark/-solid and light/dashed marks denote the student and teacher.

Across the three cohorts, student/teacher $p(\mathrm{EOS})$ falls from 92.6%/90.4% at initialization to 26.3%/41.3% at step 200, while reflection-cue probability rises from 30.1%/32.8% to 77.0%/77.4%. The privileged branch therefore evolves with the student, acquiring a stronger preference to revise

rather than terminate an incorrect solution. This complements the frozen-teacher mismatch in figure 1 and directly supports co-evolution's round-to-round transfer.

5.4 Which Representation of the Verified Solution Works Better?

Privileged-solution placement also affects signal quality: assistant-side prefill outperforms user-side instruction-last by 9.79, 6.39, and 14.66 Average@12 points at 8B, 4B, and 1.7B, respectively (table 2).

Table 2: Privileged-solution placement across model scales under non-thinking, 32K evaluation. Cells report Average@12 accuracy (Acc., %) and mean tokens (Tok.); task-level results appear in table 17.

Model	Assistant-side		User-side		Δ Acc.
	Acc.	Tok.	Acc.	Tok.	
Qwen3–8B	**65.97**	17,561	56.18	18,341	+9.79
Qwen3–4B	**61.88**	17,265	55.49	15,072	+6.39
Qwen3–1.7B	**26.88**	19,504	12.22	6,426	+14.66

At the matched 8B checkpoint, assistant-side influence is strongest near the response start, while user-side influence persists later (table 18). Under a frozen teacher, reference-last instead exceeds assistant-side prefill (table 16), so placement alone is insufficient; figure 7 gives the exact prompts.

5.5 Does the Advantage Persist under Tight Budgets?

We compare three inference-only controls under matched 8K and 16K output budgets. *Truncate* cuts the original 32K response at the budget; *Cue* generates the first 7K (8K budget) or 14K (16K budget) tokens, appends a short continuation instruction, and completes the response within the same cap; *TTS* extends OPSD responses to the exact budget with a `Wait` cue. Table 3 shows that forced OPSD-TTS tokens give little benefit, whereas DCE+SRCL remains substantially stronger without exhausting the budget.

Table 3: Qwen3-8B under matched 8K and 16K output budgets. Cells report Average@12 accuracy (Acc., %) and mean output tokens (Tok.); task-level results appear in table 25. Blue bold marks the strongest result at each budget.

Method	Condition	8K		16K	
		Acc.	Tok.	Acc.	Tok.
OPSD	Standard	29.17	4,252	30.28	5,109
OPSD	**TTS**	28.47	8,192	30.76	16,384
DCE	Truncate	24.38	7,795	48.68	13,392
DCE	**Cue**	31.04	7,746	55.76	13,171
DCE+SRCL	Truncate	28.54	7,565	52.01	12,724
DCE+SRCL	**Cue**	**35.07**	7,500	**57.64**	12,407

6 Conclusion

We presented DCE, a recursive on-policy self-distillation framework in which each updated checkpoint serves as both the next student and a detached, gold-conditioned privileged teacher, so that revision learned in one round guides the next. SRCL complements DCE by training on shorter, answer-verified rewrites of the model's own responses. Experiments across multiple benchmarks and model sizes show that DCE+SRCL consistently outperforms OPSD, that refreshing the privileged teacher is the key ingredient, and that SRCL shortens outputs without sacrificing accuracy at 4B and 8B. Fixed-trace probes further show that the co-evolving teacher shifts from favoring termination to supporting revision.

AI USE STATEMENT

We did not use generative AI tools for any task on the required-disclosure list: the research questions, hypotheses, method design and implementation, experiments, data, and interpretation of results are the authors' own. We used generative AI tools only to polish the content, namely to improve the wording and readability of the manuscript, format the LaTeX source and references, and assist with plotting and code review. The authors reviewed and verified all AI-assisted edits and take full responsibility for the final content of this work, including text produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

Section 4 states the models, data split, decoding protocol, and loss weights used for every reported run. Section I lists the complete optimization and evaluation configuration for DCE, SFT, GRPO, and OPSD; section L specifies the SRCL acceptance rules and answer verifier; section M gives the exact prompt serializations for the student, privileged, and rewrite contexts; and sections B and C report checkpoint-wise trajectories for every training curve shown in the main text.

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, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=3zKtaqxLhW.

Bradley C. A. Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *CoRR*, abs/2407.21787, 2024. doi: 10.48550/ARXIV.2407.21787. URL https://doi.org/10.48550/arXiv.2407.21787.

Mehul Damani, Idan Shenfeld, Andi Peng, Andreea Bobu, and Jacob Andreas. Learning how hard to think: Input-adaptive allocation of LM computation. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=6qUUgw9bAZ.

Hamed Firooz, Rui Liu, Yuchen Lu, Zhenyu Hou, Fangzhou Xiong, Xiaoyang Zhang, Changshu Jian, Zhicheng Zhu, Jiayuan Ma, Jacob Tao, Chaitali Gupta, Xiaochang Peng, Shike Mei, Hang Cui, Yang Qin, Shuo Tang, Jason Gaedtke, and Arpit Mittal. Scaling reinforcement learning for content moderation with large language models. *CoRR*, abs/2512.20061, 2025. doi: 10.48550/ARXIV.2512.20061. URL https://doi.org/10.48550/arXiv.2512.20061.

Kanishk Gandhi, Ayush Chakravarthy, Anikait Singh, Nathan Lile, and Noah D. Goodman. Cognitive behaviors that enable self-improving reasoners, or, four habits of highly effective stars. *CoRR*, abs/2503.01307, 2025. doi: 10.48550/ARXIV.2503.01307. URL https://doi.org/10.48550/arXiv.2503.01307.

Soumya Suvra Ghosal, Souradip Chakraborty, Avinash Reddy, Yifu Lu, Mengdi Wang, Dinesh Manocha, Furong Huang, Mohammad Ghavamzadeh, and Amrit Singh Bedi. Does thinking more always help? mirage of test-time scaling in reasoning models. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/fc067ac218430c409d6f65403328f740-Abstract-Conference.html.

Etash Kumar Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, Ashima Suvarna, Benjamin Feuer,

Liangyu Chen, Zaid Khan, Eric Frankel, Sachin Grover, Caroline Choi, Niklas Muennighoff, Shiye Su, Wanjia Zhao, John Yang, Shreyas Pimpalgaonkar, Kartik Sharma, Charlie Cheng-Jie Ji, Yichuan Deng, Sarah M. Pratt, Vivek Ramanujan, Jon Saad-Falcon, Jeffrey Li, Achal Dave, Alon Albalak, Kushal Arora, Blake Wulfe, Chinmay Hegde, Greg Durrett, Sewoong Oh, Mohit Bansal, Saadia Gabriel, Aditya Grover, Kai-Wei Chang, Vaishaal Shankar, Aaron Gokaslan, Mike A. Merrill, Tatsunori Hashimoto, Yejin Choi, Jenia Jitsev, Reinhard Heckel, Maheswaran Sathiamoorthy, Alexandros G. Dimakis, and Ludwig Schmidt. Openthoughts: Data recipes for reasoning models. *CoRR*, abs/2506.04178, 2025. doi: 10.48550/ARXIV.2506.04178. URL https://doi.org/10.48550/arXiv.2506.04178.

Çaglar Gülçehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. Reinforced self-training (rest) for language modeling. *CoRR*, abs/2308.08998, 2023. doi: 10.48550/ARXIV.2308.08998. URL https://doi.org/10.48550/arXiv.2308.08998.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, Hao Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, Tao Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nat.*, 645(8081):633–638, 2025. doi: 10.1038/S41586-025-09422-Z. URL https://doi.org/10.1038/s41586-025-09422-z.

Harvard–MIT Mathematics Tournament. Past tournaments, 2026. URL https://www.hmmt.org/www/archive/problems. Accessed August 26, 2026.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In Joaquin Vanschoren and Sai-Kit Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

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. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=IkmD3fKBPQ.

Andrew Lee, Lihao Sun, Chris Wendler, Fernanda B. Viégas, and Martin Wattenberg. The geometry of self-verification in a task-specific reasoning model. *CoRR*, abs/2504.14379, 2025. doi: 10.48550/ARXIV.2504.14379. URL https://doi.org/10.48550/arXiv.2504.14379.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *CoRR*, abs/2503.20783, 2025. doi: 10.48550/ARXIV.2503.20783. URL https://doi.org/10.48550/arXiv.2503.20783.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

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.

