SR-GRPO: REINFORCEMENT LEARNING WITH SELF-REFINEMENT FROM VERIFIABLE REWARDS

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement learning from verifiable rewards (RLVR) with a rule-based binary outcome verifier produces strong chain-of-thought reasoners, yet it rewards only the final answer. The policy never learns to inspect or correct its own intermediate steps, so any error in the reasoning chain propagates to the final answer. Addressing this self-refinement gap faces a dilemma: inference-time prompting methods are deployable but, without ground-truth feedback, often degrade accuracy; training-time approaches do learn to self-refine, but mostly by using a supervision signal stronger than RLVR's binary verifier—teacher correction traces, step-level annotations, or separately trained process reward models. We introduce SR-GRPO (Group Relative Policy Optimization with Self-Refinement), a two-stage RLVR framework in which a single shared policy acts as both solver (generating an initial solution) and refiner (correcting its own mistakes), with both stages trained jointly by GRPO. SR-GRPO uses only the binary verifier RLVR already requires—no external teacher, no separate critic, and no ground-truth answer at refinement. SR-GRPO has three mechanisms: wrong-first sampling keeps self-refinement focused on genuine solver errors; a ground-truth-free refinement prompt makes the refinement stage deployable at inference, since the model never sees the answer when it refines; and self-distilling feedback recycles every verified refinement into the solve stage as additional supervision, so the same policy serves as solver, critic, and teacher within a single training loop. Under a single fixed recipe across three backbone families and two model scales (1.5B and 4B), SR-GRPO matches or surpasses the strongest RL baselines on several math reasoning benchmarks: at 1.5B it matches QwQ-32B with far fewer parameters; at 4B, where vanilla GRPO saturates, SR-GRPO continues to improve, surpassing several open-source 20B+ reasoners. Beyond the math benchmarks, SR-GRPO also improves on out-of-distribution general-reasoning benchmarks (GPQA Diamond, MMLU-Pro) across all three backbone families—evidence that the gains are not math-specific reward hacking. Double-blind LLM judges prefer SR-GRPO's reasoning over strong RL baselines, and qualitative analysis confirms that SR-GRPO genuinely locates and revises erroneous steps during self-refinement. Overall, SR-GRPO turns self-refinement into a training-time self-distillation procedure that adds no inference-time overhead and generalizes across backbone families, model scales, and reasoning domains.

1 INTRODUCTION

Reinforcement learning from verifiable rewards (RLVR) has emerged as the dominant approach for training reasoning language models (DeepSeek-AI, 2025; Shao et al., 2024; OpenAI, 2024; Kimi Team, 2025; Gemini Team, 2025), producing strong chain-of-thought reasoners without human-annotated rationales. However, how to incentivize and utilize the model's capability of self-refinement during RLVR training with a rule-based binary outcome verifier remains under-explored. First, RLVR's reward signal reaches only the final answer; the policy is never explicitly trained to identify—let alone refine—its own intermediate mistakes, so a single flawed step propagates unchecked to the final answer. Second, even when a model is trained to self-refine, its refinement outputs are not fed back to improve the solver itself. As a result, no prior work jointly trains a solver and a refiner that improve each other within a single training loop.

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.

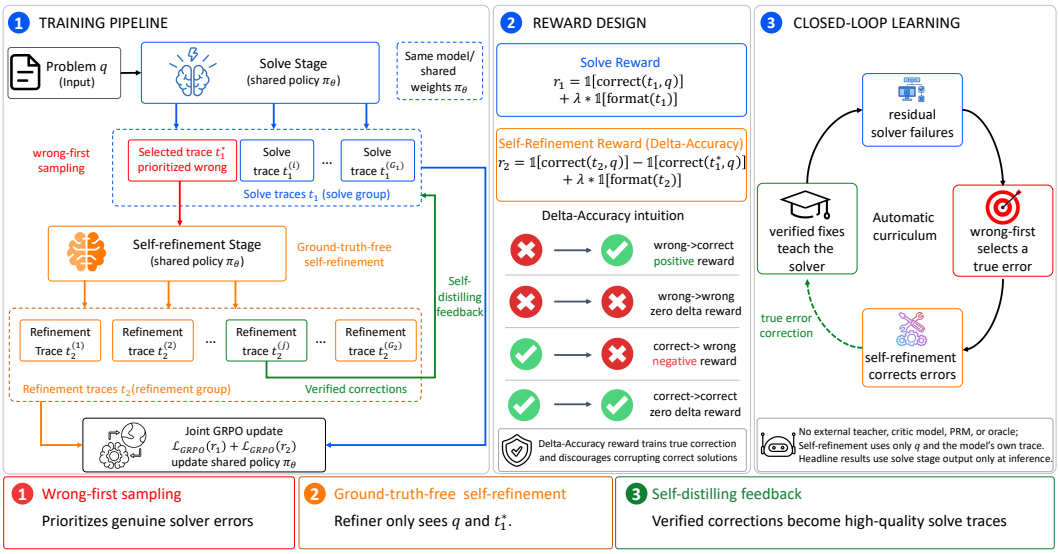

Figure 1: **General framework of SR-GRPO.**

Prior attempts to address these gaps fall into two camps, and neither closes both at once. The first camp does not train self-refinement at all, instead eliciting it through inference-time prompting (Madaan et al., 2023; Shinn et al., 2023; Gou et al., 2024); yet without ground-truth feedback, prompting-based self-correction has been shown to *degrade* accuracy (Huang et al., 2024; Havrilla et al., 2024). The second camp does train self-refinement, but mostly by relying on a supervision signal stronger than RLVR's binary verifier—teacher correction traces, step-level annotations, or separately trained process reward models (Zelikman et al., 2022; Dong et al., 2023; Wan et al., 2025; Qu et al., 2024; Lightman et al., 2024; Wang et al., 2024a; Choi et al., 2025; Shani et al., 2024). Even when self-refinement is trained this way, or from outcome rewards alone as in SCoRe (Kumar et al., 2025), the refinements are discarded rather than recycled into the solver, so the second gap remains open.

We address both gaps with **SR-GRPO**, a two-stage RLVR framework in which a *single shared policy* acts as both solver (generating an initial solution) and refiner (correcting its own mistakes), with both stages trained jointly by GRPO (Shao et al., 2024) (Figure 1). Concretely, SR-GRPO consists of three mechanisms. **Wrong-first sampling** prioritizes an erroneous solve trace as the refinement context, so the refinement gradient is based on a genuine solver error when one exists. A **ground-truth-free refinement prompt** makes self-refinement fully deployable at inference, since the model never sees the answer when it refines. **Self-distilling feedback** recycles every verified refinement into the solve stage, so one shared policy plays all three roles—solver, critic, and teacher—within a single training loop. To isolate gains from training rather than from extra test-time compute, our main results use only first-stage outputs, matching the inference cost of the baselines. Our contributions are:

1. **Joint training of reasoning and self-refinement under pure RLVR.** We introduce **SR-GRPO**, a two-stage RLVR framework that jointly trains one shared policy as solver and refiner with GRPO, using only the binary verifier RLVR already requires: no external teacher, critic, PRM, or ground-truth answer at refinement. Its *wrong-first sampling* rule prioritizes incorrect solve traces as the refinement context and so concentrates training on the cases the solver currently fails on (Table 4).
2. **Self-distilling feedback: the refinement stage as the solver's adaptive teacher.** Every verified correction is recycled into the solve-stage GRPO update, improving the solver itself rather than only the refiner (Table 4). To our knowledge, this is the first closed solve–refine–feedback loop under pure RLVR with a binary verifier.
3. **SOTA math reasoning with strong generalization.** Under one fixed recipe across three backbone families and two scales, SR-GRPO matches or exceeds the strongest GRPO baselines on several math reasoning benchmarks: at 1.5B it sets a new state of the art (SOTA) for its size and matches QwQ-32B, the open-source SOTA, with $\sim$20$\times$ fewer parameters,

and at 4B it surpasses several open-source 20B+ reasoners (Table 1). Despite math-only training, SR-GRPO further improves on both out-of-distribution (OOD) benchmarks (GPQA Diamond, MMLU-Pro) across all three backbones (Table 2).

4. **Improved reasoning quality over the SOTA RL baseline.** Double-blind LLM judges prefer SR-GRPO's reasoning paths over the matched RL baseline across all three backbones on three reasoning-dense math benchmarks; qualitative analysis confirms specific error revisions (Fig. 2, Table 3).

2 Related Work

RL for Reasoning LLMs. Reinforcement learning from verifiable rewards (RLVR) has become the dominant paradigm for training open-source reasoning models (DeepSeek-AI, 2025; OpenAI, 2024; Kimi Team, 2025; Yang et al., 2024), typically realized via GRPO (Shao et al., 2024) and scaled by recipes such as DAPO (Yu et al., 2025), BroRL (Hu et al., 2025), ProRL (Liu et al., 2025), and JustRL (He et al., 2025), the strongest publicly reported GRPO recipe at the 1.5B scale on competition math. Learned outcome reward models (ORMs) (Cobbe et al., 2021) and process reward models (PRMs) (Lightman et al., 2024; Wang et al., 2024a) score the final answer or intermediate steps at the cost of extra annotation; we use only a rule-based binary outcome verifier. In all of these, the policy is *single-stage*: a rollout produces one chain of thought and one final answer, with no training-time signal for detecting or repairing mid-generation mistakes. SR-GRPO extends each rollout into a two-stage solve-and-self-refine trajectory in which the same shared policy plays both solver and refiner under one joint RLVR objective; we adopt JustRL as our primary GRPO baseline on the two 1.5B backbones we share with it.

Self-Refinement at Inference Time. A large body of work elicits self-refinement at *inference time* through prompting and verbal reflection, without updating the policy weights. Self-Refine (Madaan et al., 2023) cycles generate-critique-rewrite purely at inference; Reflexion (Shinn et al., 2023) uses environment signals; and CRITIC (Gou et al., 2024) relies on external tools. Yet careful studies show that without ground-truth feedback, intrinsic self-correction *degrades* rather than improves accuracy (Huang et al., 2024; Havrilla et al., 2024). SR-GRPO instead trains the refinement capability directly into the policy *weights* via RL, so the deployed model self-refines without prompting protocols or external feedback.

Training-Time Self-Refinement. A recent line trains self-refinement into the model itself, but each prior approach either imports supervision stronger than RLVR's binary verifier or runs a multi-turn correction schedule: filter-and-retrain on self-generated samples (Zelikman et al., 2022; Gülçehre et al., 2023; Singh et al., 2024; Dong et al., 2023); SFT of a separate corrector or critic (Welleck et al., 2023; Saunders et al., 2022); online RL with externally generated critiques (closest in spirit to ours, but relying on a GPT-4o critic and ground-truth-conditioned filtering) (Zhang et al., 2025); reflection-augmented RL with teacher SFT warm-up (Wan et al., 2025); separately trained PRMs or auxiliary verifiers (Hosseini et al., 2024; Wang et al., 2024a; Havrilla et al., 2024); and a range of multi-turn or preference-based methods, including reward-weighted regression (Qu et al., 2024), preference optimization (Choi et al., 2025), multi-turn RLHF (Shani et al., 2024), hierarchical agent RL (Zhou et al., 2024), MCTS-guided DPO with self-critique (Putta et al., 2024), and reward-shaped self-correction (SCoRe; Kumar et al., 2025). SCoRe has a different goal: it learns self-correction alone, and its first-attempt accuracy can even fall below the base model. To our knowledge, SR-GRPO is the first method to combine four properties: (i) self-refinement is *trained*, not elicited at inference; (ii) no auxiliary model (teacher, critic, or PRM); (iii) no ground-truth answer at refinement, either at training or inference; and (iv) solver and refiner are jointly optimized under one shared two-stage GRPO objective, with verified refinements recycled into the solver's own training (App. A.1).

3 Method

We propose SR-GRPO, a two-stage RLVR framework in which a *single shared policy* π_θ acts as both *solver* (producing a solve trace) and *refiner* (refining its own attempt). Each stage runs its own GRPO update—its own response group, group-relative advantages, and reward—and at every iteration we take a single gradient step on θ from the sum of the two stages' GRPO losses. After formalizing the

setting (Sec. 3.1), we introduce three mechanisms: a two-stage GRPO update with a ground-truth-free refinement prompt (Sec. 3.2), wrong-first sampling (Sec. 3.3), and self-distilling feedback (Sec. 3.4); Sec. 3.5 assembles the full algorithm and Figure 1 illustrates the loop.

3.1 Problem Formulation

Setup. Let $\mathcal{D} = \{(q_i, y_i^\star)\}_{i=1}^N$ be a dataset of reasoning problems q_i with verifiable ground-truth answers $y_i^\star$. A policy π_θ produces a chain-of-thought trace $t \sim \pi_\theta(\cdot \mid q)$ that terminates in an extractable final answer $y(t)$. A binary verifier $\mathrm{correct}(t, q) = \mathbb{1}[y(t){=}y^\star]$ checks whether the final answer matches the ground truth; a format check $\mathrm{format}(t) \in \{0, 1\}$ verifies that the answer is in a parseable form. The standard RLVR reward is $r(t, q) = \mathrm{correct}(t, q) + \lambda\, \mathrm{format}(t)$, with $\lambda \geq 0$ weighting the format bonus.

Standard RLVR objective. The policy is optimized by

$$J(\theta) \;=\; \mathbb{E}_{q\sim\mathcal{D},\, t\sim\pi_\theta(\cdot|q)}\big[\, r(t, q)\,\big] \;-\; \beta\, D_{\mathrm{KL}}(\pi_\theta \,\|\, \pi_{\mathrm{ref}})\,, \tag{1}$$

typically realized by GRPO (Shao et al., 2024): for each q, sample a group $\{t_i\}_{i=1}^G \sim \pi_\theta(\cdot \mid q)$, compute group-relative advantages

$$A_i = \tfrac{r_i - \mu_r}{\sigma_r}, \quad \mu_r = \tfrac{1}{G}\textstyle\sum_j r_j, \quad \sigma_r = \sqrt{\tfrac{1}{G}\textstyle\sum_j (r_j - \mu_r)^2}, \tag{2}$$

and update with the clipped policy-gradient loss

$$\mathcal{L}_{\mathrm{GRPO}}(\theta;\, r) = -\,\mathbb{E}\left[\frac{1}{G}\sum_{i=1}^{G} \min\big(\rho_i A_i,\; \mathrm{clip}(\rho_i, 1{-}\epsilon, 1{+}\epsilon) A_i\big)\right] + \beta\, D_{\mathrm{KL}}(\pi_\theta \,\|\, \pi_{\mathrm{ref}})\,, \tag{3}$$

where π_{old} is the pre-update policy snapshot used to draw the group, $\rho_i = \frac{\pi_\theta(t_i|q)}{\pi_{\mathrm{old}}(t_i|q)}$ is the importance ratio, and ϵ is the clipping threshold. This removes the need for a learned value function and makes GRPO efficient for reasoning fine-tuning.

The self-refinement gap. The objective in Eq. (1) treats every rollout as a single attempt: once generation begins, an early error propagates unchecked to $y(t)$, and the policy never receives a training signal for detecting or repairing mid-generation mistakes. Our goal is to train a single shared policy π_θ that both solves and refines its own attempts using only the binary verifier $\mathrm{correct}$, with no stronger teacher and no separate critic.

