CLPO: CURRICULUM LEARNING MEETS POLICY OPTIMIZATION FOR LLM REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Online reinforcement learning with verifiable rewards (RLVR) has become an effective paradigm for improving the reasoning abilities of large language models, but most methods still optimize reasoning trajectories over the static problem set, wasting rollout budget on solved or overly difficult problems. We propose **CLPO (Curriculum Learning meets Policy Optimization)**, a self-evolving curriculum framework that uses on-policy rollout accuracy to identify solved, medium-difficulty, and hard problems, then restructures selected tasks according to the model's current capability. Hard problems are simplified to become learnable, while medium-difficulty problems are diversified to provide useful training variation. This allows the learning curriculum to co-evolve with the policy rather than remaining fixed as the model's capability boundary shifts. Rather than treating these rewrites as static data augmentation, CLPO optimizes restructuring trajectories with credit assigned by the downstream accuracy gain of the rewritten problem, requiring no additional human annotations beyond the original verifiable answers. Experiments across mathematical reasoning and out-of-domain general reasoning benchmarks show that CLPO substantially outperforms GRPO and DAPO on Qwen3-8B by 10.21 and 7.75 average points, respectively. Ablation studies on math and code domains further show that both the restructuring mode and the rewriting loss contribute to the final gains, demonstrating that CLPO provides a scalable and robust pathway for eliciting stronger reasoning capabilities through a self-evolving curriculum.

1 INTRODUCTION

Large Language Models (LLMs) (Brown et al., 2020; Touvron et al., 2023; Achiam et al., 2023) have made substantial progress, but complex multi-step reasoning tasks such as mathematics, scientific problem solving, and code generation remain challenging. Reinforcement Learning with Verifiable Rewards (RLVR) (Wen et al., 2025; Shao et al., 2024) has emerged as an effective post-training paradigm for these domains: models sample their own solutions, receive automatic feedback from verifiers, and improve from the resulting outcomes (Guo et al., 2025; Silver & Sutton, 2025). As most RLVR methods optimize reasoning trajectories over a static task pool, when tasks are either too simple or too difficult for the current model, the resulting rollouts provide limited learning signal and make it difficult to acquire effective reasoning capabilities (Yu et al., 2025a). Existing approaches partially address this issue by enhancing exploration and sampling strategies (Yu et al., 2025a; Cheng et al., 2025; Cui et al., 2025b; Wang et al., 2025a), or by incorporating external guidance via critique models or offline expert trajectories (Zhang et al., 2025; Yan et al., 2025). However, these methods focus on eliciting better reasoning behaviors from the model on the given tasks, rather than actively aligning the task distribution around the model's current capability boundary.

In our work, we explore how to restructure the static training task pool to align with the model's evolving capability during RL. This raises two key challenges: (i) **Dynamic Capability Boundary**: As RL training progresses, a problem that is challenging in the early stage may become trivial after several policy updates, while an initially unsolvable problem may transition into the learnable zone Liu et al. (2026); Huang et al. (2026). Therefore, both which tasks should be restructured and how they should be restructured must be decided dynamically. (ii) **Lack of Reshaping Anchors**: A high-quality restructured problem should accurately reflect the intended difficulty and fall within the model's learnable region. However, unconstrained LLM-based rewriting can easily produce problems

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.

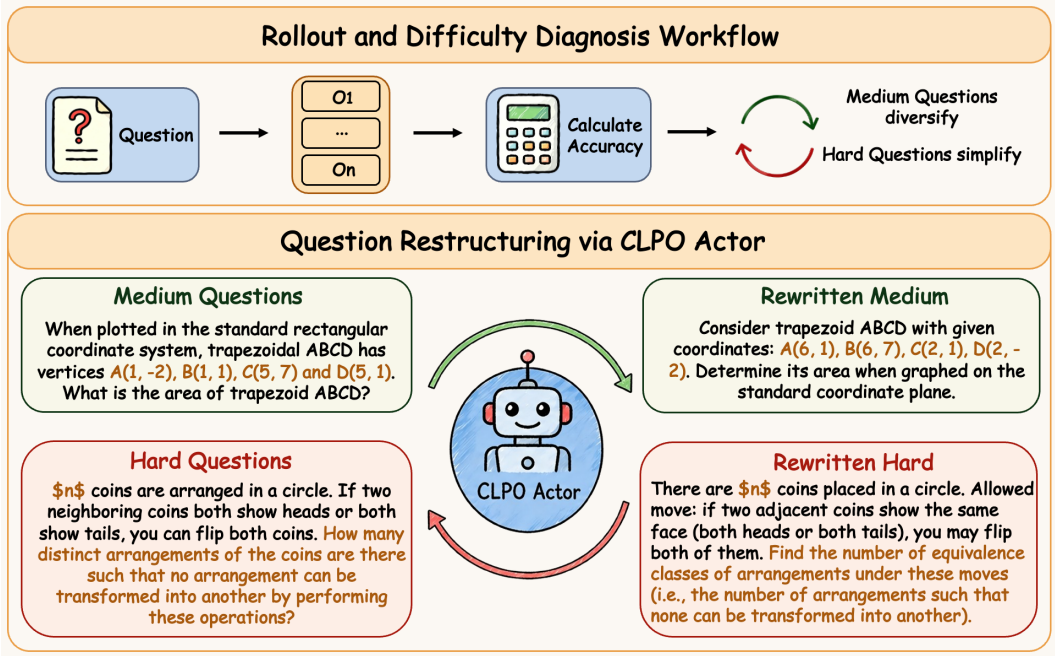


Figure 1: CLPO transforms the static training task pool into a dynamic curriculum in two stages: (1) diagnosing problem difficulty during rollout, and (2) restructuring questions to diversify medium-level tasks and simplify hard ones.

with poorly controlled difficulty, degraded quality, or collapsed diversity Shumailov et al. (2023); Yu et al. (2025b). These two challenges make it difficult for LLMs to reshape the tasks in a scalable and robust way.

To overcome these challenges, we propose **CLPO (Curriculum Learning meets Policy Optimization)**, a self-evolving curriculum framework for RLVR. As shown in Figure 1, CLPO first performs **online difficulty diagnosis** to estimate each problem's difficulty from on-policy accuracy. Second, it performs **adaptive problem restructuring**: hard problems are simplified to become learnable, while medium-difficulty problems are diversified to improve robustness around the model's capability frontier. To ensure the quality of restructured problems, CLPO further incorporates **restructuring-aware policy optimization**, which treats rewriting as an additional policy behavior alongside task solving. Specifically, it reinforces beneficial rewriting actions when the restructured problem moves into the model's informative learning region.

We validate CLPO against various SFT-based and RL-based baselines on mathematical and general reasoning benchmarks. On Qwen3-8B (Yang et al., 2025), CLPO improves over strong SFT and RLVR baselines across benchmarks including MATH-500 (Hendrycks et al., 2021), Minerva-Math (Lewkowycz et al., 2022), Olympiad (He et al., 2024), AMC23, AIME2024 (Li et al., 2024), TheoremQA (Chen et al., 2023), GPQA Diamond (Rein et al., 2024), and MMLU Pro (Wang et al., 2024). We further extend the evaluation to code generation domain, fixed wall-clock budgets, test-time scaling, and various model scales. The results consistently demonstrate the effectiveness of CLPO over existing mainstream training methods. Our main contributions are:

- We identify that existing RLVR methods fail to align the task distribution with the model's dynamic capability boundary. We therefore propose a self-evolving curriculum framework for RLVR to use rollout accuracy as a signal to restructure training problems into the model's current learning region.
- We introduce adaptive problem restructuring, which simplifies hard problems and diversifies medium-difficulty problems to create more informative training instances.
- We design a restructuring-aware optimization objective that assigns learning credit to problem-transformation trajectories based on downstream accuracy improvement.