Mathematical Association of America. MAA invitational competitions, 2026. URL https://maa.org/maa-invitational-competitions/. Accessed August 26, 2026.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel J. Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 20275–20321. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.EMNLP-MAIN.1025. URL https://doi.org/10.18653/v1/2025.emnlp-main.1025.

Leyi Pan, Shuchang Tao, Yunpeng Zhai, Lingzhe Zhang, Zhaoyang Liu, Bolin Ding, Aiwei Liu, and Lijie Wen. RLCSD: reinforcement learning with contrastive on-policy self-distillation. *CoRR*, abs/2606.11709, 2026. doi: 10.48550/ARXIV.2606.11709. URL https://doi.org/10.48550/arXiv.2606.11709.

Qwen Team. Qwen3 technical report. *CoRR*, abs/2505.09388, 2025. doi: 10.48550/ARXIV.2505.09388. URL https://doi.org/10.48550/arXiv.2505.09388.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. *CoRR*, abs/2311.12022, 2023. doi: 10.48550/ARXIV.2311.12022. URL https://doi.org/10.48550/arXiv.2311.12022.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024. doi: 10.48550/ARXIV.2402.03300. URL https://doi.org/10.48550/arXiv.2402.03300.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/1b44b878bb782e6954cd888628510e90-Abstract-Conference.html.

Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=4FWAwZtd2n.

Chenlong Wang, Yuanning Feng, Dongping Chen, Zhaoyang Chu, Ranjay Krishna, and Tianyi Zhou. Wait, we don't need to "wait"! removing thinking tokens improves reasoning efficiency. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025, Suzhou, China, November 4-9, 2025*, pp. 7459–7482. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.FINDINGS-EMNLP.394. URL https://doi.org/10.18653/v1/2025.findings-emnlp.394.

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for LLM problem-solving. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=VNckp7JEHn.

Chenxu Yang, Chuanyu Qin, Qingyi Si, Minghui Chen, Naibin Gu, Dingyu Yao, Zheng Lin, Weiping Wang, Jiaqi Wang, and Nan Duan. Self-distilled RLVR. *CoRR*, abs/2604.03128, 2026. doi: 10.48550/ARXIV.2604.03128. URL https://doi.org/10.48550/arXiv.2604.03128.

Shangjian Yin and Zhouxing Shi. From individual to common: An early exploration of consensus in non-verifiable data for balanced preference optimization. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 34612–34630. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.1598. URL https://aclanthology.org/2026.acl-long.1598/.

Shangjian Yin, Shining Liang, Wenbiao Ding, Yuli Qian, Zhouxing Shi, Hongzhi Li, and Yutao Xie. PIKA: Expert-level synthetic datasets for post-training alignment from scratch. *CoRR*, abs/2510.06670, 2025. doi: 10.48550/ARXIV.2510.06670. URL https://arxiv.org/abs/2510.06670.

Shangjian Yin, Yu Fu, Yue Dong, and Zhouxing Shi. GRLO: Towards generalizable reinforcement learning in open-ended environments from zero. *CoRR*, abs/2605.15464, 2026a. doi: 10.48550/ARXIV.2605.15464. URL https://arxiv.org/abs/2605.15464.

Shangjian Yin, Zhepei Wei, Xinyu Zhu, Wei-Lin Chen, and Yu Meng. Aligning large language models via fully self-synthetic data. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 34553–34568. Association for Computational Linguistics, 2026b. doi: 10.18653/v1/2026.acl-long.1595. URL https://aclanthology.org/2026.acl-long.1595/.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Juncai Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Ru Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Yonghui Wu, and Mingxuan Wang. DAPO: an open-source LLM reinforcement learning system at scale. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/a4277440d50f1f15d2cb4c14f7e0c0d2-Abstract-Conference.html.

Eric Zelikman, Yuhuai Wu, Jesse Mu, and Noah D. Goodman. Star: Bootstrapping reasoning with reasoning. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/639a9a172c044fbb64175b5fad42e9a5-Abstract-Conference.html.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *CoRR*, abs/2503.18892, 2025. doi: 10.48550/ARXIV.2503.18892. URL https://doi.org/10.48550/arXiv.2503.18892.

Anqi Zhang, Yulin Chen, Jane Pan, Chen Zhao, Aurojit Panda, Jinyang Li, and He He. Reasoning models know when they're right: Probing hidden states for self-verification. *CoRR*, abs/2504.05419, 2025a. doi: 10.48550/ARXIV.2504.05419. URL https://doi.org/10.48550/arXiv.2504.05419.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics, ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, volume ACL 2025 of *Findings of ACL*, pp. 10495–10516. Association for Computational Linguistics, 2025b. doi: 10.18653/V1/2025.FINDINGS-ACL.547. URL https://doi.org/10.18653/v1/2025.findings-acl.547.

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. *CoRR*, abs/2601.18734, 2026. doi: 10.48550/ARXIV.2601.18734. URL https://doi.org/10.48550/arXiv.2601.18734.

Xudong Zhu, Jiachen Jiang, Mohammad Mahdi Khalili, and Zhihui Zhu. From emergence to control: Probing and modulating self-reflection in language models. *CoRR*, abs/2506.12217, 2025. doi: 10.48550/ARXIV.2506.12217. URL https://doi.org/10.48550/arXiv.2506.12217.

A Extended Results

A.1 Does Recursive Improvement Scale to a Larger Model?

On Qwen3-14B, Average@12 rises from 21.39% at initialization to 42.64% at step 20 and 66.39% at step 30. This 45.00-point gain spans all four benchmarks and is largest on AIME26 (+50.28 points). Step 50 remains comparable at 65.21% but requires 4,173 additional mean tokens, underscoring the value of early stopping on the accuracy–length frontier (table 4).

Table 4: Qwen3-14B recursive self-improvement under non-thinking, 32K, Average@12 evaluation. Cells report accuracy (Acc., %) and mean tokens (Tok.).

Step	**AIME24**		**AIME25**		**AIME26**		**HMMT25**		**Average**	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
Base	30.56	4,056	24.17	3,094	19.44	4,045	11.39	2,998	21.39	3,548
10	36.39	5,044	25.83	4,589	26.39	4,604	15.00	4,102	25.90	4,585
20	55.28	7,377	42.50	7,255	45.28	7,717	27.50	7,502	42.64	7,463
30	**76.11**	14,391	**71.94**	15,973	69.72	14,539	47.78	18,605	**66.39**	15,877
50	71.94	18,111	69.72	19,975	**70.28**	18,792	**48.89**	23,323	65.21	20,050

A.2 Does Recursive Improvement Transfer Beyond Qwen3?

Table 5: **Transfer to Gemma-4-12B-IT.** Cells report Average@12 accuracy (Acc., %) and mean generated tokens (Tok.) under non-thinking, 32K decoding. Blue bold marks the highest accuracy in each column.

Method	AIME24		AIME25		AIME26		HMMT25		Average	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
Base	66.67	6,414	51.67	9,688	63.33	7,605	43.06	10,358	56.18	8,516
SFT	68.33	6,676	55.83	8,872	59.17	8,476	37.22	11,491	55.14	8,878
GRPO	70.83	5,726	51.39	8,849	60.56	8,507	39.72	10,659	55.63	8,435
OPSD	64.72	6,768	46.39	8,865	53.33	7,814	39.72	9,001	51.04	8,112
DCE	71.39	6,501	**65.00**	8,117	65.28	8,724	46.39	11,909	62.01	8,812
+ SRCL	**72.50**	6,661	64.72	8,341	**68.06**	7,965	**49.17**	11,137	**63.61**	8,526

To test whether recursive improvement transfers across model families, we apply our framework to Gemma-4-12B-IT. As shown in table 5, DCE and DCE+SRCL reach 62.01% and 63.61% Average@12, respectively, outperforming all non-DCE comparisons. SRCL adds 1.60 percentage points while reducing mean output length from 8,812 to 8,526 tokens. Across the saved trajectory, DCE+SRCL attains higher accuracy at five of seven checkpoints, ties once, and is both more accurate and shorter than DCE at four checkpoints. The framework therefore transfers beyond Qwen3.

