Walk Before You Run!
Group Length Policy Optimization for Efficient LLM Reasoning

Anonymous ACL submission

Abstract

Scaling test-time computation has improved the reasoning ability of large language models, but RL-trained reasoners often produce verbose chain-of-thought (CoT) traces that inflate inference cost with limited, and sometimes negative, accuracy returns. Existing length-control methods apply a uniform penalty regardless of whether the model has solved the problem (Wen et al., 2025; Hochlehnert et al., 2025; Hou et al., 2025; Fatemi et al., 2025), which can clip useful steps on hard problems and limit further exploration. We propose **G**roup **L**ength **P**olicy **O**ptimization (**GLPO**), a small modification to GRPO that activates a length penalty *only* on questions where all rollouts in a group are correct, and leaves the rest unconstrained. The unanimous-correctness gate serves as an implicit mastery proxy, so easy problems pick up length pressure first while harder ones stay open for exploration. On Qwen2.5-Math 1.5B and 7B trained with RL only (no SFT), the accuracy-preserving regime cuts average response length by about 11% on 1.5B and 5% on 7B, with average accuracy moving by $+1.1$ and $+0.5$ points over a tuned GRPO baseline on five competition-level math benchmarks (AIME 2024, AMC 2023, MATH 500, Minerva Math, OlympiadBench). A heavier penalty pushes length down further, by up to about 33%, at a measurable accuracy cost. Trained models also tend to swap verbose textual self-checking for compact code-based verification, suggesting that conciseness and correctness need not be in direct conflict in this setting. We will release training code and model checkpoints.

1 Introduction

Scaling test-time computation has become a major lever behind the reasoning ability of Large Language Models (LLMs) (Snell et al., 2024; Muennighoff et al., 2025). By generating long Chain-of-Thought (CoT) (Wei et al., 2022) traces before producing final answers, frontier models such as OpenAI o1 (Jaech et al., 2024) and DeepSeek-R1 (Guo et al., 2025) have reached strong performance on challenging mathematical and scientific tasks. Much of this gain comes from reinforcement learning (RL) post-training, in particular Group Relative Policy Optimization (GRPO) (Shao et al., 2024), which induces richer reasoning behaviors such as self-reflection, backtracking, and multi-step verification (Guo et al., 2025).

This reasoning ability comes at a real computational cost. Recent studies (Cuadron et al., 2025; Sui et al., 2025; Yang et al., 2025) report a pervasive *overthinking* phenomenon in RL-trained models. These models often produce verbose traces that inflate inference cost with limited gain in accuracy. Models may loop on circular logic or rederive identical intermediate results, spending thousands of tokens when the correct answer is reachable much sooner. Overthinking can also degrade accuracy itself, since prolonged chains accumulate compounding errors and may drift away from correct partial solutions on otherwise solvable problems.

The challenge is that reasoning quality and verbosity are tightly coupled. A naive length penalty risks discarding informative reasoning steps along with the redundant ones. Most existing RL-based approaches (Wen et al., 2025; Hochlehnert et al., 2025; Liu et al., 2025; Hou et al., 2025; Fatemi et al., 2025) apply length pressure uniformly across all training samples, regardless of whether the model can currently solve the problem. While this reduces overall verbosity, the uniform pressure can force the model to compress reasoning on difficult problems before it has learned to solve them, hurting accuracy and limiting the exploration needed on harder questions. Alternative methods rely on supervised fine-tuning (SFT) on distilled concise traces (Ye et al., 2025; Chu et al., 2025), but these are bounded by teacher quality and give the model little room to find its own efficient reasoning

strategy.

In this work, we revisit the assumption that length optimization should be a global constraint. We argue length pressure should turn on only after the model has shown it can solve a given problem. We introduce **Group Length Policy Optimization (GLPO)**, guided by the principle "*accuracy first; speed comes with mastery*". The rule is direct. When all G rollouts in a group are correct, GLPO activates a length-shaping reward for that question, and otherwise leaves the model alone. Using unanimous correctness as the gate is a deliberate design choice that prioritizes accuracy stability over compression rate. Softer fractional thresholds (e.g., k of G rollouts correct) are a natural variant that we touch on in the discussion.