3.2 Two-Stage GRPO with a Shared Policy

We extend the RLVR rollout from a single solution to a *solve-and-self-refine* trajectory. For each problem q, the model samples a group of G_1 solve traces, selects one of them (t_1^*) as the conditioning context via *wrong-first sampling* (Sec. 3.3, which prioritizes an incorrect solve trace so the refiner sees a real error), and then samples a group of G_2 refinement traces conditioned on t_1^*:

$$\{t_1^{(i)}\}_{i=1}^{G_1} \sim \pi_\theta(\cdot \mid q), \qquad \{t_2^{(j)}\}_{j=1}^{G_2} \sim \pi_\theta(\cdot \mid q, t_1^*). \tag{4}$$

The refinement instruction is fixed across all experiments and contains *no ground-truth answer*: each refinement is derived from q and t_1^* alone, so the trained capability is fully usable at deployment (exact text in App. C.4). The binary verifier is used only to compute the RL reward *after* the rollout completes, exactly as in standard RLVR. Each stage receives a per-trace reward matched to its role:

$$r_1 = \mathbb{1}[\mathrm{correct}(t_1, q)] + \lambda\, \mathbb{1}[\mathrm{format}(t_1)], \tag{5}$$

$$r_2 = \mathbb{1}[\mathrm{correct}(t_2, q)] - \mathbb{1}[\mathrm{correct}(t_1^*, q)] + \lambda\, \mathbb{1}[\mathrm{format}(t_2)], \tag{6}$$

where $t_1 \in \{t_1^{(i)}\}_{i=1}^{G_1}$ is a solve trace, $t_2 \in \{t_2^{(j)}\}_{j=1}^{G_2}$ is a self-refined trace, and t_1^* is the wrong-first conditioning trace from Sec. 3.3. The solve reward r_1 is the standard RLVR outcome reward. The refinement reward r_2 is an outcome-accuracy delta plus the same format bonus. The delta separates the two regimes the refiner faces: when t_1^* is wrong, $r_2 \in \{0, +1\}$ rewards only an actual fix; when t_1^* is correct (the rare fallback), $r_2 \in \{-1, 0\}$ penalizes corrupting an already-correct answer.

The two stages share the policy parameters θ but otherwise run as *separate* GRPO procedures: independent sampling, independent group-relative advantages, and independent rewards. At every iteration we take a single gradient step on the sum of the two stages' GRPO losses,

$$\mathcal{L}(\theta) \;=\; \mathcal{L}_{\mathrm{GRPO}}\big(\theta;\, r_1\big) \;+\; \mathcal{L}_{\mathrm{GRPO}}\big(\theta;\, r_2\big), \tag{7}$$

where each $\mathcal{L}_{\mathrm{GRPO}}$ is the standard GRPO loss in Eq. (3) computed over its own response group with its own reward. The refinement-stage group is $\{t_2^{(j)}\}_{j=1}^{G_2}$; the solve-stage group is $\{t_1^{(i)}\}_{i=1}^{G_1}$ augmented by *self-distilling feedback* (Sec. 3.4, which appends every verified refinement to the solve group as an extra correct rollout) before its GRPO loss is computed. Throughout, $G_1 = G_2 = 8$.

3.3 Wrong-First Sampling

Each refinement group is conditioned on a single solve trace t_1^*. Picking t_1^* uniformly from the G_1 solve traces is wasteful: as the solver improves, most randomly chosen traces are already correct, leaving the refiner with no real error to fix. We therefore use **wrong-first sampling**: pick t_1^* uniformly from the incorrect traces in the group whenever any exist, and uniformly from all G_1 traces otherwise (a rare fallback when the solver is at ceiling on q):

$$t_1^* \;\sim\; \mathrm{Uniform}\big(\mathcal{W}\big) \text{ if } \mathcal{W} \neq \varnothing, \quad \text{else} \quad \mathrm{Uniform}\big(\{t_1^{(i)}\}_{i=1}^{G_1}\big), \quad \mathcal{W} = \{\, t_1^{(i)} : \mathrm{correct}(t_1^{(i)}, q) = 0 \,\}. \tag{8}$$

This grounds the entire refinement group in a genuine solver error whenever one exists. In the rare all-correct fallback, the refinement reward of Eq. (6) discourages the refiner from corrupting an already-correct attempt. As the solver improves and failures become rarer, the rule keeps refinement focused on the hardest cases the solver still gets wrong (implementation in App. C.6).

3.4 Self-Distilling Feedback

Whenever the refinement stage succeeds on a problem the solver itself failed, that correction is a verified demonstration of how to solve a problem at the frontier of the solver's current capability, so we recycle every such correction into the solve-stage GRPO update.

Definition 1 (Feedback Condition)**.** *A refinement trace $t_2^{(j)}$ satisfies the* feedback condition *when* $\mathrm{correct}(t_1^*, q) = 0$ *and* $\mathrm{correct}(t_2^{(j)}, q) = 1$*, i.e., it fixes an erroneous solve. On non-fallback rollouts,* $\mathrm{correct}(t_1^*, q) = 0$ *holds by wrong-first construction (Sec. 3.3), so the condition reduces to* $\mathrm{correct}(t_2^{(j)}, q) = 1$*. We call such $t_2^{(j)}$ a* verified refinement*, and let $\mathcal{V}$ denote the set of all verified refinements at the current iteration.*

We append $\mathcal{V}$ to the original solve rollouts to form the *augmented solve group*

$$\mathcal{G}_1^{\mathrm{Aug}} \;=\; \{t_1^{(i)}\}_{i=1}^{G_1} \;\cup\; \mathcal{V}, \qquad |\mathcal{G}_1^{\mathrm{Aug}}| = G_1 + |\mathcal{V}|. \tag{9}$$

The solve-stage GRPO loss in Eq. (7) is then computed over $\mathcal{G}_1^{\mathrm{Aug}}$: the solve reward in Eq. (5) and group-relative advantages in Eq. (2) are evaluated on every trace in $\mathcal{G}_1^{\mathrm{Aug}}$, treating each verified refinement as an additional correct solve. The refinement stage is unchanged: its G_2 rollouts and delta reward r_2 are reused as-is.

Because $\mathcal{V}$ is filtered by the binary verifier, every recycled trace is a known-correct solution, so no external teacher or critic is invoked; and because $\mathcal{V}$ is collected as a by-product of the refinement-stage rollout, self-distilling feedback adds no extra rollouts on top of the two-stage update. The refiner thereby teaches the solver to handle problems the solver itself missed, completing a closed solve–refine–feedback loop in which the same shared policy serves as solver, critic, and teacher within a single training step.

3.5 SR-GRPO Algorithm

Each SR-GRPO iteration samples a solve group $\{t_1^{(i)}\}_{i=1}^{G_1}$, selects the conditioning trace t_1^* by wrong-first sampling (Eq. (8)), samples a refinement group $\{t_2^{(j)}\}_{j=1}^{G_2}$ conditioned on t_1^* and the fixed self-refinement instruction, and appends the verified refinements $\mathcal{V}$ (Def. 1) to the solve rollouts to form $\mathcal{G}_1^{\mathrm{Aug}}$ (Eq. (9)). Rewards (Eqs. (5)–(6)) and advantages (Eq. (2)) are computed independently per stage, and we take a single gradient step on Eq. (7). Full pseudocode is in Algorithm 1 (App. B).

Table 1: **Main results on nine math benchmarks.** Pass@1 accuracy (%) averaged across 4 seeds per benchmark; **Avg.** is the unweighted mean. **Bold**/underline: best/second. Top block: SR-GRPO on three backbones, where the + GRPO row in each block is the matched-data (DAPO-Math-17k) vanilla baseline. Middle block: current open-source state of the art at 1.5B for reference. Bottom block: open-source 20B+ reasoners for scaling context (numbers from Hochlehnert et al. (2025)).

Model	Size	MATH 500	AIME 2024	AIME 2025	AMC 2023	Minerva Math	Olympiad Bench	BRUMO 2025	CMIMC 2025	HMMT 2025	Avg.
Our SR-GRPO vs. vanilla GRPO across three settings: weaker base (DeepSeek), stronger base (Nemotron), and a larger 4B base (Qwen3)											
DeepSeek-R1-Distill-Qwen-1.5B (DeepSeek-AI, 2025)	1.5B	85.6	30.0	22.5	66.9	36.1	46.1	31.7	11.9	14.2	38.3
+ GRPO (JustRL-DeepSeek-1.5B) (He et al., 2025)	1.5B	91.8	52.5	38.3	91.3	51.3	68.0	51.7	25.6	22.5	54.8
+ **SR-GRPO (Ours)**	1.5B	94.3	61.7	55.8	95.6	60.3	72.6	62.5	33.1	34.2	63.3
OpenMath-Nemotron-1.5B (Moshkov et al., 2025)	1.5B	92.8	59.2	48.3	91.3	27.6	72.0	60.8	30.6	30.8	57.0
+ GRPO (JustRL-Nemotron-1.5B) (He et al., 2025)	1.5B	94.5	69.2	63.3	96.3	31.1	76.1	67.5	41.9	40.8	64.5
+ **SR-GRPO (Ours)**	1.5B	97.1	78.3	72.5	98.1	42.3	79.4	70.8	51.3	52.5	71.4
Qwen3-4B-Thinking-2507 (Qwen Team, 2025a)	4B	97.8	78.3	80.8	98.8	69.7	80.8	68.3	51.9	49.2	75.1
+ GRPO (Shao et al., 2024)	4B	97.9	81.7	79.2	**99.4**	69.4	81.5	70.8	54.4	50.0	76.0
+ **SR-GRPO (Ours)**	4B	**98.7**	**89.2**	**82.5**	**99.4**	**72.2**	**85.2**	**79.2**	**58.8**	**55.8**	**80.1**
Current open-source state of the art at 1.5B (built on the same DeepSeek-R1-Distill-Qwen-1.5B and OpenMath-Nemotron-1.5B families as our 1.5B runs)											
DeepScaleR-1.5B (Tan et al., 2026)	1.5B	89.4	41.7	29.2	74.4	40.3	53.0	39.2	20.6	21.7	45.5
STILL-3-1.5B (Chen et al., 2025)	1.5B	84.7	40.8	34.2	73.1	46.9	55.3	36.7	10.6	12.5	43.9
BroRL-1.5B (Hu et al., 2025)	1.5B	92.5	58.3	37.5	88.8	52.4	61.8	41.7	21.3	15.8	52.2
QuestA-Nemotron-1.5B (Li et al., 2026)	1.5B	93.3	71.7	62.5	93.1	32.8	73.3	66.7	41.3	40.8	63.9
For scaling context: open-source large-scale reasoners (20B+; numbers from Hochlehnert et al. (2025), who do not report HMMT/BRUMO/CMIMC 2025)											
DeepSeek-R1-Distill-Qwen-32B (DeepSeek-AI, 2025)	32B	95.1	67.0	55.3	96.8	45.1	73.8	—	—	—	—
INTELLECT-2 (Prime Intellect Team et al., 2025)	32B	97.0	75.2	66.3	96.0	49.8	78.0	—	—	—	—
OpenThinker2-32B (Guha et al., 2025)	32B	96.1	71.3	61.7	95.8	46.6	75.5	—	—	—	—
TinyR1-32B-Preview (Sun et al., 2025)	32B	97.2	73.3	66.0	97.8	48.7	76.3	—	—	—	—
Light-R1-32B-DS (Wen et al., 2025)	32B	97.3	76.7	68.7	97.0	48.2	76.9	—	—	—	—
gpt-oss-20b (medium) (OpenAI, 2025)	20B	96.5	75.7	72.7	97.0	45.1	77.0	—	—	—	—
QwQ-32B (Qwen Team, 2025b)	32B	97.5	76.3	69.0	96.2	49.0	78.1	—	—	—	—

4 Experiments

4.1 Experimental Setup

We train SR-GRPO on DAPO-Math-17k (Yu et al., 2025) with no SFT warm-up, applying the same fixed recipe (single hyperparameter set, identical self-refinement instruction) to three backbones spanning three families and two scales: DeepSeek-R1-Distill-Qwen-1.5B (DeepSeek-AI, 2025), OpenMath-Nemotron-1.5B (Moshkov et al., 2025), and Qwen3-4B-Thinking-2507 (Qwen Team, 2025a). The first two are exactly the starting backbones used by JustRL (He et al., 2025), a public, well-tuned vanilla GRPO recipe trained on the same DAPO-Math-17k data; we use its released checkpoints as the GRPO SOTA baseline at matched backbone and data. We train SR-GRPO for one epoch on DAPO-Math-17k with a per-stage maximum output length of $15{,}360$ tokens (App. C.8). All runs use veRL (Sheng et al., 2025) on $8\times$A6000 Pro. We report pass@1 accuracy averaged across 4 seeds per benchmark (standard errors in App. D.5) on nine math benchmarks (MATH-500 (Lightman et al., 2024), AMC 2023 (Mathematical Association of America, 2023), AIME 2024/2025 (Balunovic et al., 2025), Minerva Math (Lewkowycz et al., 2022), OlympiadBench (He et al., 2024), BRUMO/CMIMC/HMMT 2025 (Balunovic et al., 2025)) and two OOD reasoning benchmarks (GPQA Diamond (Rein et al., 2023), MMLU-Pro (Wang et al., 2024b)). For fairness, we use the same evaluation protocol in Table 1 as the JustRL codebase (He et al., 2025). For more details on hyperparameters, prompts, baselines, and benchmark protocols, see App. C.

4.2 Main Results

Table 1 tests whether SR-GRPO consistently improves over a single-stage GRPO recipe regardless of the backbone's strength or scale; we walk through the three backbones and demonstrate the effectiveness of our method (App. D.1 has more results).

Improving a weaker general-reasoning base (DeepSeek-R1-Distill-Qwen-1.5B). On this relatively weak general-purpose backbone, SR-GRPO improves over the matched vanilla GRPO baseline (JustRL-DeepSeek-1.5B, the strongest public GRPO recipe here) on *every* one of the nine math benchmarks at identical inference cost, with the largest lifts on the harder competitions (AIME 2025, HMMT 2025, BRUMO 2025).

Table 2: **Out-of-distribution generalization to science and general reasoning.** Pass@1 accuracy (%) on GPQA Diamond and MMLU-Pro, averaged across 4 seeds per benchmark; SR-GRPO is trained on math only. Within each backbone block the + GRPO row is the corresponding vanilla GRPO (Shao et al., 2024) baseline. **Bold**: SR-GRPO improves over the GRPO baseline.