- We provide broad empirical validation across mathematical reasoning, general reasoning, and code generation domains, as well as under fixed-budget training, test-time scaling, and various model-scale settings. On Qwen3-8B, CLPO achieves an average score of **72.25**, improving over Critique-GRPO (CoT) by **5.48** points and DAPO by **7.75** points. Further ablation studies quantify the effectiveness of each component in CLPO.

2 Related Work

Adaptive online reinforcement learning. RLVR enables language models to improve from verifier feedback through online policy optimization (Wen et al., 2025; Schulman et al., 2017; Shao et al., 2024). Adaptive approaches include dynamic sampling in DAPO (Yu et al., 2025a), sample filtering in GFPO (Shrivastava et al., 2025), sequence-level importance ratios in GSPO (Zheng et al., 2025), Pass@k optimization in PKPO (Walder & Karkhanis, 2025), and outcome-guided trust regions in ETR (Zhang et al., 2026). CLPO instead learns problem transformations from online difficulty signals. Closely related, SvS (Liang et al., 2025) synthesizes answer-preserving variants to sustain exploration. CLPO combines difficulty-dependent simplification and diversification with a downstream-accuracy rewriting objective. Our ablations isolate these components while retaining online augmentation when the rewriting objective is removed; they do not establish superiority over SvS without a matched comparison.

External guidance. LUFFY (Yan et al., 2025) incorporates offline expert trajectories into online learning, while Critique-GRPO (Zhang et al., 2025) uses external linguistic feedback to guide refinement. CLPO derives its rewriting signal from the current policy's solving outcomes on original and rewritten problems, using the existing references for verification rather than an external teacher during training.

Curriculum learning. Curriculum learning adapts the presentation of training examples (Bengio et al., 2009). Recent reasoning approaches adjust context length and token budgets (Song et al., 2025; Hammoud et al., 2025), or adapt task difficulty to recent model performance (Shi et al., 2025). CLPO couples difficulty diagnosis with learnable problem restructuring: it determines not only which problems to train on, but also how to transform them into useful training instances.

3 Method

CLPO uses on-policy rollout accuracy to diagnose difficulty, restructure selected problems, and jointly train solving and rewriting behaviors (Figure 2).

3.1 Preliminary: GRPO

For a problem q with reference answer a, the rollout policy $\pi_{\theta_{\text{old}}}$ samples G responses $\{y_i\}_{i=1}^{G}$. A verifier assigns $r_i \in \{-1, 1\}$ for incorrect and correct responses, respectively. The group-relative solving advantage is

$$\hat{A}_{\text{solve},i} = \frac{r_i - \text{mean}_{j=1}^{G} r_j}{\text{std}_{j=1}^{G} r_j}. \tag{1}$$

CLPO normalizes only non-degenerate groups and optimizes solving trajectories using a clipped importance-ratio surrogate.

3.2 Online Difficulty Diagnosis

For each problem in the sampled batch, we estimate accuracy using the same rollout policy:

$$\text{Acc}(q) = \frac{1}{G} \sum_{i=1}^{G} \mathbb{I}[\text{Verifier}(y_i, a) = \text{True}]. \tag{2}$$

An active-region filter first retains only problems with $0 < \text{Acc}(q) < 1$. Thus, all-zero and all-one original groups are excluded before difficulty partitioning, rewriting, and group-relative normalization.

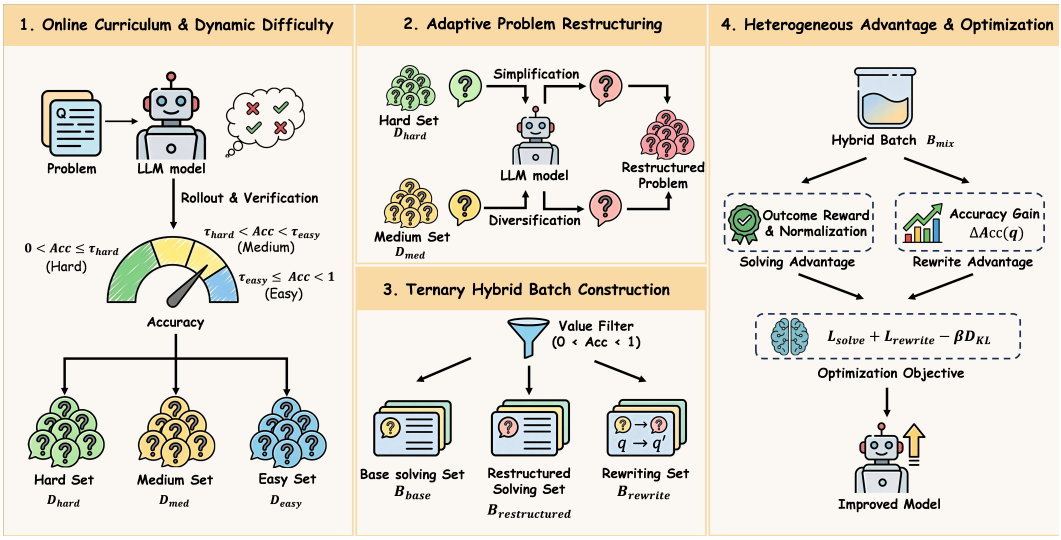


Figure 2: Overview of CLPO. Original solving groups are active-filtered before difficulty partitioning; rewritten solving groups are filtered before mixed-batch assembly. Solving and instruction-conditioned rewriting trajectories are jointly optimized.

Among active problems, define

$$k = \left\lfloor \frac{G-1}{3} \right\rfloor, \qquad \tau_{\text{hard}} = \frac{k}{G}, \qquad \tau_{\text{easy}} = \frac{G-k}{G}. \tag{3}$$

The retained problems are partitioned as

$$\mathcal{D}^{\text{hard}} = \{q : 0 < \text{Acc}(q) \leq \tau_{\text{hard}}\}, \tag{4}$$

$$\mathcal{D}^{\text{med}} = \{q : \tau_{\text{hard}} < \text{Acc}(q) < \tau_{\text{easy}}\}, \tag{5}$$

$$\mathcal{D}^{\text{easy}} = \{q : \tau_{\text{easy}} \leq \text{Acc}(q) < 1\}. \tag{6}$$

With $G = 4$, these categories correspond to one, two, and three correct responses, respectively. Here, "hard" means low but nonzero empirical accuracy. The partition is recomputed at each update to track the policy's changing capability.

3.3 Adaptive Problem Restructuring

For each selected hard or medium problem, we generate exactly one rewriting trajectory z and extract the rewritten problem q'. For math, the conditioning context is

$$c_{\text{rewrite}} = f_p(q, d, a), \qquad z \sim \pi_{\theta_{\text{old}}}(\cdot \mid c_{\text{rewrite}}), \qquad q' = \text{Extract}(z), \tag{7}$$

where f_p constructs the rewriting instruction and $d \in \{\text{hard}, \text{med}\}$ selects simplification or diversification. Hard problems are rewritten into clearer, more accessible formulations, while medium problems receive alternative formulations. Easy problems are not rewritten.

Answer-preserving augmentation. Both instructions require preserving the original answer and necessary constraints, without answer leakage or solution shortcuts (Appendix G). For math, the answer is provided to the rewriter, not the solver. Rewrites are verified against the original answer: augmentation is intended to change the formulation, not the required answer.

Handling unsuccessful rewrites. Rewrites failing extraction or format/length checks receive a rewriting signal of -1 and are not solved. For each remaining q', we sample G responses; only groups with $0 < \text{Acc}(q') < 1$ contribute to solving loss. If all responses fail, the rewrite receives negative accuracy-change feedback and no solving update. These checks do not certify semantic equivalence or solvability; Appendix C separately examines faithfulness. Evaluation uses standard solving prompts without reference answers.