This rule induces a mastery-gated curriculum with no need for explicit difficulty metrics, an auxiliary reward model, or a hand-designed schedule. Easy problems, on which the model is already consistently correct, pick up length pressure early and the model is rewarded for finding more concise reasoning paths. Harder problems, on which at least one rollout still fails, are left unconstrained so that the model can keep exploring. The set of compressed problems then grows naturally as the model masters more questions during training.

Our contributions are summarized as follows.

- We introduce GLPO, a small modification to GRPO that conditions length optimization on group-level correctness. Unlike prior length-penalized RL methods that apply length pressure uniformly across samples (Wen et al., 2025; Hochlehnert et al., 2025; Hou et al., 2025; Fatemi et al., 2025), the mastery gate restricts compression to questions the model has already solved.
- Empirically, GLPO yields a small but consistent Pareto-style improvement on the accuracy and length frontier. In the accuracy-preserving regime, it cuts thinking budgets by about 5–11% while slightly improving average accuracy ($+1.1$ on 1.5B and $+0.5$ on 7B over a tuned GRPO baseline). A heavier penalty pushes the length reduction up to about 33%, with a measurable but bounded drop in accuracy.
- Beyond compression, our keyword and case analyses suggest a strategy shift under GLPO. Trained models tend to replace verbose textual self-checking with shorter alternatives such as code-based verification, consistent with the view that mastery-gated length pressure need not hurt reasoning, although keyword frequency is a coarse proxy and a controlled study would be needed to make a causal claim.
- We evaluate GLPO on five competition-level math benchmarks (AIME 2024, AMC 2023, MATH-500, Minerva Math, OlympiadBench) with Qwen2.5-Math 1.5B and 7B under the zero-RL setting. We do not claim generalization beyond math reasoning or these scales, and treat broader transfer as future work.

2 Methodology

2.1 Group Relative Policy Optimization

GRPO (Shao et al., 2024) replaces the traditional learned value function with group-level advantage estimation. Given a question q, it samples G outputs and computes the advantage of each as

$$\hat{\mathbf{A}}_i = \frac{r_i - \text{mean}(\{r_1, r_2, \cdots, r_G\})}{\text{std}(\{r_1, r_2, \cdots, r_G\})}. \tag{1}$$

Like PPO, GRPO employs a clipped surrogate objective regularized by a KL penalty, optimizing the policy π_θ via

$$\begin{aligned}\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{q,\,\{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}} \frac{1}{G}\sum_{i=1}^{G}\Big\{ \\ \min\big[\tau_i(\theta)\hat{\mathbf{A}}_i,\ \text{clip}(\tau_i, 1-\varepsilon, 1+\varepsilon)\hat{\mathbf{A}}_i\big] \\ - \beta\mathbb{D}_{\text{KL}}[\pi_\theta||\pi_{\text{ref}}]\Big\}\end{aligned} \tag{2}$$

where

$$\tau_i = \frac{\pi_\theta(o_i|q)}{\pi_{\theta_{\text{old}}}(o_i|q)}. \tag{3}$$

where π_{ref} is the reference policy. The KL term prevents excessive policy drift.

2.2 Group Length Policy Optimization

Our primary objective is to enable LLMs to *progressively compress their thinking budgets* while preserving, and where possible improving, their reasoning accuracy. To this end, we introduce Group Length Policy Optimization (GLPO), guided by the principle "*accuracy first; speed comes with mastery*". The key idea is simple. By conditioning length optimization on group-level correctness, the model's thinking budget is compressed *only* on problems where it has reliably solved the question, enabling a gradual reduction in reasoning length. The unanimous gate $\sum_j r_j = G$ is the strictest such condition. Softer fractional thresholds ($\sum_j r_j \geq k$ for $k < G$) are natural variants