Model	GPQA Diamond	MMLU-Pro	Avg.
DeepSeek-R1-Distill-Qwen-1.5B (DeepSeek-AI, 2025)	34.7	38.3	36.5
+ GRPO (JustRL-DeepSeek-1.5B) (He et al., 2025)	40.5	44.7	42.6
+ **SR-GRPO (Ours)**	**46.3**	**47.1**	**46.7**
OpenMath-Nemotron-1.5B (Moshkov et al., 2025)	21.5	26.8	24.2
+ GRPO (JustRL-Nemotron-1.5B) (He et al., 2025)	27.1	28.9	28.0
+ **SR-GRPO (Ours)**	**33.6**	**31.8**	**32.7**
Qwen3-4B-Thinking-2507 (Qwen Team, 2025a)	64.9	73.3	69.1
+ GRPO (Shao et al., 2024)	65.3	73.6	69.5
+ **SR-GRPO (Ours)**	**68.8**	**74.7**	**71.8**

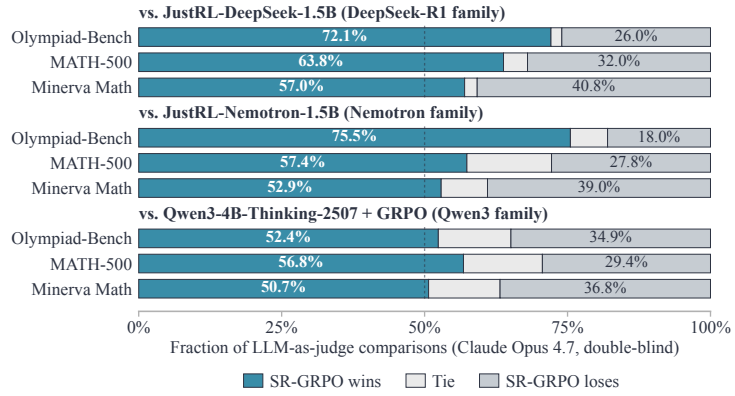

Figure 2: **Reasoning-quality evaluation via LLM-as-judge.** Pairwise, double-blind preference of SR-GRPO reasoning paths vs. the matched RL baseline at each backbone, scored by Claude Opus 4.7 (Anthropic, 2026) on three reasoning-dense benchmarks. SR-GRPO wins the majority in every (backbone, benchmark) pair.

Lifting a strong math-focused base to a new 1.5B SOTA (OpenMath-Nemotron-1.5B). On this already-strong, math-specialized backbone, SR-GRPO outperforms both the raw backbone and the matched vanilla GRPO baseline (JustRL-Nemotron-1.5B) on every benchmark, and the resulting 1.5B model outperforms the current 1.5B SOTA (QuestA-Nemotron-1.5B (Li et al., 2026)) on every one of the nine benchmarks, establishing a new SOTA at 1.5B.

Generalizing to a larger 4B base where vanilla GRPO has saturated (Qwen3-4B-Thinking-2507). At the 4B Qwen3 backbone, matched-recipe vanilla GRPO yields only marginal gains over the raw backbone (Table 1), indicating saturation of single-stage GRPO; yet SR-GRPO adds a substantial further lift, surpassing the GRPO baseline on every benchmark we tested (with one tie on the near-saturated AMC 2023). Even with its sampling budget matched to ours, vanilla GRPO stays well below SR-GRPO (App. D.2), so the gain is not bought with extra sampling.

Matching or beating much larger open-source reasoners. We compare against the 20–32B reasoner suite reported in Hochlehnert et al. (2025) on the six math benchmarks. On average across these six benchmarks, our 1.5B SR-GRPO (Nemotron family) matches the strongest of these models, QwQ-32B, and outperforms every other reported large reasoner in the suite (Light-R1-32B-DS, TinyR1-32B-Preview, OpenThinker2-32B, INTELLECT-2, gpt-oss-20b, DeepSeek-R1-Distill-Qwen-32B), with $\sim$20$\times$ fewer parameters. At 4B, SR-GRPO (Qwen3 family) outperforms *every* large reasoner in the suite on *every* one of the six shared benchmarks with far fewer parameters. Separately,

Table 3: **Qualitative comparison.** Ours vs. the SOTA RL baseline (Example 1) and a test-time self-refinement demonstration (Example 2). Excerpts of the actual model CoTs; green highlights the key correct steps, red highlights the wrong commitments. Additional cases in App. D.7.

JustRL-DeepSeek-1.5B (SOTA RL method)	**SR-GRPO (Ours)**
Example 1 — HMMT 2025 (minimal monic polynomial).	
Problem: Let $\mathcal{S}$ be the set of nonconstant monic polynomials $P \in \mathbb{Z}[x]$ with $P(\sqrt{3}+\sqrt{2})=P(\sqrt{3}-\sqrt{2})$. If $Q \in \mathcal{S}$ has minimal degree, compute $Q(10)-Q(0)$. **Ground Truth:** 890	
"*What's the minimal polynomial over $\mathbb{Q}$ for $\sqrt{3}+\sqrt{2}$? ... $\alpha^4-10\alpha^2+1=0$. So the minimal polynomial is degree 4.*" "*Which suggests that P must be a polynomial in the minimal polynomial of α... P must be a polynomial in terms of x^2. We find that the minimal degree polynomial satisfying the condition is of degree 4. The polynomial $Q(x)=x^4-10x^2+b$... satisfies the condition.*" $Q(10)-Q(0)=(10000-1000+b)-b=9000$. **Answer:** $\boxed{9000}$ ✗	"*What about degree 3? Let's assume $P(x)=x^3+cx+d$... Set $P(a)=P(b)$: $(a-b)(a^2+ab+b^2+c)=0$. Since $a\neq b$, so $a^2+ab+b^2+c=0$.*" ***"Compute*** a^2+ab+b^2. ... $a^2=5+2\sqrt{6}$, $b^2=5-2\sqrt{6}$, $ab=3-2=1$. ***Therefore*** $a^2+ab+b^2=5+1+5=11$. ***So*** $11+c=0 \Rightarrow c=-11$.**"** $Q(x)=x^3-11x+d$, and $Q(10)-Q(0)=1000-110=890$. **Answer:** $\boxed{890}$ ✓
SR-GRPO (Ours)	**SR-GRPO + Test-Time Self-Refinement (Ours)**
Example 2 — CMIMC 2025 ("good day" divisibility count).	
Problem: A "good day" is one on which both the day D and the month M divide the concatenation $\overline{MD}$ (e.g., Mar. 15: 3, 15 | 315). How many good days are in March, April, and May combined? **Ground Truth:** 13	
"*Case1b: D >=10. N=400 + D... Divisors of400:1,2,4,5,8,10,16,20,25,...*" "*In 10-31:10,16,20. So D=10,16,20.*" "*Thus, April: 4,8,10,16,20 ⇒5 days.*" "*Total: 5+5+4=14.*" **Answer:** $\boxed{14}$ ✗	***"...I realize I might have made a miscalculation earlier. Let me verify the counts again."*** "*Case2: D >=10 ⇒ N=400 + D. 4|N ⇒ D divisible by4. D|400. ... divisible by4: D=12,16,20,24. Check which of these also divide400. 12: 400/12 ≈ 33.33 ⇒ no. 16: 400/16=25 ⇒ yes. 20: 400/20=20 ⇒ yes. 24: 400/24 ≈ 16.66 ⇒ no. Thus, D=16,20.*" ***"So April has D=4,8,16,20 ⇒4 days?"*** ***"earlier recounting gave 5+5+4=14, which was incorrect."*** "*Total:5+4+4=13.*" **Answer:** $\boxed{13}$ ✓

App. D.1 also compares SR-GRPO against Critique-GRPO (Zhang et al., 2025), which uses GPT-4o critiques filtered for ground-truth consistency.

Bonus: test-time self-refinement also lifts accuracy. The numbers in Table 1 report SR-GRPO at the same inference cost as the matched RL baseline: the refinement capability is already absorbed into the first-stage policy via self-distilling feedback, and these training-time gains are our central contribution. Because the refinement prompt requires neither ground truth nor an external critic, that capability is also directly deployable at test time. Invoking it on SR-GRPO yields a positive lift on *every* benchmark, whereas the same prompt produces only a marginal change on the GRPO baseline, echoing prior findings on intrinsic self-correction without ground-truth feedback (Huang et al., 2024; Havrilla et al., 2024). Test-time scaling is not the focus of this work; see App. D.4 (Table 7).

4.3 Out-of-Distribution Generalization to Science and General Reasoning

Table 1 already shows SR-GRPO generalizes across backbone families and scales. A more demanding test is whether the trained refinement capability transfers *outside* the math training distribution. We evaluate on GPQA Diamond (Rein et al., 2023) (graduate-level physics, chemistry, and biology) and MMLU-Pro (Wang et al., 2024b) (a 14-domain reasoning benchmark spanning humanities, law, economics, engineering, and the natural sciences). Table 2 shows SR-GRPO improves over the SOTA RL baseline on *every* OOD benchmark across all three backbones, despite training data containing only math problems. The lift is most pronounced on graduate-level science (GPQA Diamond), and largest in absolute terms on the math-focused OpenMath-Nemotron-1.5B backbone, where the OOD headroom is greatest. This suggests SR-GRPO induces a general reasoning capability rather than math-specific patterns; a judge study with length and format controls supports this (App. D.6).

4.4 Quantifying Improved Reasoning Quality via LLM-as-Judge

Pass@1 alone cannot distinguish correct reasoning from flawed reasoning that arrives at the right answer by accident. We address this with a pairwise, double-blind LLM-as-judge protocol that compares SR-GRPO's reasoning path to the matched SOTA RL baseline's path with anonymized side

Table 4: **Ablation study** on the DeepSeek-R1-Distill-Qwen-1.5B backbone, removing each SR-GRPO mechanism in turn (**WF**: Wrong-First Sampling; **SR Train**: Self-Refine Stage Training, i.e. the second-stage update under the ground-truth-free refinement prompt; **SD FB**: Self-Distilling Feedback). Pass@1 accuracy (%) averaged across 4 seeds per benchmark; **Bold**/underline: best/second.

Configuration	WF	SR Train	SD FB	MATH 500	AIME 2024	AIME 2025	AMC 2023	Minerva Math	Olympiad Bench	BRUMO 2025	CMIMC 2025	HMMT 2025	Avg.
SR-GRPO w/o Wrong-First Sampling	×	✓	✓	92.1	53.3	39.2	92.5	57.8	66.9	51.7	28.8	20.8	55.9
SR-GRPO w/o Self-Refine Stage Training	✓	×	✓	91.8	50.8	40.8	94.4	55.4	69.3	47.5	23.8	21.7	55.1
SR-GRPO w/o Self-Distilling Feedback	✓	✓	×	92.3	51.7	41.7	93.1	56.0	68.4	54.2	25.6	22.5	56.2
SR-GRPO (full)	✓	✓	✓	**94.3**	**61.7**	**55.8**	**95.6**	**60.3**	**72.6**	**62.5**	**33.1**	**34.2**	**63.3**

identifiers (Figure 2). SR-GRPO wins the majority of comparisons in *every* (backbone, benchmark) pair, suggesting that the accuracy gains in Tables 1–2 reflect path-level reasoning quality rather than a final-answer shortcut. Preference margins are most pronounced on Olympiad-Bench at the two 1.5B backbones, where multi-step deductions and case analysis are central.

4.5 Qualitative Results of Improved Reasoning Capability

Two reasoning patterns recur in our qualitative analysis, demonstrating the effectiveness of SR-GRPO. (1) On hard contest problems where even the SOTA RL recipe (JustRL) fails, SR-GRPO's first stage tends to identify the *minimum* condition the answer must satisfy, while the GRPO baseline anchors on a familiar-looking but stronger one: it over-engineers to the full minimal polynomial when the problem only requires invariance under a single conjugation. SR-GRPO demonstrates better reasoning and correctly solves the problem (Table 3). (2) When SR-GRPO's first stage is itself flawed, our model in the refinement stage tends to flag the prior mistake (*"I realize I might have made a miscalculation earlier. Let me verify the counts again"* on the CMIMC 2025 example in Table 3), locate the specific step that went wrong, and re-apply the correct constraint uniformly rather than restarting from the problem statement; for more qualitative results, see App. D.7.

4.6 Ablation Study: Isolating the Contribution of Each Mechanism

We isolate each mechanism by removing it one at a time in Table 4. Each removal decreases the average performance of SR-GRPO: no single mechanism suffices, and the three are not interchangeable. **Disabling second-stage training** is the most damaging removal, consistent with prior findings that intrinsic self-correction must be *trained* rather than elicited at inference (Huang et al., 2024; Havrilla et al., 2024). **Removing wrong-first sampling** disproportionately hurts the hardest competition benchmarks (e.g., HMMT 2025): uniform sampling starves the refinement-stage gradient of hard mistakes. **Removing self-distilling feedback** prevents the refinement stage's verified corrections from improving the solver itself, so the gain over the vanilla GRPO baseline is substantially reduced even though the refinement stage is still trained.

5 Conclusion

Standard RLVR never trains a self-refine capability: a single flawed step propagates unchecked, and most existing remedies either degrade accuracy when prompted at inference or import supervision stronger than RLVR's binary verifier. We introduce **SR-GRPO**, a two-stage RLVR framework whose three mechanisms (*wrong-first sampling*, a *ground-truth-free* refinement prompt, and *self-distilling feedback*) let one shared policy serve as solver, critic, and teacher in a single training loop, without auxiliary models or extra inference compute. Under one fixed recipe across three backbone families and two scales, SR-GRPO matches or exceeds the strongest public GRPO baselines on several math reasoning benchmarks; at 1.5B it matches QwQ-32B, and at 4B it surpasses several open-source 20B+ reasoners. Despite math-only training, both OOD benchmarks improve across all three backbone families (Table 2). Double-blind LLM-as-judge and qualitative inspection confirm path-level reasoning gains; ablations find all three mechanisms necessary. SR-GRPO recasts self-refinement as training-time self-distillation, generalizable across architectures and reasoning domains. See App. A for further discussion of our findings, insights, and limitations.

AI USE STATEMENT

We fine-tune three publicly available math reasoning LLMs—DeepSeek-R1-Distill-Qwen-1.5B (DeepSeek-AI, 2025), OpenMath-Nemotron-1.5B (Moshkov et al., 2025), and Qwen3-4B-Thinking-2507 (Qwen Team, 2025a)—under SR-GRPO. We use Claude Opus 4.7 (Anthropic, 2026) as the LLM-as-judge for evaluating reasoning quality (Sec. 4.4, App. C.11). Beyond these uses, LLMs are used only to assist with writing (polishing and editing the manuscript for readability). They do not contribute to the core methodology, scientific rigor, or originality of the research. We have reviewed all AI-assisted text and take full responsibility for the content of this paper.

ETHICS STATEMENT

We have read and adhere to the ICLR Code of Ethics. This work involves no crowdsourcing or research with human subjects, and it does not introduce new datasets; we use publicly available backbones, training data (DAPO-Math-17k (Yu et al., 2025)), and benchmarks (App. C.1, App. C.3). By training self-refinement with only the binary verifier RLVR already requires, SR-GRPO removes the dependency on proprietary critic models or human-curated correction traces, lowering the barrier to strong reasoners; its limitations are discussed in App. A.2.

REPRODUCIBILITY STATEMENT

The method is specified in Sec. 3, and Algorithm 1 (App. B) fully specifies the SR-GRPO procedure. Experimental details are given in App. C: base models (App. C.1), baselines (App. C.2), benchmarks (App. C.3), the self-refinement instruction (App. C.4), reward function details (App. C.5), implementation notes for wrong-first sampling and self-distilling feedback (App. C.6 and App. C.7), hyperparameters (App. C.8), computation resources (App. C.9), training data (App. C.10), the LLM-as-judge protocol (App. C.11), and further reproducibility details (App. C.12); standard errors are reported in App. D.5. We use publicly available backbones, training data, and the veRL (Sheng et al., 2025) training framework, and we plan to release our code on top of the veRL framework after publication.

REFERENCES

Anthropic. Introducing Claude Opus 4.7. https://www.anthropic.com/news/claude-opus-4-7, 2026.

Mislav Balunovic, Jasper Dekoninck, Ivo Petrov, Nikola Jovanovic, and Martin T. Vechev. Matharena: Evaluating llms on uncontaminated math competitions. *CoRR*, abs/2505.23281, 2025. doi: 10.48550/ARXIV.2505.23281. URL https://doi.org/10.48550/arXiv.2505.23281.