3.4 Restructuring-Aware Policy Optimization

CLPO trains **solving** with correctness rewards on original and rewritten problems, and **rewriting** with downstream accuracy gains. The rewriting signal trains problem construction; it does not replace the solving reward.

Solving and rewriting advantages. Solving trajectories use the group-relative advantage in Equation 1. For a rewrite that passes extraction and format/length checks, the rewriting advantage is

$$\hat{A}_{\text{rewrite}} = \Delta\,\text{Acc}(q) = \text{Acc}(q') - \text{Acc}(q). \tag{8}$$

A positive difference encourages the rewrite, a negative value discourages it, and zero gives no rewriting advantage. With $G = 4$, improving from one correct response to two gives 0.25. Invalid rewrites receive -1. We use these advantages without further normalization or clipping and weight rewriting by $\lambda = 0.1$. Accuracy gain measures usefulness, not semantic correctness.

Mixed-batch construction. The batch B_{mix} concatenates active original solving groups, active rewritten solving groups, and all selected rewriting trajectories, including invalid rewrites. Subset sizes vary online without a fixed mixing ratio. Excluding a solving group does not remove its rewriting feedback. Appendix H details batch sizes and the training algorithm.

Joint policy update. Let x denote a generated trajectory and c_x its conditioning context: a standard solving prompt for solving trajectories, or c_{rewrite} for rewriting trajectories. Each generated token uses the trajectory's corresponding advantage $\hat{A}_x$. Define

$$\rho_{x,t}(\theta) = \frac{\pi_\theta(x_t \mid c_x, x_{<t})}{\pi_{\theta_{\text{old}}}(x_t \mid c_x, x_{<t})}, \tag{9}$$

$$\ell_{x,t}(\theta) = \min\Big(\rho_{x,t}\hat{A}_x, \text{clip}(\rho_{x,t}, 1-\epsilon, 1+\epsilon)\hat{A}_x\Big). \tag{10}$$

With $w_x = \lambda$ for rewriting and $w_x = 1$ for solving, we maximize

$$J_{\text{CLPO}}(\theta) = \frac{1}{T}\sum_{x\in B_{\text{mix}}}\sum_{t=1}^{|x|}\left[w_x\ell_{x,t}(\theta) - \beta K_{x,t}(\theta)\right], \qquad T = \sum_{x\in B_{\text{mix}}}|x|. \tag{11}$$

Here, $K_{x,t}$ regularizes the policy toward the reference model, and T counts generated, non-padding tokens. The minimized loss is $-J_{\text{CLPO}}$. All trajectories share one PPO update and one token-mean denominator, rather than separate subset normalization.

The two feedback signals therefore act on different outputs: solving advantages increase the likelihood of correct solution trajectories, whereas rewriting advantages increase the likelihood of useful problem reformulations. Both update the same policy under their respective instructions. The coefficient λ scales rewriting feedback, but its effective contribution also depends on the number and lengths of rewriting trajectories.

4 Experiments

4.1 Experimental Setup

4.1.1 Base Models

We use Qwen3-8B (Yang et al., 2025) as the default backbone for our main experiments. To study the robustness across different model scales and families, we additionally evaluate Qwen3-4B, Qwen3.5-9B, Qwen3-30B-A3B, and Llama3.1-8B in Section 4.6. All variants are evaluated using identical decoding parameters.

4.1.2 Baselines

We compare CLPO against two categories of competitive fine-tuning methods.

Table 1: Main results on Qwen3-8B. We compare CLPO against competitive SFT and RLVR baselines across eight benchmarks. CLPO obtains the highest average among the compared methods; bold marks the best score in each column.

Method	Math (In-Domain)					General Reasoning (OOD)			Avg
	MATH 500	Minerva Math	Olympiad Bench	AMC23	AIME24	MMLU Pro	Theorem QA	GPQA Diamond	
*All methods are finetuned on **Qwen3-8B***									
Supervised Fine-Tuning (SFT)									
RAFT	76.20	35.58	36.86	50.00	26.67	65.93	43.50	36.76	46.44
Refinement-FT	83.20	47.58	40.71	70.00	33.33	67.84	41.25	34.47	52.30
Critique-FT	79.00	35.23	39.64	67.50	33.33	63.16	46.00	34.84	49.84
CITL-FT	76.40	37.20	38.57	62.50	30.00	66.13	44.25	36.36	48.93
Reinforcement Learning with Verifiable Rewards (RLVR)									
GRPO	89.20	51.47	57.40	82.50	43.33	69.86	54.75	47.80	62.04
DAPO	**91.20**	53.31	63.80	87.50	46.67	70.01	55.00	48.48	64.50
LUFFY	89.40	52.94	58.80	85.00	40.00	70.34	58.25	49.49	63.03
Critique-GRPO (Simple)	89.40	52.57	60.20	87.50	40.00	70.13	59.00	48.63	63.43
Critique-GRPO (CoT)	**91.20**	61.50	63.80	**90.00**	46.67	70.98	59.50	50.50	66.77
CLPO (Ours)	90.80	**72.79**	**77.50**	**90.00**	**50.00**	**73.12**	**67.75**	**56.06**	**72.25**

Supervised Fine-Tuning (SFT). These methods derive training signal from static or offline-generated supervision, without online environment feedback. **RAFT** (Dong et al., 2023) filters rollouts by reward ranking and fine-tunes on the retained outputs. **Refinement-FT** (Chen et al., 2024) constructs training examples from iteratively refined responses guided by natural language feedback. **CITL-FT** (Xi et al., 2024) jointly fine-tunes on both initial responses and critique-conditioned refinements. **Critique-FT** (Wang et al., 2025b) directly supervises the model to produce critiques rather than final answers.

Reinforcement Learning with Verifiable Rewards (RLVR). These methods use online rollout feedback as the primary training signals. **GRPO** (Shao et al., 2024) computes group-relative advantages over sampled rollouts under a scalar correctness reward. **DAPO** (Yu et al., 2025a) combines asymmetric clipping and dynamic sampling to improve RLVR training. **LUFFY** (Yan et al., 2025) blends off-policy expert trajectories with on-policy rollouts to regularize the learned policy. **Critique-GRPO** (Zhang et al., 2025) introduces natural-language critiques as an auxiliary reward signal within the GRPO framework, which we evaluate in both simple and CoT variants. Unlike these methods, CLPO restructures problems online to control difficulty and jointly optimizes the restructuring behavior as a learnable policy component.

4.1.3 BENCHMARKS

We evaluate our models across two categories of benchmarks.

Mathematical Reasoning (In-Domain). We employ MATH-500 (Hendrycks et al., 2021), Minerva-Math (Lewkowycz et al., 2022), Olympiad Bench (He et al., 2024), AMC23, and AIME2024 (Li et al., 2024). These datasets cover a broad spectrum of difficulties, ranging from standard math competitions to advanced Olympiad levels.

General Domain Reasoning (OOD). We utilize TheoremQA (Chen et al., 2023), GPQA Diamond (Rein et al., 2024), and MMLU Pro (Wang et al., 2024) as out-of-distribution benchmarks to assess generalization beyond the mathematical training domain.

Table 1 reports single-sample Pass@1 accuracy and the average across eight benchmarks. Additional AMC23/AIME2024 evaluations use Avg@32, the mean correctness over 32 responses per problem (Appendix B).

4.1.4 Implementation Details

For mathematical reasoning, we utilize the DAPO-Math-17k dataset Yu et al. (2025a). For code generation (evaluated in the ablation studies in Section 4.3), we adopt the code-domain subset of Eurus-2-RL-Data Cui et al. (2025a). Unless otherwise specified, all compared methods use a rollout size of $G = 4$ and binary verifier rewards (± 1) under a fixed RLVR optimization budget. All experiments are conducted on a node equipped with $8\times$ NVIDIA H20 GPUs. Full hyperparameter configurations and CLPO-specific thresholds (τ_{hard}, τ_{easy}, λ) are detailed in Appendix A.