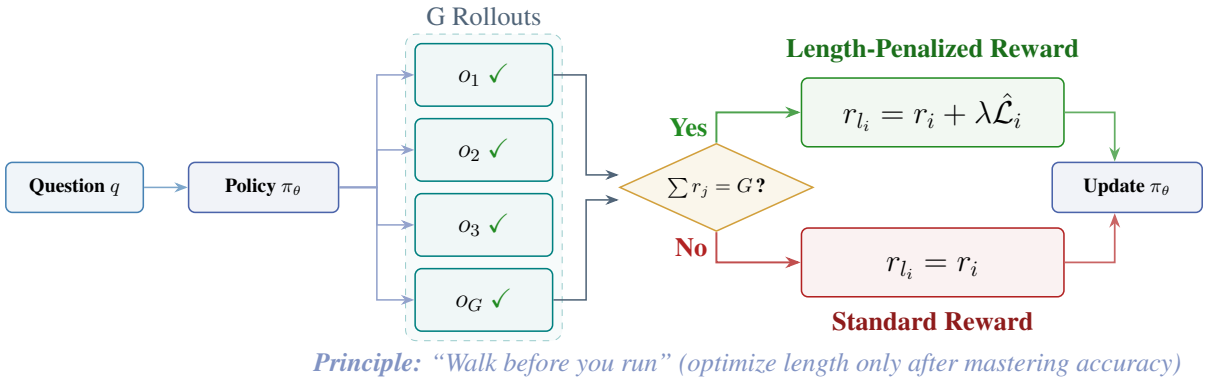

Figure 1: Overview of GLPO. The policy π_θ samples G rollout responses per question. When *all* rollouts are correct ($\sum r_j = G$), a length penalty is activated to progressively compress the thinking budget. Otherwise, only the accuracy reward is used, enforcing the "accuracy first, speed comes with mastery" principle.

that activate compression earlier on noisier signals, and we use the strict form throughout.

Prior work (Luo et al., 2025; Song et al., 2025) observes that reasoning response length is not strictly correlated with answer correctness. A lengthy CoT response does not guarantee correctness, nor does a concise one necessarily indicate an error. Empirically, correct CoT responses tend to be shorter while incorrect ones tend to be longer (Luo et al., 2025). This suggests that a near-optimal, problem-specific thinking budget may exist and can be approximated through progressive optimization.

We augment the GRPO reward as follows. When *all* rollouts for a given question are correct, the remaining context capacity is used as an additional reward signal, so that more concise reasoning earns a larger bonus:

$$\mathcal{J}_{\text{GLPO}}(\theta) = \mathbb{E}_{q,\{o_i\}\sim\pi_{\theta_{\text{old}}}}\Big[\frac{1}{G}\sum_{i=1}^{G} \min\big[\tau_i \mathbf{A}_{l_i},\ \text{clip}(\tau_i, 1-\varepsilon_l, 1+\varepsilon_h)\mathbf{A}_{l_i}\big] - \beta \mathbb{D}_{\text{KL}}[\pi_\theta \| \pi_{\text{ref}}] + \alpha \mathbb{H}(\pi_\theta)\Big] \tag{4}$$

where $\alpha\mathbb{H}(\pi_\theta)$ is an entropy bonus for exploration. The length-aware advantage $\mathbf{A}_{l_i}$ is computed as:

$$\mathbf{A}_{l_i} = \frac{r_{l_i} - \text{mean}(\{r_{l_1}, r_{l_2}, \cdots, r_{l_G}\})}{\text{std}(\{r_{l_1}, r_{l_2}, \cdots, r_{l_G}\})}. \tag{5}$$

with the length augmented reward

$$r_{l_i} = r_i + \lambda \hat{\mathcal{L}}_i, \tag{6}$$

where $\hat{\mathcal{L}}_i$ is the length reward term

$$\hat{\mathcal{L}}_i = \begin{cases} \frac{\mathcal{L}_{\text{Max}} - \mathcal{L}_i}{\mathcal{L}_{\text{Max}}}, & \text{if } \sum_{j=1}^{G} r_j = G \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Here $\mathcal{L}_i$ is the response length, $\mathcal{L}_{\text{Max}}$ the maximum context window, and $(\varepsilon_l, \varepsilon_h)$ the asymmetric clipping bounds.

2.3 Rule-Based Reward Model

To avoid reward hacking from a learned reward model, we adopt a rule-based outcome reward defined as

$$r_i(o_i, a) = \mathbb{1}[\text{is_equivalent}(o_i, a)], \tag{8}$$

where a is the ground-truth answer and o_i the predicted answer. The model must place CoT reasoning within `<think>...</think>` tags and the final answer within `<answer>...</answer>` using `\boxed{}` formatting.