The exact model ID is `google/gemma-4-12B-it`. The matched DCE runs use the same 14,717-problem OPSD shard, assistant-side privileged conditioning, non-thinking student and teacher branches, and 200-step LoRA training with seed 42 (rank 128, alpha 256, global batch 16, and learning rate 2.5×10^{-6}). Their only objective-level difference is $\lambda_S = 0$ for DCE and $\lambda_S = 25$ for DCE+SRCL, with $\lambda_G = 5 \times 10^4$ in both.

Placement also matters on Gemma-4-12B-IT. Assistant-side conditioning reaches 63.61% Average@12, compared with 55.76% for user-side reference-last and 54.31% for user-side instruction-last (table 6). Both user-side variants attain their best observed result at step 10, whereas the assistant-side run continues improving through step 100.

Table 6: Privileged-solution placement on Gemma-4-12B-IT under non-thinking, 32K, Average@12 evaluation. Assistant-side uses checkpoint 100; both user-side variants use checkpoint 10. Each cell reports accuracy (%) / mean generated tokens.

GOLD placement	AIME24	AIME25	AIME26	HMMT25	Average
Assistant-side	**72.50** / 6,661	**64.72** / 8,341	**68.06** / 7,965	**49.17** / 11,137	**63.61** / 8,526
User-side: reference-last	67.22 / 8,242	53.89 / 12,188	61.94 / 8,994	40.00 / 14,066	55.76 / 10,872
User-side: instruction-last	65.00 / 7,692	52.50 / 10,646	60.28 / 9,169	39.44 / 12,462	54.31 / 9,992

Table 7: Matched Gemma-4-12B-IT training trajectories. Accuracy is four-task Average@12 (%), Tok. is mean generated tokens, and Δ is DCE+SRCL minus DCE accuracy in percentage points.

Step	DCE		DCE+SRCL		Δ Acc.
	Acc.	Tok.	Acc.	Tok.	(pp)
10	55.76	8,381	56.39	8,357	+0.63
20	59.17	7,866	57.36	8,111	−1.81
30	59.79	7,691	59.79	8,424	0.00
50	61.04	7,701	62.85	7,328	+1.81
100	**62.01**	8,812	**63.61**	8,526	+1.60
150	61.04	10,015	62.08	9,457	+1.04
200	60.76	9,968	61.46	10,439	+0.69

A.3 How Do Different Loss Weights Affect Performance?

To explore the distinct roles of the two training objectives, we vary their coefficients separately on Qwen3-8B. The DCE weight λ_G controls the strength of privileged next-token guidance, whereas the SRCL weight λ_S controls the contribution of accepted concise rewrites. In each sweep, the other coefficient and all remaining training and evaluation settings are fixed. Table 8 reports the resulting task accuracy and output length for each setting.

Table 8: Sensitivity to the two training-loss weights on Qwen3-8B. The upper block varies the DCE guidance weight λ_G; the lower block varies the SRCL weight λ_S, with the other coefficient fixed to its main setting. Results use non-thinking, 32K, Average@12 evaluation. Accuracy (Acc., %) and mean generated tokens (Tok.) are reported separately. Blue bold marks the highest accuracy within each block; pale teal marks the main settings.

Weight	AIME24		AIME25		AIME26		HMMT25		Average	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
DCE guidance weight λ_G										
1×10^4	**76.67**	16,475	66.94	18,539	71.39	17,108	**47.22**	22,708	65.56	18,708
2.5×10^4	74.17	16,310	69.72	17,739	70.83	17,006	45.56	21,784	65.07	18,210
5×10^4	73.89	15,640	**71.94**	17,241	**71.94**	16,158	46.11	21,205	**65.97**	17,561
1×10^5	73.89	17,220	66.94	19,219	71.11	17,968	43.89	22,837	63.96	19,311
SRCL weight λ_S										
12.5	74.17	18,496	67.50	20,167	72.50	19,109	45.83	24,069	65.00	20,460
18.75	71.67	17,805	66.39	19,172	**73.33**	18,034	44.44	23,020	63.96	19,508
25	73.89	15,640	**71.94**	17,241	71.94	16,158	46.11	21,205	**65.97**	17,561
30	72.78	18,004	68.61	19,436	72.50	18,544	**47.22**	23,358	65.28	19,836
32.5	**75.56**	16,524	63.89	18,977	68.33	17,914	45.83	22,424	63.40	18,960
50	74.44	16,874	68.33	19,020	70.00	18,369	46.39	22,932	64.79	19,299
500	74.17	15,039	63.61	17,293	71.94	15,412	43.89	20,952	63.40	17,174

Both sweeps identify a broad but non-monotonic operating region. Across the tested λ_G range, Average accuracy varies only from 63.96% to 65.97%; $\lambda_G = 5 \times 10^4$ gives both the highest Average and the lowest mean length, although 1×10^4 is stronger on AIME24 and HMMT25. For SRCL, $\lambda_S = 25$ likewise gives the highest Average, 65.97%, while the other settings remain within 2.57 points despite spanning a forty-fold range. These results indicate local robustness, not a monotonic or scale-independent optimum; the corresponding 4B sweep is reported in table 27. They also clarify that the objectives are complementary: when DCE is removed, SRCL alone collapses to 2.36%/1,116 tokens at 8B and 0.76%/531 tokens at 4B. Concise self-refinement therefore improves the frontier only when paired with dynamic guidance that develops the underlying revision capability.

A.4 How Does the Guidance Objective Influence Performance?

To examine how the direction of distillation affects recursive improvement, we compare the best observed Qwen3-8B DCE+SRCL runs using Forward KL $D_{\mathrm{KL}}(q\|p)$, Reverse KL $D_{\mathrm{KL}}(p\|q)$, and JSD $\frac{1}{2}D_{\mathrm{KL}}(q\|m) + \frac{1}{2}D_{\mathrm{KL}}(p\|m)$, where $m = (q+p)/2$, under the same evaluation protocol.

The results follow the asymmetry of the three objectives. Forward KL weights each discrepancy by the teacher distribution, so a teacher-preferred correction still receives a strong learning signal when the student assigns it little probability. Reverse KL instead weights the mismatch by the student distribution; corrections that the student rarely considers contribute less, favoring its existing modes over missing teacher-supported alternatives. JSD is symmetric and bounded, which can further weaken directional transfer when the two distributions differ substantially. Consistent with this interpretation, Forward KL reaches 65.97% with 17,561 tokens, whereas Reverse KL is 5.55 percentage points lower while using 6,187 more tokens. JSD reaches only 17.99% with 3,767 tokens, suggesting premature shortening rather than useful concision. Forward KL is therefore the most effective of the tested objectives for transferring privileged revision guidance.

Table 9: Effect of the distillation objective on Qwen3-8B DCE+SRCL under non-thinking, 32K, Average@12 evaluation. Accuracy (Acc., %) and mean generated tokens (Tok.) are reported separately.

Objective	AIME24		AIME25		AIME26		HMMT25		Average	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
Forward KL	**73.89**	15,640	**71.94**	17,241	**71.94**	16,158	**46.11**	21,205	**65.97**	17,561
Reverse KL	69.44	21,818	63.61	23,800	67.50	22,574	41.11	26,801	60.42	23,748
JSD	25.83	4,776	18.33	3,014	16.39	4,453	11.39	2,823	17.99	3,767

A.5 How Does the Method Perform under Tight Reasoning Budgets?

One important deployment question is whether the learned revision behavior remains useful when the reasoning budget is tight. We therefore evaluate Qwen3-8B at total output budgets of 8K and 16K using three inference-only controls. *Truncate* directly cuts the response generated under the original 32K setting at the target budget. *Cue* first generates 7K tokens for an 8K budget (14K for 16K), appends a short instruction to continue reasoning, and generates the remainder within the same total cap. *TTS* adapts the test-time-scaling continuation procedure of Ghosal et al. (2025): when an OPSD response ends early, it appends a `Wait` cue and continues until the output reaches the exact target length. These controls modify decoding only; model parameters remain fixed.