4.2 Main Results

Table 1 summarizes the performance of CLPO on Qwen3-8B. CLPO achieves the highest average score among the compared methods of 72.25%, outperforming the strongest RLVR baseline, Critique-GRPO (CoT), by 5.48 points and DAPO by 7.75 points. The performance gain is particularly large on challenging math benchmarks: CLPO improves over Critique-GRPO (CoT) on Minerva Math (+11.29), Olympiad Bench (+13.70), and AIME24 (+3.33), while maintaining top-tier performance on AMC23 and MATH 500. This suggests that restructuring-aware optimization successfully converts difficulty signals into high-quality training curriculum rather than redundant rollouts.

The superiority of CLPO extends to OOD reasoning. Across MMLU Pro, TheoremQA, and GPQA Diamond, CLPO consistently achieves the highest scores among all baselines. This generalization suggests that CLPO fosters robust reasoning capabilities rather than dataset-specific overfitting. Compared to SFT-style methods, CLPO exhibits a substantial margin (at least 19.95 average points), underscoring the necessity of verifiable online feedback for complex reasoning tasks.

4.3 Training Dynamics and Ablation Studies

Training dynamics. Figure 4 in Appendix B shows early training dynamics. The additional step-110–200 math checkpoints in Table 2 extend this analysis to the full training horizon: CLPO remains ahead at each reported late checkpoint and ends at 49.27 Avg@32, compared with 46.15 for DAPO and 44.06 for GRPO.

Late-stage training checkpoints. Table 2 records the math checkpoints from steps 110–200, complementing the first 100 steps in Figure 4. CLPO has the highest reported score at each checkpoint in this interval. The minimum gaps to DAPO and GRPO are 2.60 and 5.21, respectively; the minimum gaps to the loss, medium, and hard ablations are 1.98, 3.86, and 0.73.

Table 2: AIME2024 Avg@32 (%) at late-stage checkpoints. Ablation labels identify the removed component.

Step	CLPO	GRPO	DAPO	w/o loss	w/o medium	w/o hard
110	48.96	42.50	46.04	46.67	45.10	47.50
120	48.85	43.12	46.25	46.67	44.90	46.88
130	49.17	42.50	46.46	45.73	44.58	47.29
140	49.48	43.75	46.25	46.56	45.42	48.02
150	50.42	44.06	46.77	46.88	45.00	48.23
160	50.31	43.96	46.46	47.19	45.73	48.75
170	50.21	43.12	46.88	46.77	45.42	49.27
180	50.52	44.58	45.94	46.25	46.04	48.44
190	50.10	44.27	46.25	46.98	44.69	48.85
200	49.27	44.06	46.15	47.29	45.21	48.54

Component ablation. Table 3 reports the 200-step results. Removing the rewriting objective preserves online restructuring and solving on rewritten problems; the removed component is optimization of rewriting trajectories using downstream accuracy changes. Its deficits of 1.98 on math and 10.55 on code support the value of learning the rewriting behavior. Removing medium diversification or hard simplification also reduces performance. These gaps are not additive contributions and do not

Table 3: Component ablations after 200 steps. Math denotes AIME2024 Avg@32; code uses Pass@1.

Setting	Math Avg@32	Code Pass@1
CLPO	**49.27**	**61.04**
w/o rewriting objective	47.29	50.49
w/o medium diversification	45.21	51.37
w/o hard simplification	48.54	58.20

replace a matched comparison with SvS; they isolate components within CLPO's own augmentation pipeline.

Table 4: Repeated-sampling accuracy (%). Each problem has 32 sampled responses.

Method	AMC23 Avg@32	AIME2024 Avg@32
GRPO	84.06	44.06
DAPO	87.34	46.15
LUFFY	83.91	40.94
Critique-GRPO (Simple)	88.67	40.73
Critique-GRPO (CoT)	**91.64**	47.19
CLPO	91.25	**49.27**

Repeated sampling and rewrite validation. Table 4 reports results using 32 responses per problem. CLPO obtains 49.27 on AIME2024 versus 47.19 for Critique-GRPO (CoT), and 91.25 on AMC23 versus 91.64 for that baseline. Thus, its AIME2024 lead persists with repeated sampling, while AMC23 remains competitive. All methods share temperature 1.00, top-p 1.00, and an 8,192-token response cap. Training-time rewrite statistics show the mean signal increasing from -0.0916 to +0.0736 and the negative ratio decreasing from 21.12% to 7.00%. Appendix B provides the full tables and their scope; Appendix C separately reports an auxiliary three-judge audit of 100 Qwen3-Max-generated rewrite pairs.

Generation-budget control. All compared RLVR methods use the same 8,192-token cap during both training and evaluation. CLPO does not add a length reward or penalty; solving rewards depend on correctness. The repeated-sampling comparison therefore measures relative reasoning performance under a shared generation budget. It complements the single-sample main table without assuming that different policies have identical truncation behavior or that their ranking is invariant to the token limit.

4.4 Training Efficiency under Fixed Budgets

As shown in Table 5, CLPO demonstrates superior training efficiency under fixed wall-clock budgets. Within a 10-hour window, CLPO reaches an average of 38.02 points, a level that baselines fail to achieve even with higher update counts. This advantage persists at the 20-hour mark (43.09 avg.), proving that CLPO extracts more informative learning signals per unit of compute time.

Interestingly, the step counts highlight that performance is not strictly tied to update frequency. CLPO achieves its lead with an intermediate number of steps (43 and 88), positioned between the high-throughput GRPO and the lower-throughput DAPO. This validates that the computational investment in online problem restructuring yields a superior "return on investment" compared to baseline RLVR methods, turning available time into more robust reasoning capabilities.

4.5 Test-Time Scaling

Figure 5 in Appendix B demonstrates that CLPO scales effectively with test-time compute, reaching 70.00%, 83.33%, and 73.33% accuracy on AIME 2024–2026 at 64 samples. Notably, on AIME 2025, CLPO achieves a 30.0-point gain over its Pass@1 baseline, consistently outperforming GRPO and DAPO. These results indicate that CLPO enhances the policy's *searchability* by producing a higher density of correct reasoning paths within the solution distribution.

Table 5: Time-budget comparison on AIME benchmarks. CLPO achieves the best average performance under both 10-hour and 20-hour budgets, showing stronger training efficiency than GRPO and DAPO.

Method	Time Budget	Steps	AIME24	AIME25	AIME26	Avg.
GRPO	10h	53	40.73	35.21	34.06	36.67
	20h	107	43.96	41.77	40.00	41.91
DAPO	10h	24	38.23	36.77	34.27	36.42
	20h	52	41.98	42.40	**40.73**	41.70
CLPO	10h	43	**41.35**	**37.08**	**35.63**	**38.02**
	20h	88	**45.94**	**42.92**	40.42	**43.09**

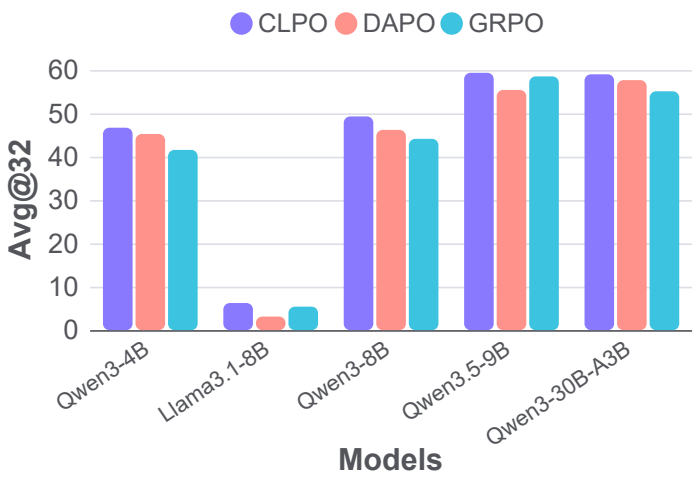


Figure 3: Model-scale comparison across representative base models on AIME24.