3 Experiments

3.1 Training Dataset Curation

We curate a high quality training set by aggregating challenging problems from three sources to maximize difficulty and diversity.

- **DeepScaleR**[1]: Contains approximately 40K unique mathematics-specific problem-answer pairs collected from AIME (1984–2023), AMC (prior to 2023), Omni-MATH, and Still datasets (Lewkowycz et al., 2022; Gao et al., 2024; Min et al., 2024).
- **DAPO-Math-17K**[2]: Contains approximately 17K problem-answer pairs, each paired with an integer answer. This dataset was compiled from the Art of Problem Solving (AoPS) website and

[1] https://huggingface.co/datasets/agentica-org/DeepScaleR-Preview-Dataset
[2] https://huggingface.co/datasets/BytedTsinghua-SIA/DAPO-Math-17k

official competition websites using a combination of web scraping and manual annotation.

- **MATH**[3] (Levels 3–5): Contains approximately 8K problem-answer pairs. Each problem includes a step-by-step solution, useful for teaching models to generate clear explanations.

After obtaining the above datasets, we employ Math-Verify[4] to extract answers from the provided textual solutions, selecting only cases where the extracted answer matches the corresponding dataset answer. We discard any samples that are empty, incomplete, or duplicates. Ultimately, we obtain approximately 59K reasoning problems. In the first stage, we use the 59K data to incentivize the model's reasoning ability. In the second stage, we use the MATH (Levels 3–5) subset to optimize the model's reasoning length.

3.2 Training Details

We adopt Qwen2.5-Math-1.5B and Qwen2.5-Math-7B (Yang et al., 2024b) as the base language models and use verl (Sheng et al., 2025) for training. Training uses the Adam optimizer with a constant learning rate of 1×10^{-6}, a batch size of 128, and 32 rollouts per question. The maximum response length is set to 3,072 tokens. Following prior work (Yu et al., 2025), we set the clipping parameters ε_l to 0.2 and ε_h to 0.28, balancing the exploration–exploitation tradeoff in RL.

3.3 Evaluation Benchmarks

Following Liu et al. (2025); Song et al. (2025), we evaluate our models on a diverse suite of competition-level benchmarks: AIME 2024[5] (30 challenge problems), AMC 2023[6] (40 mathematical problems covering algebra, geometry, number theory, and combinatorics), Minerva Math (Lewkowycz et al., 2022), MATH-500 (Hendrycks et al., 2021), and OlympiadBench (He et al., 2024).

3.4 Evaluation Setup

We adopt Pass@1 as our primary evaluation metric, computed as the expected accuracy over 32 independently sampled responses per query (i.e., Pass@1 $= \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}[o_i \text{ is correct}]$ where $N = 32$). The maximum response length is set to 3,072 tokens. Sampling uses temperature 0.6 and top-p 0.95 to generate 32 responses per query. The training template appears in Table 6 (Appendix C).

[3] https://huggingface.co/datasets/EleutherAI/hendrycks_math

[4] https://github.com/huggingface/Math-Verify

[5] https://huggingface.co/datasets/AI-MO/aimo-validation-aime

[6] https://huggingface.co/datasets/AI-MO/aimo-validation-amc

Algorithm 1 Group Length Policy Optimization

Require: Policy π_θ, Reference π_{ref}, Reward $r(o, a)$, Dataset $\mathcal{D}$, Group size G, Max length $\mathcal{L}_{\text{Max}}$, Penalty λ

```
while not converged do
    Sample batch of questions q ~ D
    for each question q do
        Sample G responses {o_1, ..., o_G} from π_θ_old(·|q)
        Compute outcome rewards r_i = r(o_i, a) for i = 1, ..., G
        if Σ_{j=1}^{G} r_j = G then
            L̂_i ← (L_Max − L_i)/L_Max, r_{l_i} ← r_i + λ L̂_i ∀i
        else
            r_{l_i} ← r_i ∀i
        end if
        Compute advantage A_{l_i} via group normalization of r_{l_i}
    end for
    Update π_θ by maximizing J_GLPO(θ)
end while
```

3.5 Baselines

Baselines include Qwen2.5-1.5B and 7B (Yang et al., 2024a), Qwen2.5-Math-1.5B and 7B (Yang et al., 2024b), SimpleRL-Zero (Zeng et al., 2025), Open-Reasoner-Zero-7B (Hu et al., 2025), Eurus-2-7B-PRIME (Cui et al., 2025), and Oat-Zero-1.5B and 7B (Liu et al., 2025).