Table 3 shows that simply forcing more tokens is ineffective: at 8K, TTS lowers OPSD from 29.17% to 28.47% while nearly doubling output length, and at 16K it gains only 0.48 points while adding 11,275 tokens. Cue, by contrast, improves every matched DCE setting while slightly reducing mean output. At 8K, DCE rises from 24.38% to 31.04% and DCE+SRCL from 28.54% to 35.07%; at 16K, the corresponding gains are 48.68% to 55.76% and 52.01% to 57.64%. Thus the learned revision behavior remains useful under tight budgets, whereas length alone does not explain the gain. Complete TTS results across model sizes appear in table 24.

A.6 How Sensitive Is Performance to Decoding Hyperparameters?

Finally, we test sensitivity to two common decoding choices under the non-thinking, 32K, Average@12 protocol. Table 10 shows that the default $T = 1.0$ gives the best Average among the tested temperatures (65.97%/17,561 tokens), while $\rho = 1.08$ is locally best among the tested repetition penalties (67.15%/17,173), gaining 1.18 points with 388 fewer tokens. We retain default decoding for the main comparisons and report the repetition-penalty result as a decoding-sensitivity analysis.

B Complete Qwen3-8B and Qwen3-4B Trajectories

Tables 11 and 12 report checkpoint-wise Average@12 accuracy and mean output length for Qwen3-8B and Qwen3-4B, complementing the learning curves in figure 3. Missing entries denote unevaluated checkpoints. The GRPO columns show the available trajectories, while table 1 reports the tuned baseline comparison.

Table 10: Decoding sensitivity of the Qwen3-8B DCE+SRCL checkpoint. We vary temperature with $\rho = 1.0$ and repetition penalty with $T = 1.0$, holding all other decoding settings fixed. Each benchmark reports accuracy (Acc., %) and mean output tokens (Tok.); pale blue marks the default setting and blue bold the highest accuracy in each row.

(a) Temperature sweep ($\rho = 1.0$)

Benchmark	$T = 0.6$		$T = 0.8$		$T = 1.0$		$T = 1.2$	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
AIME24	**75.28**	15,155	74.17	15,485	73.89	15,640	74.72	15,325
AIME25	69.44	17,530	67.78	17,990	**71.94**	17,241	71.67	17,342
AIME26	68.33	16,883	68.61	16,584	**71.94**	16,158	70.28	16,462
HMMT25	44.72	21,723	**46.67**	21,522	46.11	21,205	45.00	21,395
Average	64.44	17,823	64.31	17,895	**65.97**	17,561	65.42	17,631

(b) Repetition-penalty sweep ($T = 1.0$)

Benchmark	$\rho = 1.00$		$\rho = 1.02$		$\rho = 1.08$		$\rho = 1.10$	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
AIME24	73.89	15,640	**78.06**	15,386	77.50	15,317	75.56	14,876
AIME25	**71.94**	17,241	68.61	17,701	70.00	16,909	71.39	16,779
AIME26	71.94	16,158	72.78	16,591	**73.33**	15,813	71.39	16,139
HMMT25	46.11	21,205	47.50	21,390	**47.78**	20,652	47.22	20,962
Average	65.97	17,561	66.74	17,767	**67.15**	17,173	66.39	17,189

Table 11: Qwen3-8B checkpoint trajectories. Adjacent columns report Average@12 accuracy (Acc., %) and mean generated tokens (Tok.); blue bold marks the highest accuracy for each method. The base model obtains 19.51% with 3,922 tokens.

Step	GRPO		OPSD		DCE		+ SRCL	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
10	18.47	3,860	18.89	3,942	22.85	5,031	23.54	5,016
20	19.72	3,942	20.28	4,684	32.36	7,168	33.75	7,064
30	20.21	3,914	21.32	5,240	45.49	11,141	46.67	10,759
50	19.51	3,774	28.89	6,580	64.44	18,728	61.32	15,959
100	19.86	3,841	**30.35**	6,009	64.93	19,036	**65.97**	17,561
150	**20.28**	3,705	29.65	5,975	**65.76**	19,046	65.14	18,057
200	20.21	3,827	28.06	5,506	64.31	19,072	64.58	18,225

Table 12: Qwen3-4B checkpoint trajectories, using the same convention as table 11. The base model obtains 17.57% with 3,805 tokens.

Step	GRPO		OPSD		DCE		+ SRCL	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
10	17.08	3,622	18.96	4,078	22.99	4,792	22.43	4,912
20	**18.33**	3,653	19.65	4,726	30.00	7,723	30.21	7,388
30	17.43	3,677	20.49	6,136	54.38	16,975	53.82	15,096
50	17.99	3,478	18.26	7,676	**60.00**	19,360	57.99	15,044
100	17.99	3,479	**22.85**	8,350	56.04	22,423	60.83	17,152
150	17.99	3,444	22.36	8,594	55.56	22,057	60.63	17,463
200	17.85	3,428	22.50	8,537	49.58	24,701	**61.88**	17,265

SRCL without DCE collapses to short, mostly incorrect responses: Qwen3-8B reaches at most 2.36% with 1,116 tokens, and Qwen3-4B reaches 0.76% with 531 tokens. SRCL therefore acts as a concision objective only when paired with dynamic guidance.

C TEACHER-UPDATE SCHEDULES

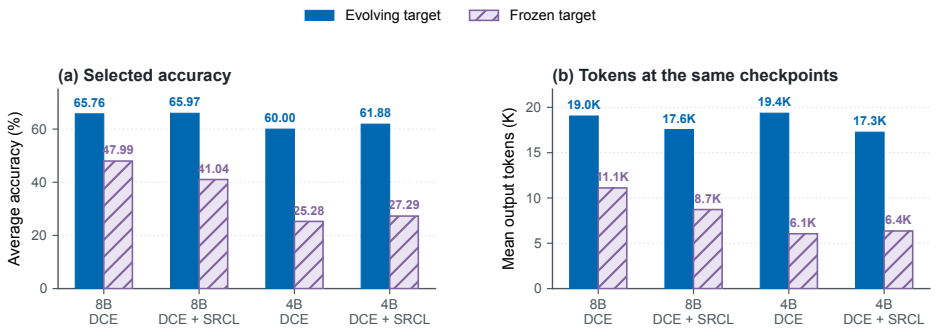


Figure 5: Effect of teacher co-evolution at 8B and 4B. Each pair compares a co-evolving teacher with its frozen counterpart for DCE and DCE+SRCL. Bars report four-task Average accuracy and mean generated tokens; complete trajectories appear in figure 6.

We isolate the effect of teacher refresh by comparing frozen, exponential-moving-average (EMA), periodic-snapshot, and fully dynamic teachers under the same Qwen3-8B setup. For EMA, the teacher after optimizer step t is

$$\theta_T^{(t)} = d\,\theta_T^{(t-1)} + (1-d)\,\theta_S^{(t)}, \tag{7}$$

where a smaller decay d follows the student more closely. Periodic refresh instead copies the current student into the teacher every m optimizer steps and keeps the teacher fixed between updates. The frozen teacher is never refreshed, while the dynamic teacher is synchronized at every step. Table 13 summarizes task-level performance, while tables 14 and 15 give the complete Average@12 trajectories.

Table 13: Task-level teacher-refresh comparison. Results are shown at the listed checkpoint; Qwen3-8B includes frozen, EMA, and fully dynamic teachers, while Qwen3-4B compares frozen and fully dynamic updates. Accuracy (Acc., %) and mean generated tokens (Tok.) are reported separately.

Teacher	Step	AIME24		AIME25		AIME26		HMMT25		Average	
		Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
Qwen3–8B											
Frozen θ_0	50	53.06	8,600	42.22	8,538	44.72	8,899	24.17	8,856	41.04	8,723
EMA $d = 0.3$	150	73.33	17,984	69.72	19,365	69.17	18,341	**49.17**	23,276	65.35	19,741
EMA $d = 0.5$	100	71.11	18,634	68.06	19,921	**72.50**	18,647	46.39	23,359	64.51	20,140
EMA $d = 0.7$	100	**73.89**	16,264	67.22	18,567	70.83	17,089	48.06	21,728	65.00	18,412
Dynamic θ_k	100	**73.89**	15,640	**71.94**	17,241	71.94	16,158	46.11	21,205	**65.97**	17,561
Qwen3–4B											
Frozen θ_0	30	36.67	7,496	27.22	5,869	26.67	7,194	18.61	4,867	27.29	6,357
Evolving θ_k	200	**74.17**	15,273	**59.44**	17,466	**69.72**	16,181	**44.17**	20,143	**61.88**	17,265

Table 14: Qwen3-8B trajectories with exponential-moving-average teachers. Each pair reports four-benchmark Average@12 accuracy (Acc., %) and mean generated tokens (Tok.).