4.6 Model-Scale Analysis

Figure 3 investigates the generalizability of CLPO across various base-model scales and architectures on AIME24. CLPO consistently achieves the highest performance on all tested backbones, outperforming both DAPO and GRPO across: Qwen3-4B (46.7% vs. 45.3%/41.6%), Llama3.1-8B (6.3% vs. 3.1%/5.4%), Qwen3-8B (49.3% vs. 46.2%/44.1%), Qwen3.5-9B (59.4% vs. 55.4%/58.5%), and Qwen3-30B-A3B (59.0% vs. 57.7%/55.1%).

The model-scale comparison suggests that CLPO complements scaling rather than specializing to one checkpoint. Larger models generally have stronger initial reasoning ability, but CLPO can still convert intermediate online signals into useful training examples.

5 Conclusion

We present CLPO, a self-evolving curriculum framework for RLVR that uses rollout accuracy to diagnose current competence, restructure problems around the productive learning zone, and optimize restructuring behavior through downstream accuracy improvement. By turning the model from a passive solver into an active curriculum constructor, CLPO achieves consistent gains over strong RLVR baselines.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pp. 41–48, 2009.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Angelica Chen, Jérémy Scheurer, Jon Ander Campos, Tomasz Korbak, Jun Shern Chan, Samuel R Bowman, Kyunghyun Cho, and Ethan Perez. Learning from natural language feedback. *Transactions on machine learning research*, 2024.

Wenhu Chen, Ming Yin, Max Ku, Pan Lu, Yixin Wan, Xueguang Ma, Jianyu Xu, Xinyi Wang, and Tony Xia. Theoremqa: A theorem-driven question answering dataset. *arXiv preprint arXiv:2305.12524*, 2023.

Daixuan Cheng, Shaohan Huang, Xuekai Zhu, Bo Dai, Wayne Xin Zhao, Zhenliang Zhang, and Furu Wei. Reasoning with exploration: An entropy perspective. *arXiv preprint arXiv:2506.14758*, 2025.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Yuchen Zhang, Jiacheng Chen, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025a.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, et al. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*, 2025b.

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. *arXiv preprint arXiv:2304.06767*, 2023.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Hasan Abed Al Kader Hammoud, Kumail Alhamoud, Abed Hammoud, Elie Bou-Zeid, Marzyeh Ghassemi, and Bernard Ghanem. Train long, think short: Curriculum learning for efficient reasoning. *arXiv preprint arXiv:2508.08940*, 2025.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. *arXiv preprint arXiv:2402.14008*, 2024.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. R-zero: Self-evolving reasoning llm from zero data, 2026. URL https://arxiv.org/abs/2508.05004.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, et al. Solving quantitative reasoning problems with language models. *Advances in neural information processing systems*, 35:3843–3857, 2022.

Jia Li, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Huang, Kashif Rasul, Longhui Yu, Albert Q Jiang, Ziju Shen, et al. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13(9):9, 2024.

Xiao Liang, Zhongzhi Li, Yeyun Gong, Yelong Shen, Ying Nian Wu, Zhijiang Guo, and Weizhu Chen. Beyond pass@ 1: Self-play with variational problem synthesis sustains rlvr. *arXiv preprint arXiv:2508.14029*, 2025.

Shiyu Liu, Yongjing Yin, Jianhao Yan, Yunbo Tang, Qinggang Zhang, Bei Li, Xin Chen, Jingang Wang, Xunliang Cai, and Jinsong Su. Bapo: Boundary-aware policy optimization for reliable agentic search. *arXiv preprint arXiv:2601.11037*, 2026.

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. In *First Conference on Language Modeling*, 2024.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

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*, pp. 1279–1297, 2025.

Taiwei Shi, Yiyang Wu, Linxin Song, Tianyi Zhou, and Jieyu Zhao. Efficient reinforcement finetuning via adaptive curriculum learning. *arXiv preprint arXiv:2504.05520*, 2025.

Vaishnavi Shrivastava, Ahmed Awadallah, Vidhisha Balachandran, Shivam Garg, Harkirat Behl, and Dimitris Papailiopoulos. Sample more to think less: Group filtered policy optimization for concise reasoning. *arXiv preprint arXiv:2508.09726*, 2025.

Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Yarin Gal, Nicolas Papernot, and Ross Anderson. The curse of recursion: Training on generated data makes models forget. *arXiv preprint arXiv:2305.17493*, 2023.

David Silver and Richard S Sutton. Welcome to the era of experience. *Google AI*, 1, 2025.

Mingyang Song, Mao Zheng, Zheng Li, Wenjie Yang, Xuan Luo, Yue Pan, and Feng Zhang. Fastcurl: Curriculum reinforcement learning with stage-wise context scaling for efficient training r1-like reasoning models. *arXiv preprint arXiv:2503.17287*, 2025.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Christian Walder and Deep Karkhanis. Pass@ k policy optimization: Solving harder reinforcement learning problems. *arXiv preprint arXiv:2505.15201*, 2025.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, et al. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*, 2025a.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37: 95266–95290, 2024.

Yubo Wang, Xiang Yue, and Wenhu Chen. Critique fine-tuning: Learning to critique is more effective than learning to imitate. *arXiv preprint arXiv:2501.17703*, 2025b.

Xumeng Wen, Zihan Liu, Shun Zheng, Zhijian Xu, Shengyu Ye, Zhirong Wu, Xiao Liang, Yang Wang, Junjie Li, Ziming Miao, et al. Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base llms. *arXiv preprint arXiv:2506.14245*, 2025.

Zhiheng Xi, Dingwen Yang, Jixuan Huang, Jiafu Tang, Guanyu Li, Yiwen Ding, Wei He, Boyang Hong, Shihan Do, Wenyu Zhan, et al. Enhancing llm reasoning via critique models with test-time and training-time supervision. *arXiv preprint arXiv:2411.16579*, 2024.

Jianhao Yan, Yafu Li, Zican Hu, Zhi Wang, Ganqu Cui, Xiaoye Qu, Yu Cheng, and Yue Zhang. Learning to reason under off-policy guidance. *arXiv preprint arXiv:2504.14945*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025a.

Wenhao Yu, Zhenwen Liang, Chengsong Huang, Kishan Panaganti, Tianqing Fang, Haitao Mi, and Dong Yu. Guided self-evolving llms with minimal human supervision, 2025b. URL https://arxiv.org/abs/2512.02472.

Shijie Zhang, Kevin Zhang, Zheyuan Gu, Xiang Guo, Rujun Guo, Shaoyu Liu, Guanjun Jiang, and Xiaozhao Wang. Etr: Outcome-guided elastic trust regions for policy optimization. *arXiv preprint arXiv:2601.03723*, 2026.

Xiaoying Zhang, Hao Sun, Yipeng Zhang, Kaituo Feng, Chaochao Lu, Chao Yang, and Helen Meng. Critique-grpo: Advancing llm reasoning with natural language and numerical feedback. *arXiv preprint arXiv:2506.03106*, 2025.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*, 2025.

A Implementation Details

All runs use verl (Sheng et al., 2025) with vLLM (Kwon et al., 2023) for rollout generation. The main backbone is Qwen3-8B; the scale study additionally evaluates Qwen3-4B, Qwen3.5-9B, Qwen3-30B-A3B, and Llama3.1-8B-Instruct. We use identical decoding settings across compared methods and report optimization steps under the 10-hour and 20-hour training budgets.

Table 6: Training data and experimental configuration.