Zhipeng Chen, Yingqian Min, Beichen Zhang, Jie Chen, Jinhao Jiang, Daixuan Cheng, Wayne Xin Zhao, Zheng Liu, Xu Miao, Yang Lu, Lei Fang, Zhongyuan Wang, and Ji-Rong Wen. An empirical study on eliciting and improving r1-like reasoning models. *CoRR*, abs/2503.04548, 2025. doi: 10.48550/ARXIV.2503.04548. URL https://doi.org/10.48550/arXiv.2503.04548.

Eugene Choi, Arash Ahmadian, Matthieu Geist, Olivier Pietquin, and Mohammad Gheshlaghi Azar. Self-improving robust preference optimization. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=ZSdubdbOoi.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021. URL https://arxiv.org/abs/2110.14168.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948, 2025. doi: 10.48550/ARXIV.2501.12948. URL https://doi.org/10.48550/arXiv.2501.12948.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, Kashun Shum, and Tong Zhang. RAFT: reward ranked finetuning for generative foundation model alignment. *Trans. Mach. Learn. Res.*, 2023, 2023. URL https://openreview.net/forum?id=m7p5O7zblY.

Gemini Team. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *CoRR*, abs/2507.06261, 2025. doi: 10.48550/ARXIV.2507.06261. URL https://doi.org/10.48550/arXiv.2507.06261.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. CRITIC: large language models can self-correct with tool-interactive critiquing. 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=Sx038qxjek.

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.

Alexander Havrilla, Sharath Chandra Raparthy, Christoforos Nalmpantis, Jane Dwivedi-Yu, Maksym Zhuravinskyi, Eric Hambro, and Roberta Raileanu. Glore: When, where, and how to improve LLM reasoning via global and local refinements. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, Proceedings of Machine Learning Research, pp. 17719–17733. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/havrilla24a.html.

Bingxiang He, Zekai Qu, Zeyuan Liu, Yinghao Chen, Yuxin Zuo, Cheng Qian, Kaiyan Zhang, Weize Chen, Chaojun Xiao, Ganqu Cui, Ning Ding, and Zhiyuan Liu. Justrl: Scaling a 1.5b LLM with a simple RL recipe. *CoRR*, abs/2512.16649, 2025. doi: 10.48550/ARXIV.2512.16649. URL https://doi.org/10.48550/arXiv.2512.16649.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. Olympiadbench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 3828–3850. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.211. URL https://doi.org/10.18653/v1/2024.acl-long.211.

Andreas Hochlehnert, Hardik Bhatnagar, Vishaal Udandarao, Samuel Albanie, Ameya Prabhu, and Matthias Bethge. A sober look at progress in language model reasoning: Pitfalls and paths to reproducibility. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=90UrTTxp5O.

Arian Hosseini, Xingdi Yuan, Nikolay Malkin, Aaron C. Courville, Alessandro Sordoni, and Rishabh Agarwal. V-star: Training verifiers for self-taught reasoners. *CoRR*, abs/2402.06457, 2024. doi: 10.48550/ARXIV.2402.06457. URL https://doi.org/10.48550/arXiv.2402.06457.

Jian Hu, Mingjie Liu, Ximing Lu, Fang Wu, Zaïd Harchaoui, Shizhe Diao, Yejin Choi, Pavlo Molchanov, Jun Yang, Jan Kautz, and Yi Dong. Brorl: Scaling reinforcement learning via broadened exploration. *CoRR*, abs/2510.01180, 2025. doi: 10.48550/ARXIV.2510.01180. URL https://doi.org/10.48550/arXiv.2510.01180.

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.

Jonas Hübotter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, and Andreas Krause. Reinforcement learning via self-distillation. *CoRR*, abs/2601.20802, 2026. doi: 10.48550/ARXIV.2601.20802. URL https://doi.org/10.48550/arXiv.2601.20802.

Kimi Team. Kimi k1.5: Scaling reinforcement learning with llms. *CoRR*, abs/2501.12599, 2025. doi: 10.48550/ARXIV.2501.12599. URL https://doi.org/10.48550/arXiv.2501.12599.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D. Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei M. Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal M. P. Behbahani, and Aleksandra Faust. Training language models to self-correct via reinforcement learning. In *The Thirteenth International Conference on Learning Representations, ICLR 2025, Singapore, April 24-28, 2025*. OpenReview.net, 2025. URL https://openreview.net/forum?id=CjwERcAU7w.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay V. Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. 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/18abbeef8cfe9203fdf9053c9c4fe191-Abstract-Conference.html.

Jiazheng Li, Hongzhou Lin, Hong Lu, Kaiyue Wen, Zaiwen Yang, Jiaxuan Gao, Yi Wu, and Jingzhao Zhang. Questa: Expanding reasoning capacity in LLMs via question augmentation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=3MifB0f7qR.

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.

Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. Prorl: Prolonged reinforcement learning expands reasoning boundaries in large language models. *CoRR*, abs/2505.24864, 2025. doi: 10.48550/ARXIV.2505.24864. URL https://doi.org/10.48550/arXiv.2505.24864.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback. 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/91edff07232fb1b55a505a9e9f6c0ff3-Abstract-Conference.html.

Mathematical Association of America. American mathematics competitions, 2023.

Ivan Moshkov, Darragh Hanley, Ivan Sorokin, Shubham Toshniwal, Christof Henkel, Benedikt Schifferer, Wei Du, and Igor Gitman. AIMO-2 winning solution: Building state-of-the-art mathematical reasoning models with openmathreasoning dataset. *CoRR*, abs/2504.16891, 2025. doi: 10.48550/ARXIV.2504.16891. URL https://doi.org/10.48550/arXiv.2504.16891.

OpenAI. Openai o1 system card. *CoRR*, abs/2412.16720, 2024. doi: 10.48550/ARXIV.2412.16720. URL https://doi.org/10.48550/arXiv.2412.16720.

OpenAI. gpt-oss-120b & gpt-oss-20b model card. *CoRR*, abs/2508.10925, 2025. doi: 10.48550/ARXIV.2508.10925. URL https://doi.org/10.48550/arXiv.2508.10925.

Prime Intellect Team, Sami Jaghouar, Justus Mattern, Jack Min Ong, Jannik Straube, Manveer Basra, Aaron Pazdera, Kushal Thaman, Matthew Di Ferrante, Felix Gabriel, Fares Obeid, Kemal Erdem, Michael Keiblinger, and Johannes Hagemann. INTELLECT-2: A reasoning model trained through globally decentralized reinforcement learning. *CoRR*, abs/2505.07291, 2025. doi: 10.48550/ARXIV.2505.07291. URL https://doi.org/10.48550/arXiv.2505.07291.

Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. Agent Q: advanced reasoning and learning for autonomous AI agents. *CoRR*, abs/2408.07199, 2024. doi: 10.48550/ARXIV.2408.07199. URL https://doi.org/10.48550/arXiv.2408.07199.

Yuxiao Qu, Tianjun Zhang, Naman Garg, and Aviral Kumar. Recursive introspection: Teaching language model agents how to self-improve. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/639d992f819c2b40387d4d5170b8ffd7-Abstract-Conference.html.

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

Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, March 2025b. URL https://qwenlm.github.io/blog/qwq-32b/.

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.

William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. Self-critiquing models for assisting human evaluators. *CoRR*, abs/2206.05802, 2022. doi: 10.48550/ARXIV.2206.05802. URL https://doi.org/10.48550/arXiv.2206.05802.

Lior Shani, Aviv Rosenberg, Asaf B. Cassel, Oran Lang, Daniele Calandriello, Avital Zipori, Hila Noga, Orgad Keller, Bilal Piot, Idan Szpektor, Avinatan Hassidim, Yossi Matias, and Rémi Munos. Multi-turn reinforcement learning from preference human feedback. *CoRR*, abs/2405.14655, 2024. doi: 10.48550/ARXIV.2405.14655. URL https://doi.org/10.48550/arXiv.2405.14655.

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.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient RLHF framework. In *Proceedings of the Twentieth European Conference on Computer Systems, EuroSys 2025, Rotterdam, The Netherlands, 30 March 2025 - 3 April 2025*, pp. 1279–1297. ACM, 2025. doi: 10.1145/3689031.3696075. URL https://doi.org/10.1145/3689031.3696075.

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.

Avi Singh, John D. Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Xavier Garcia, Peter J. Liu, James Harrison, Jaehoon Lee, Kelvin Xu, Aaron T. Parisi, Abhishek Kumar, Alexander A. Alemi, Alex Rizkowsky, Azade Nova, Ben Adlam, Bernd Bohnet, Gamaleldin Fathy Elsayed, Hanie Sedghi, Igor Mordatch, Isabelle Simpson, Izzeddin Gur, Jasper Snoek, Jeffrey Pennington, Jiri Hron, Kathleen Kenealy, Kevin Swersky, Kshiteej Mahajan, Laura Culp, Lechao Xiao, Maxwell L. Bileschi, Noah Constant, Roman Novak, Rosanne Liu, Tris Warkentin, Yundi Qian, Yamini Bansal, Ethan Dyer, Behnam Neyshabur, Jascha Sohl-Dickstein, and Noah Fiedel. Beyond human data: Scaling self-training for problem-solving with language models. *Trans. Mach. Learn. Res.*, 2024, 2024. URL https://openreview.net/forum?id=lNAyUngGFK.

Lin Sun, Guangxiang Zhao, Xiaoqi Jian, Yuhan Wu, Weihong Lin, Yongfu Zhu, Qilong Shi, Change Jia, Aomufei Yuan, Yuxuan Tian, Linglin Zhang, Jinzhu Wu, Junfeng Ran, Sai er Hu, Zihan Jiang, Junting Zhou, Wenrui Liu, Xusen Xiao, Bin Cui, Tong Yang, and Xiangzheng Zhang. Tinyr1-32b-preview: Boosting accuracy with branch-merge distillation. *CoRR*, abs/2503.04872, 2025. doi: 10.48550/ARXIV.2503.04872. URL https://doi.org/10.48550/arXiv.2503.04872.

Sijun Tan, Michael Luo, Justin Wong, Colin Cai, Xiaoxiang Shi, William Yuan Tang, Manan Roongta, Tianjun Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Effective RL scaling of reasoning models via iterative context lengthening, 2026. URL https://openreview.net/forum?id=I6GzDCne7U.

Zhongwei Wan, Zhihao Dou, Che Liu, Yu Zhang, Dongfei Cui, Qinjian Zhao, Hui Shen, Jing Xiong, Yi Xin, Yifan Jiang, Chaofan Tao, Yangfan He, Mi Zhang, and Shen Yan. SRPO: enhancing multimodal LLM reasoning via reflection-aware reinforcement learning. *CoRR*, abs/2506.01713, 2025. doi: 10.48550/ARXIV.2506.01713. URL https://doi.org/10.48550/arXiv.2506.01713.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 9426–9439. Association for Computational Linguistics, 2024a. doi: 10.18653/V1/2024.ACL-LONG.510. URL https://doi.org/10.18653/v1/2024.acl-long.510.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024b. URL http://papers.nips.cc/paper_files/paper/2024/hash/ad236edc564f3e3156e1b2feafb99a24-Abstract-Datasets_and_Benchmarks_Track.html.

Sean Welleck, Ximing Lu, Peter West, Faeze Brahman, Tianxiao Shen, Daniel Khashabi, and Yejin Choi. Generating sequences by learning to self-correct. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=hH36JeQZDaO.

Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Lifu Tang, Xiaowei Lv, Haosheng Zou, Yongchao Deng, Shousheng Jia, and Xiangzheng Zhang. Light-r1:

Curriculum sft, DPO and RL for long COT from scratch and beyond. In Georg Rehm and Yunyao Li (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pp. 318–327. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.ACL-INDUSTRY.24. URL https://doi.org/10.18653/v1/2025.acl-industry.24.

Sen Xu, Yi Zhou, Wei Wang, Jixin Min, Zhibin Yin, Yingwei Dai, Shixi Liu, Lianyu Pang, Yirong Chen, and Junlin Zhang. Tiny model, big logic: Diversity-driven optimization elicits large-model reasoning ability in vibethinker-1.5b. *CoRR*, abs/2511.06221, 2025. doi: 10.48550/ARXIV.2511.06221. URL https://doi.org/10.48550/arXiv.2511.06221.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *CoRR*, abs/2409.12122, 2024. doi: 10.48550/ARXIV.2409.12122. URL https://doi.org/10.48550/arXiv.2409.12122.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Weinan Dai, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. DAPO: an open-source LLM reinforcement learning system at scale. *CoRR*, abs/2503.14476, 2025. doi: 10.48550/ARXIV.2503.14476. URL https://doi.org/10.48550/arXiv.2503.14476.

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.

Xiaoying Zhang, Yipeng Zhang, Hao Sun, Kaituo Feng, Chaochao Lu, Chao Yang, and Helen Meng. Critique-grpo: Advancing LLM reasoning with natural language and numerical feedback. *CoRR*, abs/2506.03106, 2025. doi: 10.48550/ARXIV.2506.03106. URL https://doi.org/10.48550/arXiv.2506.03106.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. Archer: Training language model agents via hierarchical multi-turn RL. In Ruslan Salakhutdinov, Zico Kolter, Katherine A. Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, Proceedings of Machine Learning Research, pp. 62178–62209. PMLR / OpenReview.net, 2024. URL https://proceedings.mlr.press/v235/zhou24t.html.

A DISCUSSION

In this paper, we set out to address a structural limitation of RLVR for reasoning that prior work could only sidestep, not solve: the policy is trained to produce one chain of thought per rollout and is never asked to inspect, diagnose, or repair its own mistakes. Existing remedies fall into two camps that cannot address this gap. Inference-time prompting protocols (Madaan et al., 2023; Shinn et al., 2023) elicit self-refinement without updating weights and have been repeatedly shown to *degrade* accuracy when no ground-truth feedback is available (Huang et al., 2024; Havrilla et al., 2024); tool-augmented or critic-augmented variants (Gou et al., 2024) only sidestep this by importing an external feedback channel. Training-time approaches *do* update weights, but most import a supervision signal stronger than RLVR's binary verifier (a teacher's correction trace, an oracle, step-level annotations, or a separately trained PRM); SCoRe, which learns from outcome rewards alone, is different: it trains self-correction only, its refinements never reach the solver, and its first attempt can even fall below the base model (App. A.1). We ask whether self-refinement can be learned with only the binary verifier RLVR already uses, and whether its verified refinements can in turn train a stronger solver. SR-GRPO addresses both questions by reframing self-refinement not as an inference protocol nor as an extra supervised dataset but as a *training-time data-augmentation device* inside the GRPO update.

Key Insight. Self-refinement need not be supervised, nor prompted at deployment. It can be *trained directly into the policy weights using outcome correctness alone*, by extending each rollout into a solve-and-self-refine trajectory and recycling every verified correction back into the solve stage. A single shared policy plays both solver and refiner, with no external oracle, no separate critic, and no ground-truth answer at refinement.

We summarize the empirical evidence as five findings, each tied to a specific result in the paper.

Key Finding 1. Under one fixed recipe, SR-GRPO surpasses the strongest public GRPO baselines on every math benchmark across three backbone families and two scales, with the only tie on the near-saturated AMC 2023 (Table 1).