Step	**EMA** $d = 0.3$		**EMA** $d = 0.5$		**EMA** $d = 0.7$	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
10	23.06	4,916	24.31	4,772	22.50	4,603
20	32.71	6,972	31.18	7,202	31.88	7,051
30	45.69	11,164	46.53	10,590	41.67	9,589
50	62.78	18,760	63.33	18,890	60.69	16,613
100	64.17	19,729	**64.51**	20,140	**65.00**	18,412
150	**65.35**	19,741	62.29	21,161	62.99	18,766
200	63.75	20,002	64.24	21,314	62.71	18,597

Table 15: Qwen3-8B trajectories with periodic hard-snapshot teachers. Interval m copies the current student into the teacher every m optimizer steps and keeps it fixed between updates.

Step	**Interval 10**		**Interval 20**		**Interval 30**	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
10	24.17	4,609	22.99	4,802	23.40	4,747
20	31.60	6,864	31.94	6,923	31.39	6,475
30	43.54	10,023	42.57	9,342	41.39	8,931
50	60.76	16,031	57.57	15,867	49.17	11,195
100	**63.47**	18,355	62.43	17,597	56.46	13,178
150	62.50	19,469	63.13	16,878	59.44	15,582
200	62.36	19,499	**63.26**	16,856	**60.00**	16,026

Every refresh strategy substantially improves over the frozen teacher. EMA $d = 0.3$ reaches the highest EMA accuracy at 65.35%, while $d = 0.7$ offers the better accuracy–length balance at 65.00% with 18,412 tokens. Periodic snapshots are weaker, reaching at most 63.47%. Updating the teacher every step performs best overall at 65.97% with 17,561 tokens. Because each schedule is represented by one training trajectory, small differences should not be overinterpreted.

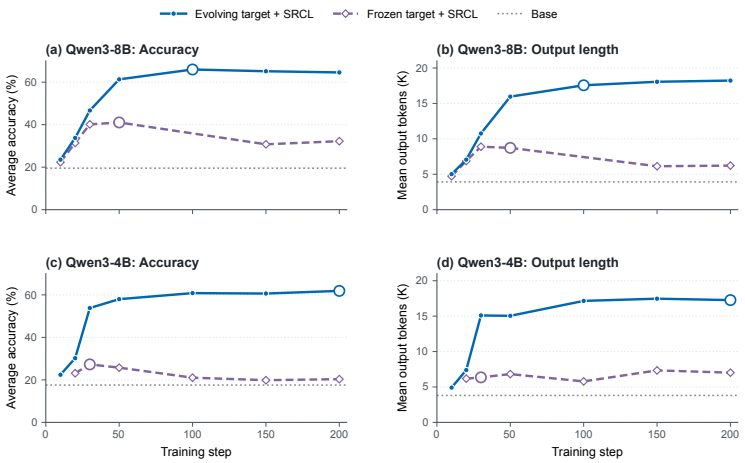


Figure 6: Frozen and fully dynamic DCE+SRCL trajectories at 8B (top) and 4B (bottom). Solid blue refreshes the privileged teacher every step, dashed purple keeps it frozen, and the dotted line denotes the base model.

D Effect of Privileged-Context Placement

The placement ablation separates teacher refresh from the representation of the verified solution. Table 16 reports the complete Qwen3-8B comparison across the three prompt formats illustrated in figure 7. Every dynamic variant outperforms its frozen counterpart. With a dynamic teacher, assistant-side prefill is strongest, reference-last is intermediate, and instruction-last is weakest. Under a frozen teacher, reference-last performs best but remains far below the dynamic variants.

Table 16: Teacher refresh and verified-solution placement on Qwen3-8B. Each cell reports Average@12 accuracy (%) / mean generated tokens under non-thinking, 32K evaluation. The three prompt formats are illustrated in figure 7.

Teacher	GOLD placement	AIME24	AIME25	AIME26	HMMT25	Average
Dynamic	**Assistant-side**	**73.89** / 15,640	**71.94** / 17,241	**71.94** / 16,158	**46.11** / 21,205	**65.97** / 17,561
Dynamic	Reference-last	71.94 / 15,685	67.50 / 17,706	66.94 / 16,737	40.56 / 21,645	61.74 / 17,944
Dynamic	Instruction-last	66.11 / 16,982	61.94 / 18,423	62.50 / 16,336	34.17 / 21,625	56.18 / 18,341
Frozen	Assistant-side	53.06 / 8,600	42.22 / 8,538	44.72 / 8,899	24.17 / 8,856	41.04 / 8,723
Frozen	Reference-last	57.22 / 9,512	43.33 / 9,319	50.28 / 10,087	26.11 / 10,119	44.24 / 9,759
Frozen	Instruction-last	45.83 / 7,394	30.83 / 7,015	33.89 / 7,658	19.17 / 6,223	32.43 / 7,072

Table 17: Task-level privileged-solution placement across model scales under non-thinking, 32K, Average@12 evaluation. Only prompt serialization changes within each model; cells report accuracy (Acc., %) and mean generated tokens (Tok.).

Representation	AIME24		AIME25		AIME26		HMMT25		Average	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
Qwen3–8B										
Assistant-side prefill	**73.89**	15,640	**71.94**	17,241	**71.94**	16,158	**46.11**	21,205	**65.97**	17,561
User-side instruction-last	66.11	16,982	61.94	18,423	62.50	16,336	34.17	21,625	56.18	18,341
Qwen3–4B										
Assistant-side prefill	**74.17**	15,273	**59.44**	17,466	**69.72**	16,181	**44.17**	20,143	**61.88**	17,265
User-side instruction-last	65.83	13,894	57.22	15,498	59.17	13,156	39.72	17,741	55.49	15,072
Qwen3–1.7B										
Assistant-side prefill	**33.61**	19,849	**28.33**	18,396	**28.89**	19,725	**16.67**	20,046	**26.88**	19,504
User-side instruction-last	15.00	7,274	13.33	5,594	11.94	7,797	8.61	5,040	12.22	6,426

After generating Y_0 without g, we hold it fixed. At position t, both branches score the same $Y_{0,<t}$; only the privileged branch also receives g, and neither observes future tokens.

Table 18: Position-resolved signal on a common set of fixed, correct Qwen3-8B responses. Each cell is $D_{\mathrm{KL}}(q_T \| p_S)/[-\log q_T(Y_{0,t})]$. KL measures teacher–student distribution shift; NLL measures teacher support for the observed token. Larger KL means more shift, while lower NLL means more support, not greater correctness. Each serialization column comes from its own trained model rather than from rescoring one model under three prompt formats.

Position in Y_0	Assistant-side	Reference-last	Instruction-last
1–32	0.8427 / 0.6425	0.4787 / 0.5150	0.4648 / 0.4857
33–128	0.4092 / 0.4324	0.4177 / 0.4483	0.4341 / 0.4382
129–512	0.2667 / 0.2806	0.4198 / 0.3957	0.4277 / 0.3849
513–1024	0.1724 / 0.1945	0.2761 / 0.2638	0.2863 / 0.2610

Table 18 shows a front-loaded, not uniformly stronger, intervention. At positions 1–32, assistant-side has the highest KL/NLL, indicating the largest distribution shift and less probability on the

observed Y_0 token. After position 32, both metrics are lowest, indicating agreement with the fixed continuation rather than correctness. This pattern is consistent with route initialization, while the frozen reversal in table 16 shows that placement alone is insufficient.

Evolution of the privileged signal. We next track the same diagnostic over training. At each checkpoint, KL compares the verified-solution-conditioned teacher distribution q_T with the no-GOLD student distribution p_S on the same fixed correct trajectories. KL records how strongly the privileged context changes the prediction, but not whether that change favors the observed correct continuation. We therefore also report

$$\Delta\text{NLL} = \text{NLL}_{\text{no-GOLD}} - \text{NLL}_{\text{with-GOLD}}, \tag{8}$$

where a positive value means that privileged conditioning assigns greater likelihood to the fixed correct continuation.