Setting	Value
Data and environment	
Math training data	DAPO-Math-17k
Code training data	Code subset of Eurus-2-RL-Data
Code sources	APPS, CodeContests, TACO, Codeforces
Code split	25,276 training / 1,024 validation examples
Hardware	$8\times$ NVIDIA H20
Framework	verl with vLLM rollout generation
Optimization	
Training steps	200
Optimizer	AdamW
Learning rate	10^{-6}, constant
Global batch size	64
Rollouts per prompt (G)	4
Correct / incorrect reward	$+1$ / -1
KL coefficient (β)	0.001
Gradient clipping	1.0
Generation: training and evaluation	
Temperature	1.00
Top-p	1.00
Maximum response length	8,192 tokens

Table 7: CLPO-specific settings. Difficulty categories apply only to active groups.

Setting	Value	Meaning
Hard threshold τ_{hard}	0.25	One correct rollout when $G = 4$.
Easy threshold τ_{easy}	0.75	Three correct rollouts when $G = 4$.
Rewrite weight λ	0.1	Weight on the rewriting surrogate.
Active-region filter	$(0, 1)$	Excludes all-zero and all-one solving groups.

We use the official train/validation split of the code subset of Eurus-2-RL-Data.[1] The benchmark suite and mixed-batch construction are specified in Sections 4 and 3.

[1] https://huggingface.co/datasets/PRIME-RL/Eurus-2-RL-Data

B Additional Evaluation and Training Analysis

B.1 Repeated-sampling evaluation

We sample 32 responses for each of the 40 AMC23 and 30 AIME2024 problems. With correctness indicator $c_{q,s}$, the reported metric is $\mathrm{Avg@32} = 100(32N)^{-1} \sum_{q=1}^{N} \sum_{s=1}^{32} c_{q,s}$. It estimates average single-response accuracy, not oracle Pass@32. Temperature, top-p, and maximum response length are 1.00, 1.00, and 8,192, respectively, for every method. Table 4 reports the additional evaluation.

CLPO obtains the highest AIME2024 Avg@32, with margins of 2.08 over Critique-GRPO (CoT) and 3.12 over DAPO, and is competitive on AMC23 (91.25 versus 91.64). The AIME2024 ranking persists under repeated sampling. This evaluation reduces dependence on a single sampled response, but does not estimate variation across independent training seeds or enlarge the number of benchmark problems.

B.2 Training-time rewriting statistics

Table 8 summarizes the reported arithmetic means of per-step rewriting statistics. Hard and medium columns count selected problems per update; their sum is the rewriting trajectory count. These training-time statistics are distinct from the text-level faithfulness audit.

Table 8: Reported training-time rewriting statistics. The mean signal is not a semantic-faithfulness score.

Steps	Avg. hard	Avg. medium	Mean signal	Positive (%)	Negative (%)
1–50	27.44	5.62	-0.0916	24.69	21.12
51–100	17.96	4.20	+0.0267	26.71	10.13
101–150	14.72	4.16	+0.0641	28.37	8.52
151–200	15.94	3.57	+0.0736	29.00	7.00

Mean signal increases from -0.0916 to +0.0736 and the negative ratio decreases from 21.12% to 7.00%. Average rewriting trajectory counts are 33.06, 22.16, 18.88, and 19.51 over the four intervals. These statistics describe the changing usefulness of rewriting actions, separately from the text-level audit below.

Table 9: Finer-grained rewriting statistics over the first 100 steps, with the same definitions as Table 8.

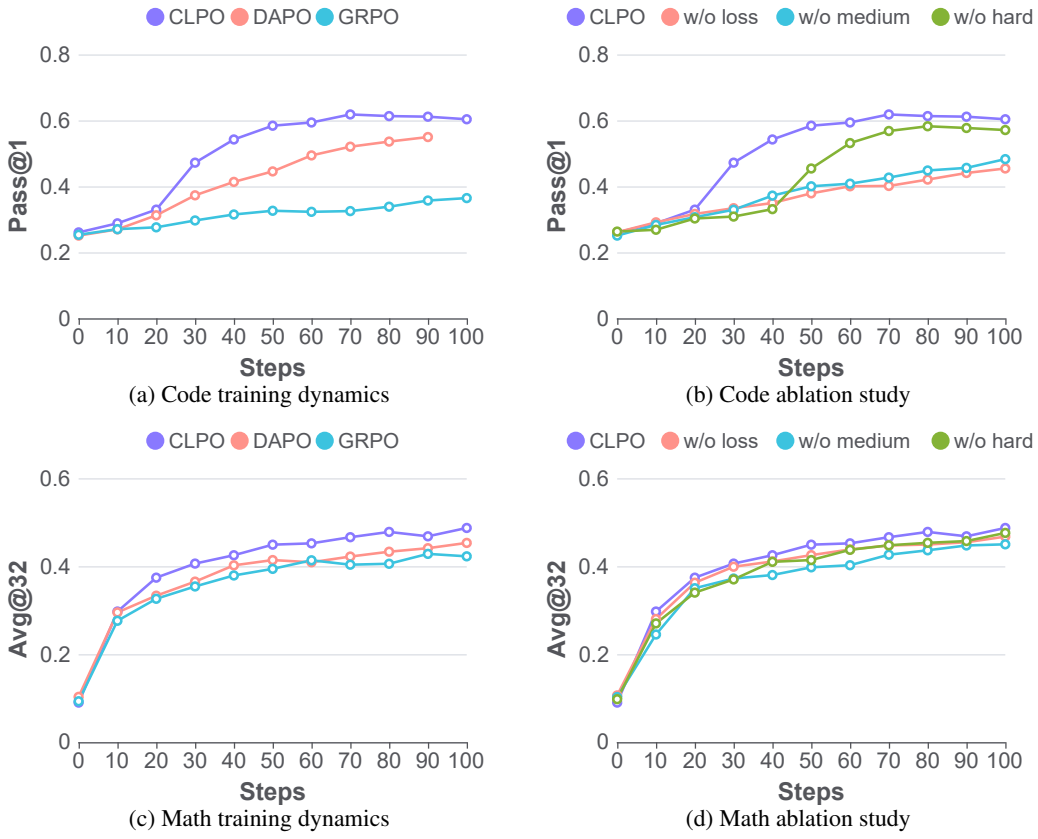
Steps	Avg. hard	Avg. medium	Mean signal	Positive (%)	Negative (%)
1–10	42.10	9.10	-0.1027	27.57	24.05
11–20	28.30	6.10	-0.1565	20.28	27.27
21–30	27.20	4.90	-0.0959	21.67	19.75
31–40	20.70	3.70	-0.0945	24.69	18.77
41–50	18.90	4.30	-0.0084	29.25	15.75
51–60	21.10	3.30	+0.0172	30.05	11.62
61–70	19.50	4.80	-0.0194	22.74	15.93
71–80	17.80	4.70	+0.0313	27.59	8.05
81–90	15.50	4.10	+0.0546	30.97	8.42
91–100	15.90	4.10	+0.0496	22.19	6.65

B.3 Rollout group size and generation cost

We use $G = 4$ for all compared RLVR methods and ablations. Increasing G gives finer and less noisy per-problem accuracy estimates, but increases generation work. For N original problems, $H + M$ selected rewrites, and V valid rewrites, CLPO generates GN original solving responses, $H + M$ rewriting trajectories, and GV rewritten solving responses per update. The actual token cost also depends on their lengths. Holding problem counts fixed, increasing G from 4 to 10 multiplies solving-response counts by 2.5; at a fixed generation budget it reduces problem coverage. This motivates

$G = 4$ as a practical choice, without establishing it as optimal. Fixed wall-clock comparisons in Table 5 account for online restructuring cost in end-to-end training, while a systematic larger-G study remains future work.