3.6 Main Results

The empirical results in Tables 1 and 2 suggest that GLPO is competitive with, and on average slightly above, tuned baselines across five competition-level benchmarks. The absolute gains over a strongly tuned GRPO baseline are modest (+1.1 at 1.5B and +0.5 at 7B on average) and not uniform across benchmarks. Gains concentrate on the harder ones (AIME, AMC, OlympiadBench), while MATH-500 and Minerva are flat or slightly below the GRPO baseline at 1.5B. We report all five benchmarks rather than a curated subset.

Performance on 1.5B Scale. As shown in Table 1, GLPO-Zero-1.5B reaches 43.7% average accuracy, slightly outperforming Oat-Zero-1.5B (41.1%) and the tuned GRPO baseline (42.6%). Gains are most visible on the harder benchmarks such as AIME 2024 (23.3% vs. 16.0 for Oat-Zero) and AMC 2023 (62.8% vs. 57.8 for GRPO). Over the Qwen2.5-Math-1.5B-Base model, GLPO yields a relative average improvement of about 51%. These

Model	AIME 2024	MATH-500	AMC 2023	Minerva Math	Olympiad Bench	Avg. Score
Qwen2.5-1.5B-Base (Yang et al., 2024a)†	0.0	3.3	2.5	1.8	1.5	1.82
Qwen2.5-Math-1.5B-Base (Yang et al., 2024b)†	11.3	51.7	44.0	11.3	26.0	28.9
Qwen2.5-1.5B-Instruct (Yang et al., 2024a)†	1.3	57.5	26.2	19.4	20.3	24.9
Qwen2.5-Math-1.5B-Instruct (Yang et al., 2024b)†	12.0	74.7	26.7	35.0	37.9	37.3
Oat-Zero-1.5B (Liu et al., 2025)†	16.0	73.5	52.5	26.3	37.2	41.1
Qwen2.5-Math-1.5B-GRPO ($\varepsilon_h = 0.28, \alpha = 10^{-3}$)	20.0	75.4	57.8	23.2	36.5	42.6
GLPO-Zero-1.5B	23.3	73.8	62.8	20.8	37.7	43.7

Table 1: Overall performance on five competition-level reasoning benchmarks at the 1.5B scale under the zero RL paradigm. † indicates results taken from (Hochlehnert et al., 2025).

Model	AIME 2024	MATH-500	AMC 2023	Minerva Math	Olympiad Bench	Avg. Score
Qwen2.5-7B-Base (Yang et al., 2024a)†	3.3	64.6	30.0	25.7	29.0	30.5
Qwen2.5-Math-7B-Base (Yang et al., 2024b)†	20.7	64.3	56.2	17.3	29.0	37.5
Qwen2.5-7B-Instruct (Yang et al., 2024a)†	12.3	77.1	52.8	34.9	38.7	43.2
Qwen2.5-Math-7B-Instruct (Yang et al., 2024b)†	15.7	82.9	67.0	35.0	41.3	48.4
Eurus-2-7B-PRIME (Cui et al., 2025)†	17.8	80.1	63.0	37.5	43.9	48.5
Open-Reasoner-Zero-7B (Hu et al., 2025)†	19.7	83.9	59.5	31.6	47.6	48.5
SimpleRL-Zero-7B (Zeng et al., 2025)†	14.0	77.9	58.0	33.0	39.0	44.4
SimpleRL-Zero-Math-7B (Zeng et al., 2025)†	22.7	76.9	62.2	30.1	39.3	46.2
Oat-Zero-7B (Liu et al., 2025)†	28.0	79.4	66.2	34.4	43.8	50.4
Qwen2.5-Math-7B-GRPO ($\varepsilon_h = 0.28, \alpha = 10^{-3}$)	48.3	83.9	76.0	28.5	45.0	56.3
GLPO-Zero-7B	48.2	84.1	76.3	29.7	45.8	56.8

Table 2: Overall performance on five competition-level reasoning benchmarks at the 7B scale under the zero RL paradigm. † indicates results taken from (Hochlehnert et al., 2025).

gains come together with shorter responses (Table 3), suggesting that shorter traces do not necessarily degrade reasoning performance. On MATH-500 and Minerva, however, accuracy is slightly below the GRPO baseline (73.8 vs. 75.4 and 20.8 vs. 23.2), suggesting that on easier problems the mastery-gated length pressure may occasionally remove useful intermediate steps. We discuss this trade-off in the ablation analysis.