Table 19: Training dynamics of privileged conditioning on Qwen3-8B. KL measures the magnitude of the teacher's intervention, whereas ΔNLL measures whether that intervention increases likelihood on the fixed correct continuation. Each cell reports KL / ΔNLL. Each serialization is a separately trained model, so the curves provide training-dynamics evidence rather than an isolated causal effect of prompt position.

Step	Assistant-side	Reference-last	Instruction-last
10	0.1303 / +0.0414	0.0655 / −0.0010	0.0663 / −0.0039
20	0.1888 / +0.1364	0.1005 / +0.0391	0.1023 / +0.0323
30	0.2556 / +0.2474	0.1495 / +0.0713	0.1466 / +0.0506
50	0.3646 / +0.3729	0.2387 / +0.2209	0.2178 / +0.1492
100	0.4038 / +0.4139	0.2207 / +0.0266	0.1947 / +0.0026
150	0.4367 / +0.6053	0.1973 / −0.0358	0.2151 / +0.1201
200	0.4371 / +0.6474	0.2140 / −0.0228	0.2122 / +0.0990

The two metrics separate intervention strength from direction: KL asks how much GOLD changes the prediction, while ΔNLL asks whether that change favors the recorded correct continuation. For assistant-side conditioning, both signals increase throughout training: KL rises monotonically from 0.1303 to 0.4371, and ΔNLL rises from $+0.0414$ to $+0.6474$. Assistant-side also has the largest ΔNLL at every checkpoint, with its advantage widening later in training. The two user-side runs strengthen through step 50 but then saturate or regress. From step 50 to 100, ΔNLL increases from $+0.3729$ to $+0.4139$ for assistant-side, but falls from $+0.2209$ to $+0.0266$ for reference-last and from $+0.1492$ to $+0.0026$ for instruction-last. At step 200, the assistant-side improvement exceeds reference-last and instruction-last by 0.6702 and 0.5484, respectively. Thus, the user-side contexts can continue to alter the output distribution without reliably increasing likelihood on the recorded correct path.

These curves show three training trajectories progressively separating; they do not by themselves identify prompt position as the cause because each column comes from a separately trained model. Nor is larger KL synonymous with higher benchmark accuracy: assistant-side accuracy peaks before its KL does. Finally, ΔNLL measures alignment with the fixed correct trajectories used in this probe, not final-answer accuracy or support for every valid derivation.

E Qwen3-1.7B Training Stability

At 1.7B, both recursive variants improve rapidly but become unstable later in training: DCE peaks at step 30, while DCE+SRCL peaks at step 50 before declining. Table 20 reports the complete trajectory and highlights the importance of early stopping at this scale.

Table 20: Qwen3-1.7B checkpoint trajectories under non-thinking, 32K, Average@12 evaluation. Adjacent columns report accuracy (Acc., %) and mean generated tokens (Tok.); blue bold marks the highest accuracy for each method.

Step	OPSD		GRPO		DCE		+ SRCL	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
10	9.72	3,359	9.10	3,077	11.46	4,099	12.22	4,266
20	9.79	4,025	9.31	3,122	15.76	6,159	16.46	6,874
30	9.38	4,237	8.89	3,095	**23.33**	12,600	26.32	14,797
50	**10.35**	5,380	9.03	3,151	22.57	18,930	**26.88**	19,504
100	9.58	5,749	**9.58**	2,959	1.53	30,241	25.90	23,412
150	8.47	6,222	8.82	2,916	21.94	29,370	20.35	24,059
200	9.03	6,282	8.89	2,816	16.25	31,033	1.53	32,005

F Fixed-Trace Probe Values

For a stored incorrect response y_i^- with its terminal EOS removed, the student scores the problem and full response, while the teacher scores the same response with the verified solution in its preceding assistant context. At an observed revision point t, both branches instead score the earlier response prefix and its actual next cue $r_{i,t}$, such as `Wait`. Let c^{end} and c^{ref} denote these endpoint and pre-revision contexts. For branch $b \in \{S, T\}$, we compute

$$P_b^{(k)}(\text{EOS}) = \frac{1}{|\mathcal{E}|} \sum_{i \in \mathcal{E}} p_{\theta_k}(\text{EOS} \mid c_{i,b}^{\text{end}}), \qquad P_b^{(k)}(r) = \frac{1}{|\mathcal{W}|} \sum_{(i,t) \in \mathcal{W}} p_{\theta_k}(r_{i,t} \mid c_{i,t,b}^{\text{ref}}). \tag{9}$$

Both quantities come directly from the full next-token distribution; the second scores the cue observed in the stored trajectory rather than summing over a hand-built reflection lexicon.

Figure 4 summarizes these probabilities on fixed cohorts from AIME24, AIME25, and AIME26. The cohorts contain 227, 212, and 245 incorrect responses and 879, 805, and 583 observed revision events, respectively. Table 21 reports the complete AIME26 values, including EOS probability at the same pre-reflection positions.

Table 21: AIME26 next-token probabilities on 245 fixed incorrect responses containing 583 observed reflection events. Student and teacher probabilities are reported separately; pre-reflection EOS values are shown in units of 10^{-12}.

Checkpoint	Wrong-response endpoint		Before observed reflection cue			
	$p(\text{EOS})$ (%)		$p(r_t)$ (%)		$p(\text{EOS})$ (10^{-12})	
	Student	Teacher	Student	Teacher	Student	Teacher
Base	88.71	85.57	29.12	37.22	26.5	35.3
s10	86.28	83.44	41.11	48.00	10.3	15.8
s20	80.79	79.01	59.98	64.04	1.14	2.67
s30	68.73	68.72	73.83	75.93	0.114	0.378
s50	54.52	57.40	80.99	81.33	0.299	0.285
s100	33.10	37.24	81.13	81.35	0.984	0.985
s150	26.28	35.81	80.39	81.02	0.866	1.02
s200	22.75	34.87	79.93	80.84	0.589	1.04

G EOS-Penalty Ablation

Objective. To test whether concision can be induced directly, we add an EOS-specific auxiliary loss rather than learning from SRCL rewrites. A verified response is truncated after its final balanced boxed expression and retained as y_i^{cut} only if the answer judge still accepts it. Each eligible response

receives one EOS target:

$$\mathcal{L}_{\mathrm{EOS}}^{(k)} = \begin{cases} \dfrac{1}{N_{\mathrm{valid}}} \displaystyle\sum_{i=1}^{N_{\mathrm{valid}}} \left[1 - p_{\theta_k}\left(\mathrm{EOS} \mid x_i, y_i^{\mathrm{cut}}\right)\right], & N_{\mathrm{valid}} > 0, \\ 0, & N_{\mathrm{valid}} = 0. \end{cases} \tag{10}$$

The loss is averaged over eligible responses and set to zero when none are available. We evaluate it as an alternative to SRCL rather than as part of the main method.

Table 22: Task-level EOS-Penalty results under non-thinking, 32K, Average@12 evaluation. Accuracy (Acc., %) and mean generated tokens (Tok.) are reported separately.

Model	Step	AIME24		AIME25		AIME26		HMMT25		Average	
		Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
Qwen3–8B	200	77.50	13,688	66.11	16,052	63.06	15,059	37.78	19,469	61.11	16,067
Qwen3–4B	150	42.78	8,160	32.78	6,846	35.00	6,895	20.28	6,774	32.71	7,169

Table 23: Average@12 accuracy and mean output length across EOS-Penalty checkpoints.

Step	Qwen3–8B		Qwen3–4B	
	Acc.	Tok.	Acc.	Tok.
s50	44.93	9,464	29.51	6,042
s100	58.19	14,273	29.51	6,185
s150	60.14	15,631	32.71	7,169
s200	61.11	16,067	32.57	6,927

Results. At 8B, EOS Penalty is 2,979 tokens shorter than DCE but 4.65 points less accurate and remains below DCE+SRCL. At 4B, it reaches only 32.71% with 7,169 tokens. Direct termination pressure therefore shortens responses at a substantially larger accuracy cost than learning from concise verified rewrites.