The same fixed SR-GRPO recipe applied to three backbones (one weaker general-reasoning, one stronger math-focused, one larger 4B reasoner) consistently outperforms the corresponding vanilla GRPO baseline. At 1.5B SR-GRPO outperforms the current 1.5B SOTA (QuestA-Nemotron-1.5B) and is on par with the open-source 32B SOTA (QwQ-32B) at roughly an order of magnitude fewer parameters; at 4B, where vanilla GRPO has visibly saturated, SR-GRPO continues to deliver substantial further gains. That the same fixed recipe lifts every backbone we tested indicates the gain is attributable to the training objective rather than backbone-specific tuning. The cross-backbone consistency across three distinct families (DeepSeek, Nemotron, Qwen3) at two scales also rules out the hypothesis that SR-GRPO merely overfits a particular backbone's pretraining template: a template-matching effect would not lift all three families uniformly under the same fixed recipe.

Key Finding 2. Despite training only on math, SR-GRPO improves over the GRPO baseline on *every* OOD benchmark (GPQA Diamond, MMLU-Pro) and on *every* backbone (Table 2), arguing against per-benchmark reward hacking.

If the math gain were merely template matching, the lift would not extend to graduate-level science questions or to multi-domain multiple-choice reasoning. That it does, on every backbone, suggests that the refinement objective rewards *transferable* reasoning (the act of locating and repairing flawed steps) rather than memorization of math-specific patterns. The OOD judge study in App. D.6 points the same way and finds similar format accuracy and mean response length on MMLU-Pro (DeepSeek backbone).

Key Finding 3. Reasoning-path quality, scored by a double-blind LLM-as-judge, improves in *every* backbone $\times$ benchmark pair against the matched SOTA RL baseline; qualitative inspection shows that the refinement stage tends to locate and revise specific erroneous steps rather than restart from scratch (Figure 2, Table 3).

Aggregate accuracy is necessary but not sufficient: a model could arrive at the correct final answer through a flawed chain. Pairwise LLM-as-judge directly scores the quality of the chain itself, and SR-GRPO wins the majority of comparisons in every (backbone, benchmark) pair. The qualitative examples make this concrete: in the refinement-save cases, refinement does not regenerate a fresh solution but re-examines the previous one, *explicitly flags* a prior mistake, locates the specific step that went wrong, and corrects only that step. The verbatim self-refinement openings on our two

highlighted refinement-save cases make this operational signature of *intrinsic diagnosis* (as opposed to independent restart) directly visible: *"...I realize I might have made a miscalculation earlier. Let me verify the counts again"* (CMIMC 2025, Table 3) and *"looking back at the previous solution, the user converted* $408\,nm$ *to angstroms by dividing by* 10*, resulting in* $40.8\,Å$*. That was incorrect"* (Minerva, App. D.7.2). In both cases, the model names the prior step before rewriting it, exactly the behavior the delta refinement reward r_2 incentivizes.

Key Finding 4. The refinement capability must be *trained* into the weights, not merely elicited at inference. As a natural by-product of the ground-truth-free training design, the same prompt-only refinement pass used during training also serves as a test-time procedure: applied to SR-GRPO it yields a positive lift on every benchmark, while applied to the matched GRPO baseline it produces only mixed per-benchmark effects (App. D, Table 7).

The matched-prompt comparison isolates *prompt* from *capability*: the prompt is identical in both rows, so the asymmetry in average lift can only come from where the self-refine capability lives. In SR-GRPO it lives in the weights, where it strengthens systematically under reflection; in the GRPO baseline there is no such trained capability to elicit, so the same prompt produces only a small marginal lift (consistent with prior intrinsic-self-correction findings (Huang et al., 2024; Havrilla et al., 2024)). Test-time scaling is *not* the focus of this paper; we simply note that, because SR-GRPO neither depends on ground truth nor invokes an external critic, the trained refinement capability remains usable at deployment for additional accuracy at extra inference cost.

Key Finding 5. Each of the three SR-GRPO mechanisms is necessary; removing any one eliminates most of SR-GRPO's gain over the GRPO baseline despite the other two still being active (Table 4).

The three mechanisms play complementary roles and are not interchangeable: second-stage training alone produces an intrinsic refinement capability (consistent with prior negative findings on prompting alone); wrong-first sampling keeps that capability's gradient dense as the solver improves and most rollouts become correct; and self-distilling feedback feeds the refinement stage's verified successes back into the solver, so the solver itself improves rather than only the refinement stage.

A.1 Relation to RL-Based Self-Correction and Self-Distillation

SCoRe (Kumar et al., 2025) has a different goal from SR-GRPO: it learns a self-correction skill, whereas SR-GRPO uses self-refinement to train a better solver. Accordingly, SCoRe's refinements never become solver training data, and its first attempt can even fall below the base model (HumanEval, Table 3 of Kumar et al. (2025)); it also refines every first attempt rather than prioritizing failures, and it trains in two separate stages. SR-GRPO prioritizes incorrect solve traces (Sec. 3.3), feeds every verified refinement back into the solve stage (Sec. 3.4), and trains solver and refiner jointly in a single run.

SDPO (Hübotter et al., 2026) re-scores the policy's own rollouts under a feedback-augmented context to obtain denser per-token credit; it generates no refined solution, so refinement quality is never rewarded. SR-GRPO generates refined trajectories explicitly, rewards them by the improvement they yield (Eq. (6)), and returns every verified refinement to solver training.

Unlike STaR's rationalization (Zelikman et al., 2022), which conditions on the answer as a hint, or Critique-GRPO (Zhang et al., 2025), whose refinements are conditioned on external critiques or the ground-truth answer, SR-GRPO trains on the policy's own verified refinements, obtained without an external model or hints.

A.2 LIMITATIONS

SR-GRPO incurs additional training-time compute relative to single-stage GRPO: each rollout adds a refinement group of size G_2 on top of the solve group of size G_1, roughly doubling rollout cost during training (we use $G_1 = G_2 = 8$). Our experiments validate SR-GRPO at the 1.5B and 4B parameter scales and on three backbone families (DeepSeek-R1-Distill-Qwen-1.5B, OpenMath-Nemotron-1.5B, Qwen3-4B-Thinking-2507); direct validation at the 32B+ scale is important future work as compute permits. We focus on mathematical reasoning with binary answer-level rewards. Extension to code synthesis, where execution already provides a binary signal, is a natural next step but is not validated here. Extension to open-ended generation (proof generation, scientific writing) would require a stronger structural verifier or a learned quality metric to score the refinement stage. Finally, the trained solver is not infallible. On the hardest problems (e.g., AIME/HMMT 2025), the first-stage chain still errs on a non-trivial fraction of queries. Invoking the same self-refinement prompt at test time (App. D.4) recovers many of these cases and yields a further accuracy lift, demonstrating that the trained capability remains available at inference for any deployer willing to spend additional compute.

B ALGORITHM PSEUDOCODE

Algorithm 1 gives the end-to-end SR-GRPO training loop. Each iteration produces one solve group and one refinement group per query, applies the wrong-first selection rule, forms the augmented solve group $\mathcal{G}_1^{\mathrm{Aug}}$ as in Eq. (9) by appending the verified refinements $\mathcal{V}$ (Def. 1), and takes a single gradient step on the sum of the two stages' GRPO losses in Eq. (7). For implementation-level details (prompt template, reward design, hyperparameters, compute), see App. C.

Algorithm 1 SR-GRPO: GRPO with Self-Refinement

Require: Dataset $\mathcal{D}$, base policy π_{ref}, solve group size G_1, refinement group size G_2, KL weight β, format weight λ, clip threshold ϵ

Initialize $\pi_\theta \leftarrow \pi_{\mathrm{ref}}$
for each training iteration **do**
  Sample batch $\{q_j\}$ from $\mathcal{D}$
  for each query q **do**
    # Solve stage
    Sample solve group $\{t_1^{(i)}\}_{i=1}^{G_1} \sim \pi_\theta(\cdot|q)$
    Let $\mathcal{W} = \{t_1^{(i)} : \mathrm{correct}(t_1^{(i)}, q) = 0\}$; $t_1^* \sim \mathrm{Uniform}(\mathcal{W})$ if $\mathcal{W} \neq \varnothing$, else $t_1^* \sim \mathrm{Uniform}(\{t_1^{(i)}\}_{i=1}^{G_1})$ *Eq.* (8)
    # Refinement stage, conditioned on t_1^*
    Append the self-refinement instruction (App. C.4) after t_1^*
    Sample G_2 refinement traces $\{t_2^{(j)}\}_{j=1}^{G_2} \sim \pi_\theta(\cdot|q, t_1^*)$
    # Self-distilling feedback (Def. 1)
    $\mathcal{V} \leftarrow \{\, t_2^{(j)} : \mathrm{correct}(t_1^*, q) = 0 \text{ and } \mathrm{correct}(t_2^{(j)}, q) = 1 \,\}$
    Form augmented solve group $\mathcal{G}_1^{\mathrm{Aug}} \leftarrow \{t_1^{(i)}\}_{i=1}^{G_1} \cup \mathcal{V}$ *Eq.* (9)
    # Per-stage rewards, Eqs. (5)–(6)
    $r_1(t) \leftarrow \mathbb{1}[\mathrm{correct}(t, q)] + \lambda\, \mathbb{1}[\mathrm{format}(t)]$ for each $t \in \mathcal{G}_1^{\mathrm{Aug}}$
    $r_2(t_2^{(j)}) \leftarrow \mathbb{1}[\mathrm{correct}(t_2^{(j)}, q)] - \mathbb{1}[\mathrm{correct}(t_1^*, q)] + \lambda\, \mathbb{1}[\mathrm{format}(t_2^{(j)})]$ for $j = 1, \ldots, G_2$
    Compute group-relative advantages independently in each stage's group via Eq. (2): solve stage uses $\mathcal{G}_1^{\mathrm{Aug}}$, refine stage uses $\{t_2^{(j)}\}_{j=1}^{G_2}$
  end for
  Update θ on $\mathcal{L}_{\mathrm{GRPO}}(\theta; r_1) + \mathcal{L}_{\mathrm{GRPO}}(\theta; r_2)$ from Eq. (7), averaged over the queries in the batch
end for
return π_θ

C Experimental Details

C.1 Base Models

We apply the same fixed SR-GRPO recipe to three publicly available backbones spanning three distinct families and two parameter scales:

- **DeepSeek-R1-Distill-Qwen-1.5B** (DeepSeek-AI, 2025): a 1.5B reasoning model distilled from DeepSeek-R1's reasoning traces (which span math and broader reasoning). We use this as a relatively *weaker* starting point on competition math.
- **OpenMath-Nemotron-1.5B** (Moshkov et al., 2025): NVIDIA's 1.5B math-specialized model, fine-tuned on the OpenMathReasoning corpus of long-reasoning competition-math solutions. We use this as a *strong, math-specialized* 1.5B backbone.
- **Qwen3-4B-Thinking-2507** (Qwen Team, 2025a): a 4B reasoner from the Qwen3 family released with native chain-of-thought capability. We use this as a *larger, already near-saturated* backbone to test whether SR-GRPO continues to deliver gains where vanilla GRPO has plateaued.

All three backbones are loaded from their official Hugging Face checkpoints with no SFT warm-up.

C.2 BASELINES

We benchmark SR-GRPO against four groups of baselines.

Primary GRPO baselines. JustRL (He et al., 2025) is a publicly released, well-tuned vanilla GRPO recipe trained on the same DAPO-Math-17k corpus we use, and is the strongest publicly reported single-stage GRPO recipe at the 1.5B scale on competition math. We deliberately adopt JustRL as our 1.5B baseline because it represents the saturated state of the single-stage GRPO recipe at this scale, so the SR-GRPO gain over JustRL is unlikely to be closed by further vanilla GRPO tuning or compute alone (App. D.2). We use its released checkpoints as our GRPO baseline on the two 1.5B backbones we share with it (JustRL-DeepSeek-1.5B, JustRL-Nemotron-1.5B), so the GRPO baseline at 1.5B has matched backbone and matched training data. At Qwen3-4B, where no public JustRL checkpoint exists, we run vanilla GRPO (Shao et al., 2024) ourselves under the same hyperparameter set as SR-GRPO, with the refinement stage and self-distilling feedback disabled (the GRPO baseline at 4B therefore has matched backbone, data, and recipe).

Current 1.5B SOTA reference methods. For scaling context at 1.5B we report DeepScaleR-1.5B (Tan et al., 2026), STILL-3-1.5B (Chen et al., 2025), BroRL-1.5B (Hu et al., 2025), and QuestA-Nemotron-1.5B (Li et al., 2026) (among the strongest publicly reported 1.5B competition-math models prior to our work).

Open-source large-scale reasoners (20–32B). For scaling context at much larger parameter counts we report DeepSeek-R1-Distill-Qwen-32B (DeepSeek-AI, 2025), INTELLECT-2 (Prime Intellect Team et al., 2025), OpenThinker2-32B (Guha et al., 2025), TinyR1-32B-Preview (Sun et al., 2025), Light-R1-32B-DS (Wen et al., 2025), gpt-oss-20b (medium) (OpenAI, 2025), and QwQ-32B (Qwen Team, 2025b), with numbers taken from the reproducibility-controlled evaluation of Hochlehnert et al. (2025). These are not directly comparable to SR-GRPO at matched backbone or matched training recipe; we include them for upper-scale reference.

Extended baseline: Critique-GRPO. Critique-GRPO (Zhang et al., 2025) is the closest published methodological peer to SR-GRPO. It also extends GRPO with a critique-and-refine signal, but operates with stronger supervision: its chain-of-thought critiques are produced by an external GPT-4o critic and filtered for consistency with the rule-based ground-truth verifier. We report its best 7–8B variants for context only; SR-GRPO uses only the binary verifier.

C.3 BENCHMARKS

Math (training-distribution). We evaluate on nine competition-style math benchmarks: **MATH-500** (Lightman et al., 2024), **AIME 2024** and **AIME 2025**, **AMC 2023** (Mathematical Association of America, 2023), **Minerva Math** (Lewkowycz et al., 2022), **Olympiad-Bench** (He et al., 2024), and the three 2025 competitions **BRUMO 2025**, **CMIMC 2025**, and **HMMT 2025**, all sourced from the MathArena collection (Balunovic et al., 2025) of recent and uncontaminated competition problems.

Out-of-distribution science and general reasoning. We additionally evaluate on two OOD reasoning benchmarks that test whether the trained refinement capability transfers outside the math training distribution: **GPQA Diamond** (Rein et al., 2023) (graduate-level physics, chemistry, and biology) and **MMLU-Pro** (Wang et al., 2024b) (a 14-domain multiple-choice reasoning benchmark spanning humanities, law, economics, engineering, and the natural sciences). Neither is included in the training data.

Evaluation protocol. All numbers are pass@1 accuracy averaged across 4 different seeds per (model, benchmark) pair. For the math benchmarks, we follow the same evaluation pipeline used by JustRL (He et al., 2025) (the same nine benchmarks, with the same rule-based answer extractor and verifier), so that any difference between SR-GRPO and the matched JustRL baseline is attributable to the self-refinement objective rather than to evaluation discrepancies; this makes the comparison against the SOTA RL baseline fair. For the OOD benchmarks (GPQA Diamond, MMLU-Pro), we evaluate using each benchmark's official codebase to keep the protocol consistent with how those scores are typically reported.