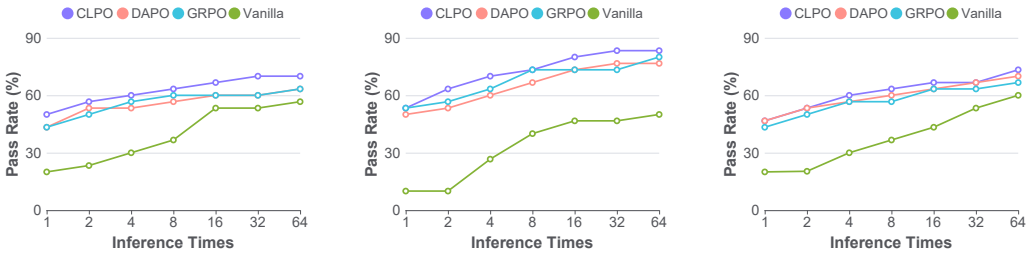
B.4 Early Training and Test-Time Scaling



(a) Code training dynamics (b) Code ablation study

(c) Math training dynamics (d) Math ablation study

Figure 4: Early training dynamics and ablations over the first 100 steps. Table 2 supplies the subsequent math checkpoints and Table 3 the 200-step ablation endpoints.



(a) AIME 2024 (b) AIME 2025 (c) AIME 2026

Figure 5: Test-time scaling of Qwen3-8B on AIME 2024, AIME 2025, and AIME 2026.

C AUXILIARY REWRITE FAITHFULNESS AUDIT

Protocol. We randomly sample 100 DAPO-Math problems and use Qwen3-Max with answer-preserving simplification instructions. These are auxiliary pairs, not outputs of the trained Qwen3-8B policy or difficulty-stratified on-policy samples. The prompt supplies the reference answer and prohibits leakage. Three external endpoints independently judge each pair with its reference answer on five binary criteria and a separate overall judgment. Each field uses a two-of-three majority vote.

Table 10: Automated audit of 100 auxiliary original–rewrite pairs (%). AP: answer preserved; CP: constraints preserved; NS: no shortcut leakage; UA: unambiguous; IS: intended simplification; OF: overall faithful. These scores concern Qwen3-Max-generated audit pairs.

Judge	Pairs	AP	CP	NS	UA	IS	OF
claude-sonnet-4-6	100	99.00	99.00	100.00	97.00	98.00	97.00
gemini-3.5-flash	100	98.00	99.00	100.00	99.00	98.00	98.00
gpt-5.1	100	92.00	92.00	100.00	97.00	92.00	91.00
Majority vote	100	98.00	98.00	100.00	98.00	97.00	97.00

The majority vote gives 98% answer preservation and 97% overall acceptability; individual overall scores range from 91% to 98%. These automated assessments are not a proof of equivalence or an estimate of the trained policy's faithfulness rate.

C.1 REPRESENTATIVE PAIR

The following example illustrates a clarification of a determinant construction. The mathematical notation below restores the row structure of the original problem.

Original problem

Define

$$D_1 = \begin{vmatrix} 1 \end{vmatrix}, \qquad D_2 = \begin{vmatrix} 1 & 1 \\ 1 & 3 \end{vmatrix}, \qquad D_3 = \begin{vmatrix} 1 & 1 & 1 \\ 1 & 3 & 3 \\ 1 & 3 & 5 \end{vmatrix}.$$

In general, for positive integer n, let the determinant D_n have 1s in every position of its first row and first column, 3s in the remaining positions of the second row and second column, 5s in the remaining positions of the third row and third column, and so forth. Find the least n such that $D_n \geq 2015$.

Rewritten problem

For each positive integer n, define the determinant D_n as follows:

- The first row and first column consist entirely of 1s.
- In the second row and second column, all entries not already set by the first row or column are 3s.
- In the third row and third column, all entries not already set by earlier rows or columns are 5s.
- This pattern continues: for each $k = 1, 2, \ldots, n$, the k-th row and k-th column have the value $2k - 1$ in all positions that have not already been filled by previous rows or columns.

For example:

$$D_1 = |1|, \qquad D_2 = \begin{vmatrix} 1 & 1 \\ 1 & 3 \end{vmatrix}, \qquad D_3 = \begin{vmatrix} 1 & 1 & 1 \\ 1 & 3 & 3 \\ 1 & 3 & 5 \end{vmatrix}.$$

Find the smallest positive integer n such that $D_n \geq 2015$.

Reference answer and inspection. The reference answer is 12. Both constructions give matrix entries $2\min(i, j) - 1$. Subtracting consecutive rows yields $D_n = 2^{n-1}$, so $D_{11} = 1024 < 2015 \leq 2048 = D_{12}$. This derivation is provided here for inspection, not in the rewrite. The rewrite retains the determinant examples and exposes neither the answer nor the determinant formula; it makes the construction rule explicit while preserving the remaining mathematical work.

D VISUALIZATION OF CURRICULUM DYNAMICS

Figure 6 shows the evolving difficulty distribution over 200 steps. Relative to GRPO, CLPO exhibits a faster decline in the hard-problem ratio and maintains more medium-difficulty problems, while its easy-problem ratio increases. These trends are consistent with a curriculum that changes with the policy's competence; the component ablations provide the corresponding controlled comparison.

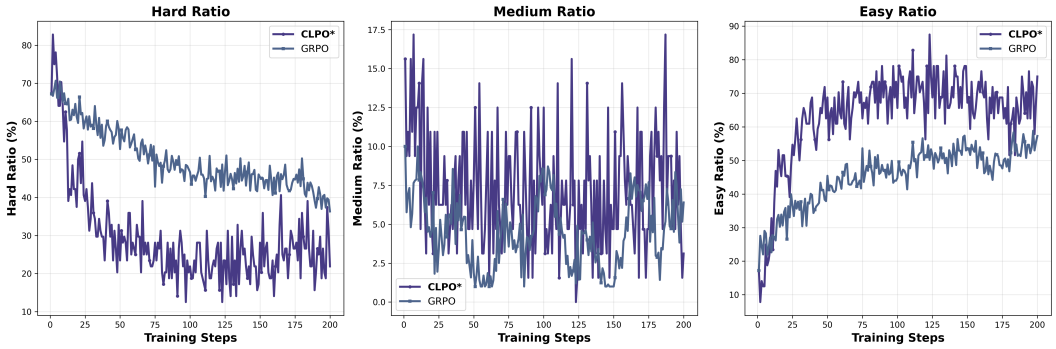


Figure 6: Hard, medium, and easy problem ratios during CLPO and GRPO training.

E LIMITATIONS AND RESEARCH STATEMENTS

E.1 LIMITATIONS

Our evaluations use verifiable math and code rewards and a shared 8,192-token response cap. Larger generation budgets, larger rollout groups, and broader open-ended tasks require further study. The auxiliary rewrite audit uses a separate generator and automated judges; it does not establish the faithfulness rate of the trained rewriting policy. A matched direct comparison with SvS and a detailed token-level cost breakdown remain open.

E.2 REPRODUCIBILITY STATEMENT

The generation, filtering, and optimization procedure is specified in Section 3 and Algorithm 1. Appendix A documents hyperparameters, decoding settings, and mixed-batch construction. Additional evaluation results, audit procedures, and restructuring prompts are provided in Appendices B, C, and G.

E.3 AI USE STATEMENT

We used LLM-based tools to assist with language editing, manuscript revision, LaTeX formatting, and preparation of evaluation scripts. The auxiliary rewrite audit in Appendix C uses Qwen3-Max to generate problem variants and three external LLMs to evaluate the original–rewrite pairs. These model judgments are reported as automated assessments rather than human annotations.

E.4 ETHICS STATEMENT

This work studies RLVR using mathematical and code-oriented datasets. Automatically generated problems and model-based judgments can contain errors; the audit in Appendix C describes the scope of the additional validation. Capability improvements should be accompanied by safety evaluations appropriate to downstream applications.