Performance on 7B Scale. At the 7B scale (Table 2), GLPO-Zero-7B reaches an average accuracy of 56.8%, slightly above Oat-Zero-7B (50.4%) and the tuned GRPO baseline (56.3%). It is strongest on AIME 2024 (48.2%) and AMC 2023 (76.3%). Relative to the base model, GLPO yields about a 51% relative improvement in average accuracy. The headline absolute gain over a strongly tuned GRPO baseline ($+0.5$ on average) is modest. We view the more meaningful contribution at this scale as the length efficiency improvement reported below.

Cross-Scale Consistency. GLPO's improvements are largely consistent across both scales. Accuracy gains concentrate on the harder benchmarks (AIME, AMC) while response length is compressed. The 1.5B model benefits more in average accuracy (+1.1%), whereas the 7B model gains primarily in length efficiency (+0.5% accuracy), consistent with larger models exhibiting greater reasoning redundancy. By deferring length optimization until consistent correctness is achieved, GLPO largely avoids the premature compression failure mode of uniform length penalties.

Per-Benchmark Response Length. Per-benchmark length reductions vary. For the 1.5B model, we observe roughly 11–22% reduction on OlympiadBench, MATH-500, AMC 2023, and Minerva, and essentially no change on AIME 2024. For the 7B model, we observe 4–6% reduction across all five benchmarks. These reductions are not always paired with accuracy gains (e.g., the 1.5B accuracy is slightly lower on MATH-500 and Minerva than the GRPO baseline, as shown in Tables 1–2), with the largest length compression on benchmarks where the model is more often unanimously correct, where the length penalty is active most often.

3.7 Ablation Study

We ablate two key hyperparameters: the length penalty coefficient λ and the rollout group size n. Figure 2 shows the training dynamics of reward and response length for both 1.5B and 7B models.

For the length penalty ablation (top row), we see a clear tradeoff between reward maximization

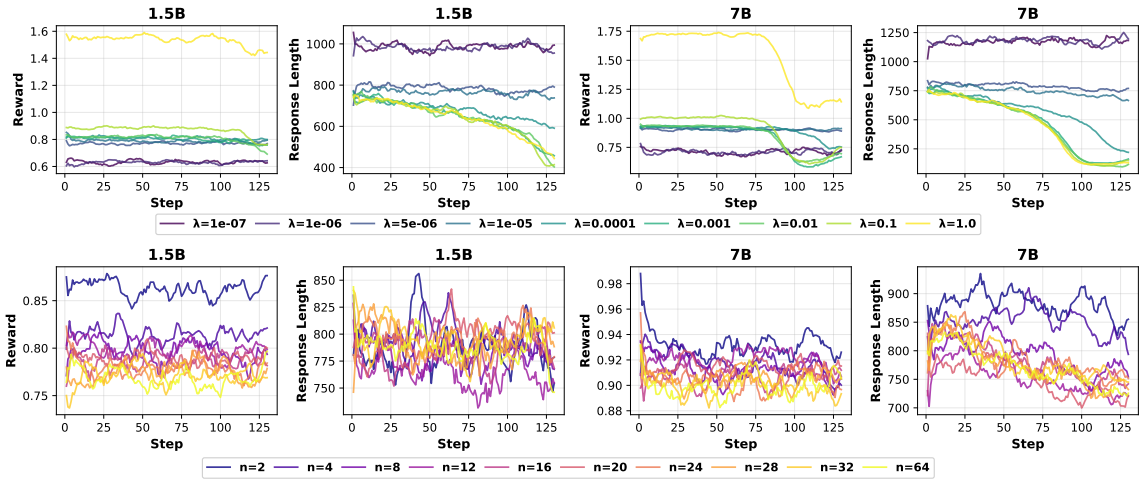

Figure 2: Evolution of accuracy and average response length across training steps.