H Cross-Scale Evaluation of OPSD-TTS

Table 24 reports OPSD-TTS at exact 8K and 16K output budgets. Doubling the forced budget changes Average@12 by only 0.07, 1.05, and 2.29 points at 1.7B, 4B, and 8B, respectively, and remains far below DCE. Additional generation alone therefore does not reproduce the benefit of co-evolving training.

Table 24: Task-level OPSD-TTS results at exact 8K and 16K output budgets. Accuracy (Acc., %) and generated tokens (Tok.) are reported separately; blue bold marks the higher Average@12 accuracy.

Model	Budget	AIME24		AIME25		AIME26		HMMT25		Average	
		Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
1.7B	8K	14.17	8,192	8.06	8,192	8.06	8,192	6.39	8,192	9.17	8,192
1.7B	16K	14.72	16,384	7.50	16,384	8.89	16,384	5.83	16,384	**9.24**	16,384
4B	8K	28.06	8,192	22.50	8,192	23.61	8,192	13.61	8,192	21.94	8,192
4B	16K	30.28	16,384	22.78	16,384	25.56	16,384	13.33	16,384	**22.99**	16,384
8B	8K	43.06	8,192	27.50	8,192	28.33	8,192	15.00	8,192	28.47	8,192
8B	16K	46.39	16,384	28.89	16,384	31.11	16,384	16.67	16,384	**30.76**	16,384

Table 25: Task-level Qwen3-8B comparisons under matched 8K and 16K output budgets. Truncate cuts at the target, Cue reserves room for a prompted continuation, and TTS fills the budget. Cells report accuracy (Acc., %) and mean output tokens (Tok.).

Method	Condition	AIME24		AIME25		AIME26		HMMT25		Average	
		Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
8K target											
OPSD	Standard	45.00	4,148	27.50	4,217	28.89	4,500	15.28	4,144	29.17	4,252
OPSD	TTS	43.06	8,192	27.50	8,192	28.33	8,192	15.00	8,192	28.47	8,192
DCE	Truncate	35.56	7,740	23.33	7,736	28.06	7,751	10.56	7,952	24.38	7,795
DCE	Cue	44.72	7,688	30.56	7,695	36.39	7,680	12.50	7,921	31.04	7,746
DCE+SRCL	Truncate	42.50	7,400	29.44	7,493	31.39	7,574	10.83	7,792	28.54	7,565
DCE+SRCL	Cue	47.22	7,339	35.83	7,444	40.83	7,492	16.39	7,727	**35.07**	7,500
16K target											
OPSD	Standard	46.11	4,965	28.61	5,236	29.72	5,374	16.67	4,862	30.28	5,109
OPSD	TTS	46.39	16,384	28.89	16,384	31.11	16,384	16.67	16,384	30.76	16,384
DCE	Truncate	57.78	12,535	51.67	13,298	56.67	12,824	28.61	14,911	48.68	13,392
DCE	Cue	67.22	12,258	54.17	13,110	64.17	12,804	37.50	14,512	55.76	13,171
DCE+SRCL	Truncate	61.67	11,576	54.44	12,605	58.89	12,302	33.06	14,411	52.01	12,724
DCE+SRCL	Cue	70.83	11,424	58.89	12,350	60.83	12,036	40.00	13,817	**57.64**	12,407

I Reproducibility Details

Algorithm 1 Recursive training with DCE and SRCL

Require: Initial checkpoint θ_0; training set $\mathcal{D} = \{(x_i, g_i)\}$; number of rounds K; weights λ_G, λ_S

for $k = 0, 1, \ldots, K-1$ **do**
  Sample a minibatch $\mathcal{B} \subset \mathcal{D}$
  Initialize $\mathcal{B}_{\mathrm{SRCL}} \leftarrow \emptyset$
  for all $(x_i, g_i) \in \mathcal{B}$ **do**
    Sample an on-policy response $y_i^{(k)} \sim p_{\theta_k}(\cdot \mid x_i)$
    Score the response prefixes with the student and detached gold-conditioned privileged teacher
    Generate a self-refinement $c_i^{(k)} = R_{\theta_k}(x_i, y_i^{(k)})$ without access to g_i
    Compute the acceptance indicator $A_i^{(k)}$
    if $A_i^{(k)} = 1$ **then**
      Add $(x_i, c_i^{(k)})$ to $\mathcal{B}_{\mathrm{SRCL}}$
    end if
  end for
  Compute $\mathcal{L}_G^{(k)}$ from the scored on-policy prefixes
  if $\mathcal{B}_{\mathrm{SRCL}} \neq \emptyset$ **then**
    Compute $\mathcal{L}_{\mathrm{SRCL}}^{(k)}$ on $\mathcal{B}_{\mathrm{SRCL}}$
  else
    Set $\mathcal{L}_{\mathrm{SRCL}}^{(k)} \leftarrow 0$
  end if
  Update

$$\theta_{k+1} \leftarrow \mathrm{Update}\Big(\theta_k,\ \lambda_G \mathcal{L}_G^{(k)} + \lambda_S \mathcal{L}_{\mathrm{SRCL}}^{(k)}\Big)$$

  Use θ_{k+1} to initialize both the student and privileged teacher for round $k+1$
end for

Ensure: Updated checkpoint θ_K

Table 26 summarizes the data, optimization, generation, and evaluation settings used throughout the main experiments; each ablation states its deviations explicitly.

Table 26: Training and evaluation configuration used in the main experiments.

Component	Configuration
Models and data	
Model family	Qwen3-1.7B, Qwen3-4B, Qwen3-8B, and Qwen3-14B; cross-family transfer uses Gemma-4-12B-IT (section A.2). Student and privileged branches operate in non-thinking mode.
Training data	The 14,717 mathematical-reasoning problems from OpenThoughts used by OPSD. DCE constructs student and privileged views of each problem; these are two contexts for the same training example, not separate problems.
Baseline setup	SFT and GRPO use the same training problems and boxed-answer prompt; both run in non-thinking mode for 200 optimizer steps, saving at steps $\{10, 20, 30, 50, 100, 150, 200\}$. They use global batch size 16 across eight GPUs and rank-128, scale-256 LoRA with dropout 0.05 on attention and MLP projections; seed 42.
Training	
SFT baseline	Cross-entropy on the complete gold solution followed by EOS, with prompt tokens masked; 12K target cap and 16,384-token sequence limit. We sweep learning rates $\{5\times10^{-7}, 10^{-6}, 2\times10^{-6}\}$, together with the historical 5×10^{-6} run; no teacher, KD loss, or rollout engine is used.
GRPO baseline	Eight generations per problem, two policy iterations, temperature 1.2, and a 12K maximum training-generation length. Training uses a binary final-answer reward, group normalization, $\beta = 0$, and learning rates $\{10^{-6}, 2\times10^{-6}, 5\times10^{-6}\}$. Rollouts use colocated vLLM.
OPSD baseline	Official OPSD implementation with its default hyperparameters (Zhao et al., 2026), using its non-thinking configuration for both the student and the privileged teacher, which stays frozen at the initial checkpoint. It uses the same training problems as the other methods, trains for up to 200 optimizer steps, and is evaluated at the same checkpoints.
Tuned baseline settings	SFT uses $(2\times10^{-6}, s20)$ at 8B, $(5\times10^{-7}, s20)$ at 4B, and $(5\times10^{-7}, s30)$ at 1.7B; GRPO uses $(2\times10^{-6}, s100)$ at 4B and $(2\times10^{-6}, s20)$ at 1.7B.
Privileged context	Current model weights; problem, first 512 reference tokens, transition, and the retained on-policy response prefix. Gradients are stopped through the privileged branch.
Guidance objective	Forward KL for the main experiments; Reverse KL and JSD are evaluated as ablations.
Qwen3 DCE optimization	AdamW, bfloat16, learning rate 5×10^{-6}, gradient clipping at 0.1, batch size 16, and rank-128 LoRA with scale 256 on attention and MLP projections; eight H100 GPUs.
Rollout and guidance	One 12K rollout per example with $T = 1.1$, top-$p = 0.95$, and top-$k = 20$; the retained prefix receives the DCE guidance objective.
SRCL rewrite	Greedy 12K rewrite conditioned on the problem and original response, without the reference solution. Accepted targets must terminate naturally, be shorter, remain self-contained, and pass structural and answer-verification filters.
Qwen3 outer loss weights	Main $\lambda_G = 5 \times 10^4$; $\lambda_S = 12.5, 25, 35$, and 25 for 14B, 8B, 4B, and 1.7B, respectively.
Training horizon	Up to 200 optimizer steps with seed 42; checkpoints are evaluated at steps $\{10, 20, 30, 50, 100, 150, 200\}$ when available.
Evaluation	
Main protocol	Four 30-problem benchmarks, 12 samples per problem (360 generations per dataset; 1,440 total), non-thinking decoding, 32K generation cap, $T = 1.0$, top-$p = 0.8$, top-$k = -1$, and repetition penalty 1.0. Accuracy uses the verified final answer; token counts include generated output only and exclude prompts.
TTS control	OPSD checkpoints evaluated at exact 8K or 16K output budgets; a `Wait` cue continues responses that terminate early. Each benchmark contains 360 generations, and token counts exclude prompts.