F Case Study: Medium-Difficulty Diversification

This pair illustrates the change from symbolic set definitions to natural language. Both prompts use the same solving instruction: solve step by step and place the final answer on its own line after `Answer:`. The response passages below are selected excerpts from the supplied examples; ellipses indicate omissions.

Original problem and response excerpts

Problem. Let set $A = \{n \mid n/3 \in \mathbb{N}_+\}$, and $B = \{y \mid y = x + 4 + \sqrt{5 - x^2}\}$. Find the number of elements in the set $A \cap B$.

Response excerpts. So, first, I need to figure out what elements are in set B. Then, find the common elements between A and B. Let me start by understanding set B more deeply.
...
Solving this inequality:

$$5 - x^2 \geq 0 \;\Rightarrow\; x^2 \leq 5 \;\Rightarrow\; -\sqrt{5} \leq x \leq \sqrt{5}.$$

So, x has to be between $-\sqrt{5}$ and $\sqrt{5}$.
...
Therefore, the maximum value of y is at $x = \sqrt{5/2} \approx 1.5811$, which is approximately 7.1623, and the minimum value of y occurs at the endpoints.
...
Therefore, the possible elements of A that can be in B are 3, 6. Because 9 is already larger than 7.16, so it cannot be achieved by any y.
Wait, but let me check. Let me verify if 3 and 6 are indeed attainable
...
Wait, but wait! Wait a second. I need to check if there are other values.
... **Answer: 2**

Diversified problem and response excerpts

Problem. Let A be the set of all positive integers n such that $n/3$ is a positive integer. Define B as the set of all real numbers y that can be expressed as $y = x + 4 + \sqrt{5 - x^2}$ for some real number x. Determine the number of elements in the intersection of sets A and B.

Response excerpts. So set A is all positive multiples of 3. **Got that.**
...
Therefore, x must be in the interval $[-\sqrt{5}, \sqrt{5}]$. So x is between approximately -2.236 and 2.236.
...
But maybe **it's better to first find the range** of possible y's in set B, and then check which multiples of 3 lie within that range. Let's try that approach.
...
Therefore, set B contains all real numbers y between $4 - \sqrt{5}$ and $4 + \sqrt{10}$.
...
So in that interval, only 3 and 6. Hence, the intersection of A and B has two elements. So, the answer is 2.
Answer: 2

Interpretation. Both statements define the same sets and preserve the answer. The first response revisits intermediate checks, while the second follows the interval calculation more directly. This is a qualitative illustration, not an estimate of average response efficiency or rewrite-faithfulness rates. Aggregate training effectiveness is evaluated by the component ablations.

G PROMPTS FOR ADAPTIVE PROBLEM RESTRUCTURING

The following templates specify simplification and diversification constraints.

Prompt for Hard Prompt Simplification

You are a master-level question rewriting expert. Your mission is TOP-SECRET: rewrite the given math question so that it is simpler, clearer, and more direct, while maintaining the EXACT same task, constraints, solution method, and CORRECT ANSWER. DO NOT solve the problem or make any assumptions about its solution.
CRITICAL RULES FOR OUTPUT:
- Do NOT include any reasoning, parsing, or explanation of the problem (e.g., avoid phrases like "Let me simplify this question" or "This problem can be rewritten as...").
- Rewrite the question to make it SIMPLER and CLEARER for the reader, while preserving the EXACT task, math content, and structure.
- Keep the rewritten question STRICTLY UNDER 400 tokens, including all characters, symbols, and spaces.
- Remove redundancy and completely clarify definitions, if needed. For example, if there are implied constraints, make them explicit.
- Preserve ALL formatting or directive rules exactly as in the original question (e.g., "write the answer in a box", "solve step by step").
- The rewritten question must lead to the SAME correct answer as the original answer. DO NOT explicitly, implicitly, or indirectly REVEAL or SUGGEST the answer.
HOW TO SIMPLIFY:
1) Eliminate redundant or overly verbose phrasing while keeping the exact math concepts, symbols, and structure.
2) Avoid complex clauses by breaking them into shorter, simpler sentences, but do NOT remove or alter necessary constraints.
3) Include IMPLICIT constraints explicitly if they improve clarity (e.g., "$x > 0$" if it is implied implicitly).
4) DO NOT add any background information, new context, or reasoning unrelated to the question.

Prompt for Question Diversification

You are a master-level question rewriting expert. Your mission is TOP-SECRET: rewrite the given math question into a diverse but semantically equivalent version, while maintaining the EXACT same task, constraints, solution method, and CORRECT ANSWER. DO NOT solve the problem or make any assumptions about its solution.
CRITICAL RULES FOR OUTPUT:
- Rewrite the question to DIVERSIFY its expression but PRESERVE its exact meaning and constraints.
- Keep the rewritten question STRICTLY UNDER 400 tokens, including all characters, symbols, and spaces.
- DO NOT change any math content (variables, relationships, solution method, etc.).
- Avoid repetitive sentence structures or template-like phrasing by using varied grammar structure, synonyms, or logical order.
- Preserve ALL formatting or directive rules exactly as in the original question (e.g., "write the answer in a box", "solve step by step").
- The rewritten question must lead to the SAME correct answer as the original: ANSWER. DO NOT explicitly, implicitly, or indirectly REVEAL or SUGGEST the answer.
HOW TO DIVERSIFY:
1) Rephrase clauses logically (e.g., change phrasing while maintaining meaning).
2) Rearrange sentence structure without altering mathematical intent.
3) Use synonyms or alternative phrasing to express the same logic and tasks.
4) DO NOT add or remove constraints, context, or instructions.

H Mixed-Batch Optimization Details

Each update combines three trajectory subsets. B_{base} contains active original solving groups, $B_{\text{restructured}}$ contains active rewritten solving groups, and B_{rewrite} contains one rewriting trajectory per selected hard or medium problem, including invalid rewrites. If N_{active} and N'_{active} count retained original and rewritten problems, their sizes are

$$\begin{aligned} |B_{\text{base}}| &= GN_{\text{active}}, \\ |B_{\text{restructured}}| &= GN'_{\text{active}}, \\ |B_{\text{rewrite}}| &= N_{\text{hard}} + N_{\text{med}}, \end{aligned} \tag{12}$$

where N_{hard} and N_{med} count selected hard and medium problems. There is no fixed mixing ratio or separate oversampling of rewriting trajectories.

Solving advantages are normalized within each retained solving group. Rewriting advantages instead use the raw accuracy difference, or -1 for invalid rewrites, without further normalization or clipping. Filtering a rewritten solving group does not remove its rewriting trajectory. In particular, a rewrite with zero correct responses receives negative accuracy-change feedback, while one with all correct responses receives positive feedback even though neither group enters the solving loss.

Equation 11 averages over all generated, non-padding tokens in the concatenated batch. Subsets are not normalized separately, so their effective contributions depend on both trajectory counts and generated lengths. The coefficient $\lambda = 0.1$ weights the rewriting surrogate; it does not prescribe a fixed fraction of the total update. The importance-ratio clipping in the PPO surrogate is distinct from the unmodified rewriting advantage.

Algorithm 1 CLPO training iteration

1: Sample original problems and generate G solving responses per problem.
2: Retain groups with $0 < \text{Acc}(q) < 1$ as B_{base}; partition active problems.
3: Generate one instruction-conditioned rewrite per selected hard/medium problem.
4: Keep rewriting trajectories in B_{rewrite}; assign invalid rewrites advantage -1.
5: For rewrites passing extraction and format/length checks, generate G solving responses and compute $\Delta\,\text{Acc}(q)$.
6: Retain rewritten solving groups with $0 < \text{Acc}(q') < 1$ as $B_{\text{restructured}}$.
7: Compute solving advantages for retained groups and rewriting advantages for rewriting trajectories.
8: Concatenate all three subsets and perform the joint token-mean update in Equation 11.