C.4 SELF-REFINEMENT INSTRUCTION

The self-refinement instruction is appended after the solve trace t_1^* at every rollout, exactly as written below, and is held fixed across all experiments, backbones, and ablations:

> Please review your previous solution carefully. Identify any errors in your reasoning or calculation. If you find a mistake, explain it clearly and provide a fully corrected solution with the correct final answer. If your solution is correct, briefly confirm the reasoning and restate your final answer.

The instruction is delivered through the backbone's standard user-turn / assistant-turn chat template, so refinement generation stays in-distribution with how the backbone was instruction-tuned. The prompt presents only the problem q and the model's own initial solution t_1^*; it never reveals the ground-truth answer, realizing the ground-truth-free constraint of Sec. 3.

C.5 Reward Function Details

The reward design follows standard RLVR practice (consistent with JustRL (He et al., 2025)) and adds only a delta term for the refinement stage.

Outcome reward. The binary verifier $\mathrm{correct}(t, q)$ checks whether the boxed final answer extracted from t matches the ground-truth answer of q via a deterministic rule-based parser covering LaTeX fractions, integers, decimals, and standard symbolic forms.

Format reward. $\mathbb{1}[\mathrm{format}(t)]$ checks that t contains a valid `<think>`$\cdots$`</think>` block followed by a boxed final answer in AMC/AIME/MATH format (e.g., `\boxed{42}`). We set $\lambda = 0.1$.

Refinement delta reward. The refinement-stage reward is the outcome delta defined in Eq. (6): $+1$ when refinement fixes a wrong solve trace ($0 \to 1$), 0 when both are correct or both wrong, and -1 when refinement breaks an already-correct solve ($1 \to 0$). We add the same format reward to the refinement stage on top of this delta.

C.6 WRONG-FIRST SAMPLING: IMPLEMENTATION NOTES

From the standard solve group of $G_1{=}8$ traces sampled per problem, the refinement input t_1^* is drawn uniformly at random from the subset of incorrect traces; if no incorrect trace exists, it is drawn uniformly from all G_1 traces (a fallback that occurs only when the solver is at ceiling on that query and is rare in practice; see Eq. (8) and Algorithm 1).

No extra solve-stage forward passes. Wrong-first sampling reuses the same G_1 solve traces that GRPO already samples for relative-advantage normalization, so it incurs no extra solve-stage forward passes beyond standard GRPO. The wall-clock overhead of SR-GRPO over single-stage GRPO comes entirely from the additional refinement-stage rollout group of $G_2{=}8$ samples used to compute refinement-stage relative advantages; for compute details, see App. C.9.

C.7 Self-Distilling Feedback: Implementation Notes

Here "self-distilling" means recycling the policy's own verified samples as training data, not matching a teacher's logits (cf. SDPO, App. A.1).

For each query, the G_2 refinement traces conditioned on the same wrong-first solve trace t_1^* form a refinement normalization group analogous to the solve group; refinement advantages are computed within this group via Eq. (2), giving the refinement stage its own RL signal. Every refinement trace that satisfies the feedback condition (i.e., produces a verified correct answer) is additionally appended to the solve-stage training group for the current step, where it participates in the solve-stage GRPO update exactly like one of the model's own correct solve rollouts. The refinement stage is always generated by the current policy π_θ (never replayed from a stale buffer), so these added trajectories are fresh samples.

Why self-distilling feedback does not collapse to mode-seeking. A natural concern is that recycling verified trajectories could over-reward already-correct chains and bias the solver toward a narrower mode. Two mechanisms prevent this. (i) Advantages are normalized within each stage's group via Eq. (2), so the appended verified t_2 traces are ranked relative to the augmented solve-stage group rather than adding an absolute reward bonus on top of own-policy correct rollouts. (ii) Refinement traces conditioned on a wrong t_1^* are informationally distinct from the solver's own correct rollouts: they encode the rewriting that turns a specific failure mode into a correct chain, which the solver has not yet produced one-shot. Empirically, removing self-distilling feedback substantially reduces the gain over vanilla GRPO (Sec. 4.6), the opposite of what a mode-seeking shortcut would predict.

C.8 HYPERPARAMETERS

Training pipeline and hyperparameters. Our training pipeline is built on veRL (Sheng et al., 2025) and uses the same fixed SR-GRPO recipe across all three backbones (App. C.1). We use the AdamW optimizer with a constant learning rate of 1×10^{-6} and a global batch size of 24 queries per update, and sample a solve group of size $G_1{=}8$ and a refinement group of size $G_2{=}8$ per query for each SR-GRPO step (the refinement group is conditioned on a single wrong-first solve trace per query). The KL coefficient is $\beta{=}0.01$ and the format reward weight is $\lambda{=}0.1$. Both the solve and refinement stages use rollout temperature 0.7 and top-p 0.9, with maximum output length $15{,}360$ tokens per stage; the model's total context window (input plus output) is $40{,}960$ tokens. We train for one epoch (688 steps) on DAPO-Math-17k (App. C.10); the same recipe is applied unchanged to all three backbones.

C.9 Computation Resources

Training uses 8×A6000 Pro GPUs with fully sharded data parallelism via veRL (Sheng et al., 2025). End-to-end wall-clock time is approximately 5–6 days for the 1.5B backbones (DeepSeek-R1-Distill-Qwen-1.5B, OpenMath-Nemotron-1.5B) and 10–11 days for the Qwen3-4B backbone, the 4B run being slower primarily because of its longer rollout-generation time at inference. For evaluation, a separate GPU runs vLLM for batch generation with tensor parallelism.

C.10 TRAINING DATA

We train on the DAPO-Math-17k corpus (Yu et al., 2025): 17K competition-math problems with integer answers, sourced from the web and official competition homepages. We use the released set as-is, without modification or augmentation, and start RL directly from the backbone weights with no SFT warm-up. This is the same training recipe as our primary baseline JustRL (He et al., 2025), ensuring that any difference between SR-GRPO and JustRL is attributable to the self-refinement objective rather than to data or initialization.

C.11 LLM-AS-JUDGE PROTOCOL

We describe the pairwise LLM-as-judge protocol that produces Figure 2. The judge is asked one focused question, *which response demonstrates better reasoning capability*, and may answer `A`, `B`, or `Tie`.

Judge. We use Claude Opus 4.7 (Anthropic, 2026) at temperature 0 for deterministic adjudication. Each judgment is a single-turn API call that receives the problem statement and two candidate reasoning paths (full chain of thought + final answer).

What is compared. For each (backbone, benchmark) pair we pair the SR-GRPO reasoning path against the matched RL baseline's path on the *same* problem and the *same* decoding seed.

Bias controls. We apply two standard mitigations to remove well-known sources of LLM-judge bias:

1. **Double-blind to model identity.** The two responses are presented only as `Response A` and `Response B`; no model name, training-method label, or system prompt that could identify SR-GRPO vs. GRPO appears anywhere in the judge prompt.
2. **Position randomization.** For every pair we randomly shuffle the order of the two responses before sending them to the judge, canceling the well-documented left/first-position preference of LLM judges.

Scoring. For a given (backbone, benchmark) pair with N paired problems, the win/tie/loss rates reported in Figure 2 are the empirical fractions of judgments that resolve to SR-GRPO, `Tie`, or the GRPO baseline, after un-shuffling the order back to the underlying model. We judge one paired response per problem on each benchmark.

Why a single judge is sufficient here. We rely on a single judge model rather than a panel because of three properties of the comparison. (i) Judge capability: Claude Opus 4.7 is a current top-tier reasoning-grade model (Anthropic, 2026), so judgment quality on math chains is not the binding bottleneck. (ii) Bias controls: the two most documented failure modes of LLM judges, position bias and identity bias, are removed by construction through position randomization and double-blind anonymization, so the residual judge noise is small. (iii) Effect size: across all nine (backbone, benchmark) pairs in Figure 2, the SR-GRPO win rate is at least 50.7% and exceeds the corresponding loss rate by at least 13 percentage points, with the largest gaps reaching 46–58 points. These win-versus-loss gaps are substantially larger than their binomial standard errors, leaving ample headroom for residual single-judge variance. Inter-judge agreement and human spot-checks would further strengthen the protocol and are an obvious extension; the LLM-judge results are supporting evidence for SR-GRPO's reasoning quality rather than the headline accuracy claim.

C.12 Reproducibility Details

Code and Models. Upon acceptance we will release: the complete training codebase with veRL integration, refinement prompt template, and reward functions; training scripts and configuration files for all experiments and ablations; the best-performing model checkpoints (merged Hugging Face format); and evaluation scripts for all nine benchmarks with deterministic seeds.

Data. All training data is drawn from publicly available sources. Evaluation benchmarks are publicly available.

D Additional Experiments

This appendix collects the supplementary quantitative and qualitative results referenced from the main paper.

D.1 Full Main Results Table

For completeness, Table 5 reports the same nine math benchmarks as Table 1 of the main paper, adding the recent VibeThinker-1.5B preprint as an additional 1.5B reference (built on a different backbone family, included for context rather than as a head-to-head baseline), an *extended baseline* block reporting Critique-GRPO at 7–8B backbones, and an optional test-time self-refinement row at the 4B backbone. Throughout this paper the **Avg.** column is the unweighted arithmetic mean across the benchmarks shown (no weighting by per-benchmark item count). Numbers for rows shared with Table 1 are unchanged; bold/underline markers are re-computed across all rows of this expanded table, including the optional test-time self-refinement row.

Crossing the supervision boundary: SR-GRPO at 4B dominates Critique-GRPO at 8B without ground-truth supervision. Critique-GRPO is the closest published methodological peer to SR-GRPO, but its strongest reported configuration operates with stronger supervision: chain-of-thought critiques are produced by an *external GPT-4o critic*, and any critique disagreeing with the rule-based ground-truth verifier is regenerated until it aligns (Zhang et al., 2025), so only ground-truth-consistent critiques enter training. SR-GRPO uses only the binary verifier, with no external critic and no ground-truth-conditioned feedback. Despite this strict information disadvantage, at the 4B Qwen3 backbone SR-GRPO outperforms Critique-GRPO at 8B on *every* one of the five shared math benchmarks (MATH-500, AIME 2024, AMC 2023, Minerva, Olympiad-Bench). At the smaller 1.5B scale, both SR-GRPO backbones already surpass Critique-GRPO's best reported 8B row on average across the five shared math benchmarks, each winning on four of the five (Nemotron's exception is Minerva; DeepSeek's exception is AIME 2024). SR-GRPO is therefore competitive with or ahead of Critique-GRPO on these matched benchmarks despite using a strictly weaker supervision signal.

On the six benchmarks reported by the Sober Reasoning Leaderboard (Hochlehnert et al., 2025), our 1.5B SR-GRPO (Nemotron) also surpasses the recent VibeThinker-1.5B preprint on every one of these benchmarks. Together with the 4B comparison against the 20B+ suite in the main paper, this shows that SR-GRPO is competitive with or surpasses the strongest publicly reported reasoners at the 1.5B and 4B scales we test. Finally, optional test-time self-refinement on the 4B model further improves the average, with no benchmark dropping. This is additional evidence that the trained refinement capability remains usable at deployment.

Table 5: **Full main results table**, extending Table 1 with: (i) the recent VibeThinker-1.5B preprint (Xu et al., 2025) as an additional 1.5B reference (numbers from the Sober Reasoning Leaderboard (Hochlehnert et al., 2025)); (ii) an extended-baseline block reporting Critique-GRPO (Zhang et al., 2025), which requires *ground-truth-conditioned* critiques from an external GPT-4o critic (supervision stronger than the binary verifier SR-GRPO uses; Critique-GRPO numbers are taken directly from Zhang et al. (2025), pass@1 as reported); and (iii) an additional row applying optional test-time self-refinement to our 4B model (one extra refinement pass at additional inference cost). **Bold**/underline: best/second across all rows. "—" marks benchmarks not reported by the source; our rows are pass@1 accuracy averaged across 4 seeds per benchmark.

Model	**Size**	MATH 500	AIME 2024	AIME 2025	AMC 2023	Minerva Math	Olympiad Bench	BRUMO 2025	CMIMC 2025	HMMT 2025	**Avg.**
Our SR-GRPO vs. vanilla GRPO across three settings											
DeepSeek-R1-Distill-Qwen-1.5B (DeepSeek-AI, 2025)	1.5B	85.6	30.0	22.5	66.9	36.1	46.1	31.7	11.9	14.2	38.3
+ GRPO (JustRL-DeepSeek-1.5B) (He et al., 2025)	1.5B	91.8	52.5	38.3	91.3	51.3	68.0	51.7	25.6	22.5	54.8
+ **SR-GRPO (Ours)**	1.5B	94.3	61.7	55.8	95.6	60.3	72.6	62.5	33.1	34.2	63.3
OpenMath-Nemotron-1.5B (Moshkov et al., 2025)	1.5B	92.8	59.2	48.3	91.3	27.6	72.0	60.8	30.6	30.8	57.0
+ GRPO (JustRL-Nemotron-1.5B) (He et al., 2025)	1.5B	94.5	69.2	63.3	96.3	31.1	76.1	67.5	41.9	40.8	64.5
+ **SR-GRPO (Ours)**	1.5B	97.1	78.3	72.5	98.1	42.3	79.4	70.8	51.3	52.5	71.4
Qwen3-4B-Thinking-2507 (Qwen Team, 2025a)	4B	97.8	78.3	80.8	<u>98.8</u>	69.7	80.8	68.3	51.9	49.2	75.1
+ GRPO (Shao et al., 2024)	4B	97.9	81.7	79.2	**99.4**	69.4	81.5	70.8	54.4	50.0	76.0
+ **SR-GRPO (Ours)**	4B	<u>98.7</u>	<u>89.2</u>	<u>82.5</u>	**99.4**	<u>72.2</u>	<u>85.2</u>	<u>79.2</u>	<u>58.8</u>	<u>55.8</u>	<u>80.1</u>
+ **SR-GRPO + Test-Time Self-Refinement (Ours)**	4B	**98.8**	**90.8**	**84.2**	**99.4**	**73.2**	**87.2**	**89.2**	**61.9**	**57.5**	**82.5**
Extended baselines (Critique-GRPO at 7–8B with GPT-4o-generated CoT critiques; numbers from Zhang et al. (2025))											
Qwen2.5-7B-Base + Critique-GRPO (Zhang et al., 2025)	7B	77.8	20.0	—	62.5	36.8	42.4	—	—	—	—
Qwen2.5-Math-7B-Base + Critique-GRPO (Zhang et al., 2025)	7B	84.2	30.0	—	70.0	40.1	48.6	—	—	—	—
Qwen3-8B (thinking) + Critique-GRPO (Zhang et al., 2025)	8B	92.0	63.3	—	92.5	52.9	66.8	—	—	—	—
Current open-source state of the art at 1.5B											
DeepScaleR-1.5B (Tan et al., 2026)	1.5B	89.4	41.7	29.2	74.4	40.3	53.0	39.2	20.6	21.7	45.5
STILL-3-1.5B (Chen et al., 2025)	1.5B	84.7	40.8	34.2	73.1	46.9	55.3	36.7	10.6	12.5	43.9
BroRL-1.5B (Hu et al., 2025)	1.5B	92.5	58.3	37.5	88.8	52.4	61.8	41.7	21.3	15.8	52.2
QuestA-Nemotron-1.5B (Li et al., 2026)	1.5B	93.3	71.7	62.5	93.1	32.8	73.3	66.7	41.3	40.8	63.9
VibeThinker-1.5B (Xu et al., 2025)	1.5B	96.7	75.7	71.3	96.5	41.5	77.2	—	—	—	—
Open-source large-scale reasoners (20B+; numbers from Hochlehnert et al. (2025))											
DeepSeek-R1-Distill-Qwen-32B (DeepSeek-AI, 2025)	32B	95.1	67.0	55.3	96.8	45.1	73.8	—	—	—	—
INTELLECT-2 (Prime Intellect Team et al., 2025)	32B	97.0	75.2	66.3	96.0	49.8	78.0	—	—	—	—
OpenThinker2-32B (Guha et al., 2025)	32B	96.1	71.3	61.7	95.8	46.6	75.5	—	—	—	—
TinyR1-32B-Preview (Sun et al., 2025)	32B	97.2	73.3	66.0	97.8	48.7	76.3	—	—	—	—
Light-R1-32B-DS (Wen et al., 2025)	32B	97.3	76.7	68.7	97.0	48.2	76.9	—	—	—	—
gpt-oss-20b (medium) (OpenAI, 2025)	20B	96.5	75.7	72.7	97.0	45.1	77.0	—	—	—	—
QwQ-32B (Qwen Team, 2025b)	32B	97.5	76.3	69.0	96.2	49.0	78.1	—	—	—	—

D.2 COMPUTE-MATCHED COMPARISON

SR-GRPO extends each rollout with a refinement group, so a training iteration draws $G_1 + G_2 = 16$ rollouts against the $G_1 = 8$ of single-stage GRPO. This appendix asks whether the gain is explained by that larger per-step sampling budget alone.