J Additional SRCL-Weight Sensitivity

Table 27 extends the SRCL-weight sweep to Qwen3-4B.

Table 27: Qwen3-4B SRCL-weight sensitivity under non-thinking, 32K, Average@12 evaluation. Accuracy (Acc., %) and mean generated tokens (Tok.) are reported separately; pale teal marks the main setting.

λ_S	AIME24		AIME25		AIME26		HMMT25		Average	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
25	65.83	18,888	58.33	20,431	67.78	18,758	43.33	23,574	58.82	20,413
30	67.22	18,254	56.94	20,374	68.33	18,132	39.72	22,597	58.06	19,839
32.5	65.28	18,990	58.33	20,562	69.17	18,513	42.22	23,556	58.75	20,405
35	**74.17**	15,273	**59.44**	17,466	**69.72**	16,181	**44.17**	20,143	**61.88**	17,265
37.5	68.06	17,344	57.22	18,789	67.22	17,437	38.89	21,279	57.85	18,712
50	66.11	16,556	57.78	17,735	64.44	16,318	39.17	21,490	56.88	18,025

The sweep peaks at $\lambda_S = 35$, reaching 61.88% Average@12 with 17,265 tokens. Relative to $\lambda_S = 25$, it improves accuracy by 3.06 points while using 3,148 fewer tokens; larger weights then reduce accuracy, showing that the balance between guidance and concise-target learning remains important at 4B.

K Results on Additional Benchmarks

We additionally evaluate Qwen3-8B and Qwen3-4B under Average@12 on three benchmarks: MATH-500 (Hendrycks et al., 2021), GPQA-Diamond (Rein et al., 2023), and AMC 2023.

Table 28: Average@12 results on three additional benchmarks. GPQA-D denotes GPQA-Diamond; accuracy (Acc., %) and mean generated tokens (Tok.) are reported separately.

Method	MATH-500		GPQA-D		AMC23	
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
Qwen3–8B						
Base	84.22	1,041	48.61	1,457	70.83	1,998
OPSD	88.58	1,340	50.29	1,877	78.54	2,633
DCE+SRCL	**88.95**	4,043	**58.29**	4,296	**95.42**	7,604
Qwen3–4B						
Base	84.02	969	42.59	1,375	69.38	1,660
OPSD	85.45	1,553	38.97	3,047	70.83	3,433
DCE+SRCL	**87.85**	4,735	**47.47**	2,740	**95.83**	7,589

DCE+SRCL improves over OPSD on all three benchmarks at both scales, with the largest gains on AMC23. The improved accuracy generally accompanies longer outputs on these broader tasks, unlike the concision gains observed on the main benchmark suite.

L SRCL Rewrite Filtering and Acceptance

SRCL learns only from model-generated rewrites that satisfy its concision and correctness criteria. We audit all 3,200 candidates generated during the Qwen3-8B DCE+SRCL run in table 29. The acceptance rate remains stable between 70.63% and 73.13% across four consecutive 50-step windows, with 2,296 rewrites (71.75%) retained overall. Accepted targets are 83.74% shorter than their source rollouts, whose mean length is 1,997 tokens. Of the 904 rejected candidates, 826 fail a structural check, 76 retain explicit revision language, and two do not terminate naturally. Rejection categories record the first failed gate. A rejected rewrite contributes no SRCL loss, although its original example still receives DCE training.

Table 29: SRCL rewrite acceptance across the Qwen3-8B training trajectory. Rejection columns record the first failed gate; Reflection denotes explicit revision markers such as `Wait`, `Actually`, and `Let me reconsider`.

Steps	Candidates	Accepted	Rate	Structural	Reflection	Non-stop
1–50	800	585	73.13%	191	22	2
51–100	800	576	72.00%	208	16	0
101–150	800	565	70.63%	214	21	0
151–200	800	570	71.25%	213	17	0
All	**3,200**	**2,296**	**71.75%**	**826**	**76**	**2**

Filtering pipeline. A candidate is retained only if it passes all four gates below.

1. **Termination and compression.** The rewrite must terminate naturally, be nonempty, remain within the 12K limit, and contain fewer tokens than its source rollout.

2. **Reflection-free rewriting.** A case-insensitive scan rejects explicit reconsideration, correction, restart, or repeated-verification phrases, including `Wait`, `Actually`, `Correction`, `double-check`, and `start over`.

3. **Answer validity.** The rewrite must contain at least 32 tokens and a balanced boxed answer near the end. The final box must be correct, and no earlier box may contain an incorrect answer.

4. **Self-containment and repetition.** The rewrite may not begin as a continuation fragment, refer to omitted material, or exceed the implementation's repeated-phrase and repeated-line thresholds.

Endpoint verification. The verifier extracts the first balanced boxed answer after the final `</think>` delimiter, when present, and compares it with the gold answer using `math_verify`. If parsing fails, it falls back to case-insensitive exact matching after whitespace removal; ratio notation $a : b$ is normalized to a/b. A separate structural check examines every boxed expression, requiring the final box to be correct and rejecting candidates with an earlier incorrect box.

M PROMPT TEMPLATES

The logical prompt templates are shown below, with model-specific chat-control tokens omitted. All privileged variants in figure 7 use the same problem x, verified solution g, transition text, and on-policy response Y_0; only the message role and ordering change. SRCL uses the separate rewrite request listed afterwards.

Student and inference context

USER `Problem:` x
`Please reason step by step, and put your final answer within \boxed{}.`
ASSISTANT $Y_{0,<t} \longrightarrow$ predict $Y_{0,t}$

Assistant-side

USER `Problem:` x
`Please reason step by step, and put your final answer within \boxed{}.`
ASSISTANT
`Here is a reference solution:` g
`After understanding the reference solution, please try to solve this problem using your own approach below:`
$Y_{0,<t} \longrightarrow$ predict $Y_{0,t}$

User-side: reference-last

USER `Problem:` x
`Please reason step by step, and put your final answer within \boxed{}.`
`Here is a reference solution:` g
`After understanding the reference solution, please try to solve this problem using your own approach below:`
ASSISTANT $Y_{0,<t} \longrightarrow$ predict $Y_{0,t}$

User-side: instruction-last

USER `Problem:` x
`Here is a reference solution:` g
`After understanding the reference solution, please try to solve this problem using your own approach below:`
`Please reason step by step, and put your final answer within \boxed{}.`
ASSISTANT $Y_{0,<t} \longrightarrow$ predict $Y_{0,t}$

Figure 7: Prompt orderings for the placement ablation, with chat-control tokens omitted. Assistant-side places g in prior assistant context; the user-side variants place it before or after the task instruction. Here $Y_{0,<t}$ is the fixed prefix used to score token t.

SRCL then applies the following standalone rewrite request to Y_0; it never contains the verified solution.

SRCL rewrite prompt

```
SOURCE ROLLOUT
{full_rollout}
END SOURCE ROLLOUT

PROBLEM
{problem}
END PROBLEM

Rewrite the source rollout into the shortest direct, self-contained,
correct solution to the problem.  Preserve only reasoning needed to
derive the final answer.  Remove every failed branch, retry, repeated
calculation, and reflection phrase such as Wait, reconsider,
actually, correction, or start over.  Do not mention the source
rollout or omitted text.  Do not add analysis about rewriting.  End
with exactly one final \boxed{...} answer.  Output only the clean
solution.
```