Budget-matched control at 4B. We run vanilla GRPO with double the group size ($G = 16$) on Qwen3-4B-Thinking-2507, so that the per-step sampling budget is essentially identical to ours; the control regenerates where SR-GRPO solves and then self-refines. Table 6 reports the result. Regenerating moves vanilla GRPO only marginally ($75.1 \rightarrow 76.2$ on average), while the two-stage solve-and-self-refine update delivers a lift several times larger ($75.1 \rightarrow 80.1$) and leads on every benchmark, including the hardest competitions (AIME 2024/2025, BRUMO 2025, CMIMC 2025, HMMT 2025). Self-refinement and re-sampling are therefore not interchangeable: conditioning on the solver's own wrong attempt yields gains that doubling fresh samples does not. The refiner is trained mainly to repair errors, since the signal against corrupting a correct answer arises only in the all-correct fallback; at 1.5B, removing the trained refinement stage erases most of the gain (Table 4).

Table 6: **Budget-matched control at 4B.** Pass@1 accuracy (%) on Qwen3-4B-Thinking-2507. The control is vanilla GRPO with double the group size (G=16), matching SR-GRPO's 8 solve + 8 refine rollouts per step. **Bold**: best.

Qwen3-4B-Thinking-2507	MATH 500	AIME 2024	AIME 2025	AMC 2023	Minerva Math	Olympiad Bench	BRUMO 2025	CMIMC 2025	HMMT 2025	**Avg.**
Base	97.8	78.3	80.8	98.8	69.7	80.8	68.3	51.9	49.2	75.1
+ GRPO (double group size, G=16)	98.0	80.8	80.0	98.8	69.6	82.0	71.7	53.8	50.8	76.2
+ SR-GRPO (Ours)	**98.7**	**89.2**	**82.5**	**99.4**	**72.2**	**85.2**	**79.2**	**58.8**	**55.8**	**80.1**

Training steps at 1.5B. At 1.5B, SR-GRPO is trained for one epoch (App. C.8), fewer steps than the JustRL baseline, so its gain there does not come from extra training compute either.

D.3 Advantage Collapse Statistics

GRPO produces no learning signal on a prompt whose solve group contains no correct trace, since all group-relative advantages in Eq. (2) are then identical. SR-GRPO narrows this failure mode: on prompts with no correct solve, its advantage collapses only if the refiner also fails, a strict subset of the all-incorrect prompts on which single-stage GRPO collapses (groups in which every solve is correct give no solve-stage signal in either method).

We measured this during training on the 4B backbone. Of all prompt groups, 23.8% had no correct solve but at least one correct refinement. Consequently the advantage collapses on only 2.4% of prompt groups under SR-GRPO (both stages failing entirely), against 13.7% for GRPO under the same sampling budget ($G{=}16$). These rates are consistent with the accuracies in Table 6: the budget-matched control reaches 76.2 while SR-GRPO reaches 80.1, so SR-GRPO does not reduce to budget-matched GRPO.

D.4 Effectiveness of Training for Self-Refinement: Trained vs. Untrained Models

SR-GRPO is a *training-time* method: all headline results in Table 1 are reported at first-stage output only, at exactly the same inference cost as the matched RL baseline. This appendix asks a complementary question: if a user is willing to pay for an extra refinement pass at inference, does our trained policy gain more from it than an untrained one? Sec. 4.2 already noted that the same self-refinement prompt lifts SR-GRPO's average on *every* math benchmark while yielding only a marginal lift on the matched RL baseline. Table 7 reports the full per-benchmark numbers, which deliver two complementary findings.

(i) Training is what makes the prompt work. At the same backbone, same data, and identical inference-time prompt, the model trained with the SR-GRPO objective gains substantially more from inference-time self-refinement than the GRPO baseline, which is not trained to self-refine, improving on every benchmark while the baseline shows mixed effects. The asymmetry directly attributes the gain to the training objective rather than the prompt itself, consistent with prior negative findings on prompting-only self-correction (Huang et al., 2024; Havrilla et al., 2024).

(ii) Additional test-time gains at extra inference cost. The training-time gains are this paper's primary claim. Because the refinement prompt requires neither ground truth nor an external critic, the same prompt is also directly deployable: spending extra inference compute yields a further lift on top of SR-GRPO's first-stage numbers, with no benchmark dropping. We report these test-time numbers as an additional finding; test-time scaling is not the focus of this work.

Table 7: **Test-time self-refinement: trained vs. untrained models.** Pass@1 accuracy (%) averaged across 4 seeds per benchmark for the matched RL baseline (*not* trained for self-refinement) and SR-GRPO (*trained* with a self-refinement objective), with and without invoking the same self-refinement prompt at inference (one extra refinement pass at additional inference cost). The same prompt yields a much larger average lift on SR-GRPO than on the GRPO baseline, and improves SR-GRPO on *every* one of the nine benchmarks while the GRPO baseline shows mixed effects, consistent with prior findings on prompting alone (Huang et al., 2024; Havrilla et al., 2024). Backbone: DeepSeek-R1-Distill-Qwen-1.5B. **Bold**/underline: best/second across all rows.

Configuration	MATH 500	AIME 2024	AIME 2025	AMC 2023	Minerva Math	Olympiad Bench	BRUMO 2025	CMIMC 2025	HMMT 2025	Avg.
JustRL-DeepSeek-1.5B (*untrained for self-refinement*)	91.8	52.5	38.3	91.3	51.3	68.0	51.7	25.6	22.5	54.8
+ Test-Time Self-Refine	91.9	51.7	39.2	91.5	52.2	68.4	50.8	26.3	24.2	55.1
SR-GRPO (*trained for self-refinement*)	<u>94.3</u>	<u>61.7</u>	<u>55.8</u>	<u>95.6</u>	<u>60.3</u>	<u>72.6</u>	<u>62.5</u>	<u>33.1</u>	<u>34.2</u>	<u>63.3</u>
+ Test-Time Self-Refine	**94.7**	**63.3**	**58.4**	**96.9**	**63.2**	**74.1**	**64.2**	**35.6**	**35.8**	**65.1**

Together these two observations match the design intent of SR-GRPO: a training-time method that internalizes the self-refinement capability into the policy weights (so that headline gains hold under standard first-stage evaluation at the same inference cost as the matched RL baseline), while leaving the same trained capability fully available at inference for any deployer willing to trade inference compute for extra accuracy.

D.5 Standard Errors and Statistical Significance

The headline numbers in Tables 1 and 2 are pass@1 averaged over 4 different evaluation seeds for every (backbone, benchmark) pair. To rule out the concern that SR-GRPO's gains might reflect seed-level variance rather than a genuine training-objective effect, Table 8 reports the per-benchmark standard error of the mean (SE=$\sigma/\sqrt{n}$, n=4) for every backbone on all nine math benchmarks of Table 1 plus the two OOD benchmarks of Table 2.

Table 8: **Per-Benchmark Standard Errors.** Pass@1 accuracy (%) reported as score$_{\pm\text{SE}}$, where SE is the standard error of the mean across n=4 seeds (one decimal place for both score and SE). Same models as Table 1 (math) and Table 2 (OOD). The + GRPO row in each backbone block is the matched-data GRPO baseline; the + SR-GRPO row is ours; the + SR-GRPO + Test-Time row applies an optional test-time self-refinement pass (App. D.4).

Model	MATH 500	AIME 2024	AIME 2025	AMC 2023	Minerva Math	Olympiad Bench	BRUMO 2025	CMIMC 2025	HMMT 2025	GPQA Diamond	MMLU Pro
DeepSeek-R1-Distill-Qwen-1.5B	$85.6_{\pm0.6}$	$30.0_{\pm3.0}$	$22.5_{\pm3.7}$	$66.9_{\pm3.3}$	$36.1_{\pm1.5}$	$46.1_{\pm1.0}$	$31.7_{\pm2.9}$	$11.9_{\pm2.6}$	$14.2_{\pm2.5}$	$34.7_{\pm1.5}$	$38.3_{\pm0.8}$
+ GRPO (JustRL-DeepSeek-1.5B)	$91.8_{\pm0.4}$	$52.5_{\pm2.5}$	$38.3_{\pm2.9}$	$91.3_{\pm2.2}$	$51.3_{\pm1.1}$	$68.0_{\pm0.7}$	$51.7_{\pm2.2}$	$25.6_{\pm1.9}$	$22.5_{\pm2.1}$	$40.5_{\pm1.8}$	$44.7_{\pm0.9}$
+ SR-GRPO (Ours)	$94.3_{\pm0.3}$	$61.7_{\pm2.2}$	$55.8_{\pm2.1}$	$95.6_{\pm1.6}$	$60.3_{\pm0.8}$	$72.6_{\pm0.6}$	$62.5_{\pm2.1}$	$33.1_{\pm1.6}$	$34.2_{\pm0.8}$	$46.3_{\pm1.1}$	$47.1_{\pm0.9}$
OpenMath-Nemotron-1.5B	$92.8_{\pm0.6}$	$59.2_{\pm2.5}$	$48.3_{\pm2.2}$	$91.3_{\pm2.2}$	$27.6_{\pm1.1}$	$72.0_{\pm1.1}$	$60.8_{\pm2.5}$	$30.6_{\pm1.6}$	$30.8_{\pm2.1}$	$21.5_{\pm1.4}$	$26.8_{\pm1.1}$
+ GRPO (JustRL-Nemotron-1.5B)	$94.5_{\pm0.5}$	$69.2_{\pm2.1}$	$63.3_{\pm2.7}$	$96.3_{\pm0.7}$	$31.1_{\pm0.7}$	$76.1_{\pm1.0}$	$67.5_{\pm2.1}$	$41.9_{\pm1.9}$	$40.8_{\pm2.5}$	$27.1_{\pm1.2}$	$28.9_{\pm0.6}$
+ SR-GRPO (Ours)	$97.1_{\pm0.2}$	$78.3_{\pm1.0}$	$72.5_{\pm2.1}$	$98.1_{\pm0.6}$	$42.3_{\pm0.9}$	$79.4_{\pm0.9}$	$70.8_{\pm2.1}$	$51.3_{\pm0.7}$	$52.5_{\pm2.1}$	$33.6_{\pm0.8}$	$31.8_{\pm0.6}$
Qwen3-4B-Thinking-2507	$97.8_{\pm0.3}$	$78.3_{\pm2.9}$	$80.8_{\pm3.4}$	$98.8_{\pm0.7}$	$69.7_{\pm1.3}$	$80.8_{\pm1.2}$	$68.3_{\pm2.2}$	$51.9_{\pm1.6}$	$49.2_{\pm0.8}$	$64.9_{\pm1.5}$	$73.3_{\pm0.5}$
+ GRPO	$97.9_{\pm0.4}$	$81.7_{\pm2.2}$	$79.2_{\pm2.1}$	$99.4_{\pm0.6}$	$69.4_{\pm1.9}$	$81.5_{\pm0.8}$	$70.8_{\pm2.5}$	$54.4_{\pm0.6}$	$50.0_{\pm1.9}$	$65.3_{\pm0.8}$	$73.6_{\pm0.5}$
+ SR-GRPO (Ours)	$98.7_{\pm0.4}$	$89.2_{\pm2.1}$	$82.5_{\pm2.1}$	$99.4_{\pm0.6}$	$72.2_{\pm1.7}$	$85.2_{\pm0.9}$	$79.2_{\pm2.1}$	$58.8_{\pm1.6}$	$55.8_{\pm0.8}$	$68.8_{\pm1.3}$	$74.7_{\pm1.0}$
+ SR-GRPO + Test-Time	$98.8_{\pm0.3}$	$90.8_{\pm2.5}$	$84.2_{\pm2.5}$	$99.4_{\pm0.6}$	$73.2_{\pm1.2}$	$87.2_{\pm1.1}$	$89.2_{\pm0.8}$	$61.9_{\pm0.6}$	$57.5_{\pm2.1}$	$69.2_{\pm0.6}$	$75.1_{\pm0.5}$

The per-benchmark standard errors in Table 8 are small relative to SR-GRPO's gains over the matched RL baseline on nearly every benchmark, across all three backbones, and no benchmark shows a regression. SR-GRPO further matches or exceeds the GRPO baseline on every benchmark of Tables 1 and 2, and this across-benchmark consistency together with cross-backbone generality across three backbone families rules out seed-level variance as an explanation for the headline improvements.

D.6 LLM-AS-JUDGE ON OUT-OF-DISTRIBUTION BENCHMARKS

Sec. 4.4 scores reasoning-path quality on three reasoning-dense math benchmarks. Because the out-of-distribution gains of Sec. 4.3 could in principle come from surface effects such as longer chains or better answer formatting rather than better reasoning, we repeat the same pairwise, double-blind protocol (Claude Opus 4.7 (Anthropic, 2026), identical prompt and anonymization) on the two OOD benchmarks, using the DeepSeek-R1-Distill-Qwen-1.5B backbone. Table 9 reports the outcome: SR-GRPO is preferred on both benchmarks.

Table 9: **LLM-as-judge on the OOD benchmarks** (DeepSeek-R1-Distill-Qwen-1.5B backbone). Pairwise, double-blind preference of SR-GRPO reasoning paths against the vanilla GRPO baseline.

LLM-as-judge (Ours vs. GRPO)	**Win (ours)**	**Tie**	**Loss**
GPQA Diamond	**60.6%**	7.6%	31.8%
MMLU-Pro	**56.5%**	7.4%	36.1%

We also checked the two surface confounds directly. On the MMLU-Pro evaluation the two models show essentially the same format accuracy (99.8% for GRPO versus 99.6% for SR-GRPO) and comparable mean response length ($\sim$18.4k versus $\sim$19.5k characters), so formatting and chain length are unlikely to explain the preference gap.

D.7 Qualitative Examples

We present five additional reasoning paths spanning all three backbone families. Three are at the matched DeepSeek-R1-Distill-Qwen-1.5B backbone: two ours-vs-baseline comparisons against JustRL-DeepSeek-1.5B (App. D.7.1, D.7.3) and one within-policy refinement-save (App. D.7.2). The fourth is at the OpenMath-Nemotron-1.5B backbone and compares SR-GRPO against the matched JustRL-Nemotron-1.5B SOTA RL baseline (App. D.7.4). The fifth is at the larger Qwen3-4B-Thinking-2507 backbone and compares SR-GRPO against the base reasoner directly (App. D.7.5). Each example was selected from a held-out evaluation benchmark and verified by the same symbolic answer checker used at training and evaluation; extracted answers and quoted phrases below are exactly those produced by the trained policies.

D.7.1 SR-GRPO First Stage vs. SOTA RL Baseline on a Coupled-Cubic Algebra Problem

Table 10: Comparison of JustRL-DeepSeek-1.5B (SOTA RL method) vs. SR-GRPO first stage on a coupled-cubic algebra problem (Olympiad-Bench).

JustRL-DeepSeek-1.5B (SOTA RL method)	**SR-GRPO (Ours)**
Problem: Let a and b be real numbers such that $a^3 - 15a^2 + 20a - 50 = 0$ and $8b^3 - 60b^2 - 290b + 2575 = 0$. Compute $a + b$. **Ground Truth:** $\frac{15}{2}$	
"First, we substitute $b{=}5{-}a$ into the second equation…" [expands $8(5{-}a)^3{-}60(5{-}a)^2{-}290(5{-}a){+}2575$ and uses the first equation to eliminate a^3…] "$60a^2{-}450a{-}225{=}0$. Divide by 15: $4a^2{-}30a{-}15{=}0$. Solve: $a{=}\frac{30\pm\sqrt{1140}}{8}$." ***"However, we know that if $b{=}5{-}a$, then $a{+}b{=}5$."*** *[the irrational discriminant $\sqrt{1140}$ contradicts the linear ansatz; commits anyway.]*	*"Next, we transform the first equation by substituting $a{=}z{+}5$… the equation simplifies to $z^3{-}55z{-}200{=}0$." [and analogously $b{=}y{+}\frac{5}{2}$ gives $y^3{-}55y{+}200{=}0$.]* ***"We observe that if $z{=}{-}u$, then substituting into the first equation gives $-u^3{+}55u{-}200{=}0 \implies u^3{-}55u{+}200{=}0$"*** (matches the second), so $z{=}{-}y$. *"$a{+}b{=}(5{-}u){+}(u{+}2.5){=}7.5{=}\frac{15}{2}$."*
Answer: $\boxed{5}$ ✗	**Answer:** $\boxed{\frac{15}{2}}$ ✓

This is a first-stage solve, with no refinement pass invoked, so the contrast is what the trained reasoner does at the same inference cost as the GRPO baseline. The JustRL chain locks onto the linear ansatz $b{=}5{-}a$ (suggested by the constant terms) and carries out the algebra carefully enough to expose its own inconsistency: the residual $4a^2{-}30a{-}15{=}0$ has irrational roots, contradicting the ansatz. The chain nevertheless commits to $a{+}b{=}5$ rather than discarding the ansatz. SR-GRPO instead applies the standard depression substitution to both cubics (removing the quadratic term) and then recognizes that the two depressed cubics are related by $z \leftrightarrow -y$. With $a = z + 5$ and $b = y + \frac{5}{2}$ (so $b = -z + \frac{5}{2}$), the cubic terms cancel and the sum reduces to $a + b = 5 + \frac{5}{2} = \frac{15}{2}$. The self-distilling feedback condition (Definition 1) reinforces this kind of internally consistent chain at training time: verified refinements of contradictory chains are recycled into the solve-stage group. Chains whose final answer is supported by the derivation already carry positive solve-stage advantage under the standard GRPO update.

D.7.2 Refinement Saves a Direction-Reversed Unit Conversion on Minerva

Table 11: Refinement saves a flawed first stage on Minerva (wavelength conversion to angstroms).

SR-GRPO (Ours)	**SR-GRPO + Test-Time Self-Refinement (Ours)**
Problem: For light with a wavelength $\lambda{=}408\ \mathrm{nm}$, determine the wavelength in angstroms. **Ground Truth:** 4080	
"The wavelength given is $408\,nm$. To convert this to angstroms, I know that 1 angstrom (Å) is equal to $0.1\,nm$. Therefore, I can divide the wavelength in nanometers by 10 to get the wavelength in angstroms." ***"So, $408\,nm$ divided by 10 equals $40.8\,Å$."*** *[direction reversed: $1\ \mathrm{nm}{=}10\,Å$, so multiplication is required.]* **Answer:** $\boxed{40.8}$ ✗	***"…looking back at the previous solution, the user converted $408\,nm$ to angstroms by dividing by 10, resulting in $40.8\,Å$. That was incorrect…"*** ***"The conversion factor is: $1\,Å{=}0.1\ \mathrm{nm} \Rightarrow 1\ \mathrm{nm}{=}10\,Å$. Hence, $408\ \mathrm{nm} \times 10\,Å/\mathrm{nm}{=}4080\,Å$."*** "$\lambda{=}408{\times}10^{-9}\ \mathrm{m}$ *and* $1\,Å{=}10^{-10}\ \mathrm{m}$, *so* $\lambda{=}(408{\times}10^{-9})/(10^{-10}){=}4080\,Å$." **Answer:** $\boxed{4080}$ ✓

In the first-stage output, the model makes a single-step conversion-direction inversion: it correctly recalls the order-of-magnitude relation between the angstrom and the nanometer but applies it in the wrong direction. Invoking the test-time self-refinement pass does not re-derive the underlying physics from scratch; it names the prior wrong step verbatim ("*the user converted* $408\,nm$ *to angstroms by dividing by* $10\ldots$ *that was incorrect*"), re-states the correct factor, and re-checks against the SI definition $1\ \text{Å}{=}10^{-10}\ \mathrm{m}$. This is the operational signature of the locate-and-revise pattern of Sec. 4.5: locate the specific step that needs revision and rewrite only that step. Cases of this single-step locate-and-revise pattern recur in the test-time-self-refinement wins reported in Table 7 on benchmarks where first-stage errors concentrate at a single identifiable arithmetic or conversion step.

D.7.3 SR-GRPO First Stage vs. SOTA RL Baseline on a Counting Problem with Hidden Over-Counting

Table 12: Comparison of JustRL-DeepSeek-1.5B (SOTA RL method) vs. SR-GRPO first stage on a counting problem with hidden over-counting (Olympiad-Bench).

JustRL-DeepSeek-1.5B (SOTA RL method)	**SR-GRPO (Ours)**
Problem: Compute the number of ordered pairs of integers (a, b) with $1<a \leq 50$, $1<b \leq 50$, and $\log_b a$ rational.	
Ground Truth: 81	
"$\log_b a$ is rational iff $a=c^m$, $b=c^n$ for some common base c. For each c, count $\lfloor \log_c 50 \rfloor^2$ pairs of exponents." *"c=2: 5^2=25; c=3: 3^2=9; c=4-7: 2^2=4 each; c=8-50: 1 each (43 terms)."* ***"Sum:*** $25+9+4+4+4+4+43=93$***."*** *[double-counts: $(4, 16)=(2^2, 2^4)$ at c=2 and $(4^1, 4^2)$ at c=4.]*	*"Number of coprime pairs (c, d) with $1 \leq c, d \leq C$ is $\sum_{k=1}^{C} \mu(k)\lfloor C/k \rfloor^2$ (Möbius inclusion-exclusion)."* *"m=2, C=5: 19; m=3, C=3: 7; m=4, 5, 6, 7, C=2: 3* ***each;*** *m=8-50, C=1: 1* ***each."*** *"****Total*** $= 19+7+3+3+3+3+43=81$***."***
Answer: $\boxed{93}$ ✗	**Answer:** $\boxed{81}$ ✓

This is again a first-stage solve, with no refinement pass invoked. The baseline's enumeration is locally correct: each $(a, b)=(c^m, c^n)$ pair satisfies $\log_b a=m/n$, but the baseline counts the same ordered pair multiple times, once per common base c. SR-GRPO instead reframes the count to avoid duplicates from the start: it sums over every base m and uses Möbius inversion to count only coprime exponent pairs (c, d), so each ordered $(a, b) = (m^c, m^d)$ is counted exactly once. The accuracy gap on this problem is therefore not arithmetic but combinatorial: a clean count requires recognizing that the natural enumeration is many-to-one and correcting for it. This is consistent with the same self-distilling-feedback effect seen in App. D.7.1: refinement-stage corrections that expose double-counting are recycled into the solve-stage group, shaping the solver to prefer counts that account for the many-to-one enumeration over the first plausible one.

D.7.4 OpenMath-Nemotron-1.5B Backbone: SR-GRPO First Stage vs. SOTA RL Baseline on Nested Radicals

To confirm the operational signature is not specific to a single backbone, we add a representative case at the OpenMath-Nemotron-1.5B backbone, comparing SR-GRPO against JustRL-Nemotron-1.5B (the matched SOTA RL baseline at this backbone, used throughout Tables 1 and 2).

Table 13: Comparison of JustRL-Nemotron-1.5B (SOTA RL method) vs. SR-GRPO first stage on a nested-radical problem (Olympiad-Bench). The baseline chain mis-counts the nested radicals as three square roots stacked on top of a T-th root and applies one extra halving of the exponent, landing on 10. SR-GRPO reads the expression as written, with two square roots over a T-th root, and reduces the total exponent to a closed form $(T-1)/4$.

JustRL-Nemotron-1.5B (SOTA RL method)	**SR-GRPO (Ours)**
Problem: Let $T=9$. Compute $\sqrt{\sqrt{\sqrt[T]{10^{T^2-T}}}}$. **Ground Truth:** 100	
"$T^2-T = 81-9 = 72$, so the innermost expression is 10^{72}. The T-th root: $(10^{72})^{1/9} = 10^8$." *"First square root: $\sqrt{10^8} = 10^4$. Second square root: $\sqrt{10^4} = 10^2$."* ***"Then, take the square root one more time: $\sqrt{10^2} = 10^1 = 10$."*** *[over-counts: the expression has two outer $\sqrt{}$ symbols and one $\sqrt[T]{}$, not three $\sqrt{}$ + one $\sqrt[T]{}$; one extra halving is applied.]* **Answer:** $\boxed{10}$ ✗	*"The T-th root of 10^{T^2-T} is $(10^{T^2-T})^{1/T} = 10^{(T^2-T)/T} = 10^{T-1}$ (since $(T^2-T)/T = T-1$)."* ***"Then square-rooting twice halves the exponent twice: $10^{T-1} \to 10^{(T-1)/2} \to 10^{(T-1)/4}$. With $T=9$: $(T-1)/4 = 2$, so the value is $10^2 = 100$."*** **Answer:** $\boxed{100}$ ✓

The two chains agree on every arithmetic step ($T^2-T = 72$, $72/9 = 8$, $8/2 = 4$, $4/2 = 2$); they disagree only on *how many* halvings to apply. The baseline chain reads the expression $\sqrt{\sqrt{\sqrt[T]{10^{T^2-T}}}}$ as a stack of three square roots above the T-th root and halves the exponent three times, ending at 10^1. SR-GRPO reduces the same expression to the closed form $10^{(T-1)/4}$ and evaluates once. Beyond the specific saving on this problem, the structural pattern is the same as in the DeepSeek cases above and the 4B case below: SR-GRPO prefers a short closed-form rewrite over a longer step-by-step unfolding, and that rewrite immediately exposes the correct number of operations to apply.

D.7.5 QWEN3-4B-THINKING-2507 BACKBONE: SR-GRPO VS. THE BASE REASONER

The qualitative cases above (App. D.7.1–D.7.4) use the two 1.5B backbones, where the baseline is JustRL, the strongest publicly reported 1.5B GRPO recipe at our matched-data setting. To round out the qualitative picture at the larger backbone, we add a representative case at the Qwen3-4B-Thinking-2507 backbone where SR-GRPO improves over the *base* reasoner. This is a stricter setting: Qwen3-4B-Thinking-2507 is already a strong, chain-of-thought-tuned 4B model whose first-pass output is typically careful and well-organized, and the comparison here is not against a vanilla GRPO baseline but against the base model itself.

Table 14: Comparison of Qwen3-4B-Thinking-2507 *base* vs. SR-GRPO (4B) on an equilateral-hexagon area problem (Olympiad-Bench). The base model brute-forces with a coordinate setup whose BE^2 contains $\sqrt{3}$, leading to an irrational area $\frac{9(2-\sqrt{3})}{2}$. SR-GRPO recognizes that $\triangle CDE$ is equilateral (forcing $\triangle BCE$ to be right-isosceles at C), reducing the problem to one Pythagorean step.

Qwen3-4B-Thinking-2507 (base)	**SR-GRPO (Ours, 4B)**
Problem: In equilateral hexagon $ABCDEF$, $\angle A{=}2\angle C{=}2\angle E{=}5\angle D{=}10\angle B{=}10\angle F$, and diagonal $BE{=}3$. Compute $[ABCDEF]$. **Ground Truth:** $\frac{9}{2}$	
"Place A at $(0,0)$ and B at $(s,0)$. Using vector geometry and the known interior angles… $E = \left(-\frac{s\sqrt{3}}{2}, \frac{s}{2}\right)$." *"Then* $BE^2 = \left(s + \frac{s\sqrt{3}}{2}\right)^2 + \left(\frac{s}{2}\right)^2 = s^2(2+\sqrt{3})$. *Given* $BE{=}3$:" $s^2 = \frac{9}{2+\sqrt{3}} = 9(2-\sqrt{3})$." *"Using shoelace,* $[ABCDEF] = \frac{1}{2}s^2 = \frac{9(2-\sqrt{3})}{2}$."	*"In* $\triangle CDE$: $CD{=}DE{=}s$ *and* $\angle D{=}60°$*…so* $\triangle CDE$ *is* ***equilateral****, hence* $CE{=}s$ *and* $\angle DCE{=}60°$." ***"At vertex*** C, $\angle BCD{=}150°$, ***so*** $\angle BCE = 150° - 60° = 90°$. ***Triangle*** BCE ***is right at*** C ***with legs*** $BC{=}CE{=}s$." *"Pythagorean:* $BE^2 = 2s^2 \Rightarrow s^2{=}\frac{9}{2}$. *The configuration gives* $[ABCDEF] = s^2$."
Answer: $\boxed{\frac{18-9\sqrt{3}}{2}}$ ✗ (≈ 1.20)	**Answer:** $\boxed{\frac{9}{2}}$ ✓

The two chains diverge at exactly the geometric-insight step. The base reasoner sets up coordinates immediately and computes BE^2 directly, which introduces $\sqrt{3}$ from the angle between sides AB and AE at the reflex vertex; the resulting s^2 is irrational, propagating into an irrational area whose shoelace verification is brittle and easy to mis-sign. SR-GRPO instead spends a single line of reasoning to identify $\triangle CDE$ as equilateral, which immediately forces $\triangle BCE$ to be right-isosceles at C and reduces BE to the Pythagorean image of s. The remaining computation collapses to a single multiplication. This is the same operational signature observed in the 1.5B cases of App. D.7.1 and D.7.3: SR-GRPO does not arrive at correct answers by being more careful in arithmetic, but by selecting an interpretation of the problem that exposes a clean structural shortcut.