Model	Config	OlyBench	MATH	AIME	AMC	Minerva	Avg Score	Avg Len
1.5B	Step 0 (Baseline)	0.365	0.754	0.200	0.578	0.232	0.426	1001
	λ=1e-4, Step=100	0.377	0.738	0.233	0.628	0.208	0.437	891
	Δ	+0.012	-0.016	+0.033	+0.050	-0.024	+0.011	-110
	λ=1, Step=120	0.319	0.712	0.267	0.506	0.195	0.400	676
	Δ	-0.046	-0.042	+0.067	-0.072	-0.037	-0.026	-325
7B	Step 0 (Baseline)	0.450	0.839	0.495	0.760	0.285	0.566	951
	λ=1e-4, Step=20	0.458	0.841	0.482	0.763	0.297	0.568	903
	Δ	+0.008	+0.002	-0.013	+0.003	+0.012	+0.002	-48
	λ=1e-4, Step=90	0.432	0.823	0.422	0.699	0.286	0.532	623
	Δ	-0.018	-0.016	-0.073	-0.061	+0.001	-0.034	-328

Table 3: GLPO Length Penalty Optimization Results

and response length reduction. Larger λ (e.g., $\lambda = 1$) induces aggressive length reduction at the cost of lower rewards, while smaller values (e.g., $\lambda = 10^{-6}$) preserve rewards but yield minimal compression. A favorable tradeoff is observed at $\lambda = 10^{-4}$, where both models exhibit a clear length reduction while preserving, or slightly improving, task performance. Specifically, the 1.5B model reduces average response length from 1001 to 891 tokens (11% reduction) at step 100 while improving accuracy from 0.426 to 0.437. The 7B model achieves a 5% length reduction (951 $\rightarrow$ 903 tokens) at step 20 with a slight accuracy improvement (0.566 $\rightarrow$ 0.568). The training curves show that response length decreases monotonically across all λ values, while reward trajectories diverge depending on the penalty strength.

The rollout ablation (bottom row) shows that larger n values yield more stable training and higher rewards. However, its effect on response length is weaker than that of the length penalty. The 7B model with $n = 28$ offers a favorable balance, reaching competitive accuracy while maintaining reasonable response lengths. We also observe that the 1.5B model shows less sensitivity to the rollout count, suggesting that smaller models may require fewer samples for stable policy gradient estimation.

These results suggest that GLPO can compress model outputs without major performance degradation in the moderate-penalty regime. Under stronger penalty ($\lambda = 1$ at step 120 for 1.5B), length reduction reaches $\sim$32.5% but at a real cost of -6.1 accuracy points (42.6%$\rightarrow$40.0%). We take $\lambda = 10^{-4}$ as the operating point of GLPO and treat the heavier-penalty result as an upper bound on compression, not as our headline accuracy claim. A meaningful fraction of output verbosity in current reasoning models thus appears redundant under conditional length pressure, while very aggressive compression eventually trades accuracy for brevity.

3.8 Analysis of Length Penalty Optimization

Table 3 ablates the length penalty coefficient λ and training steps, revealing key tradeoffs between

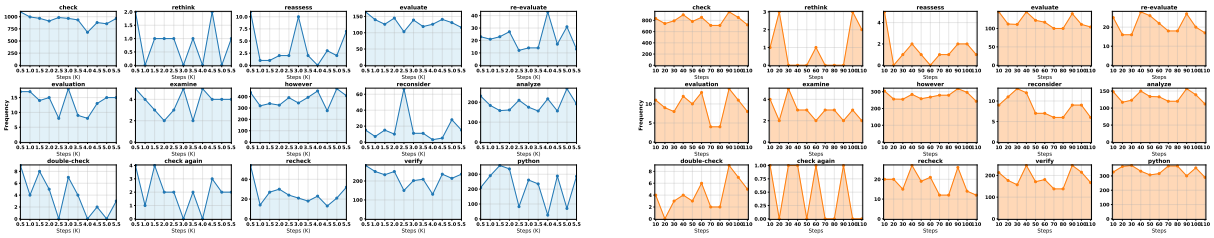

Figure 3: CoT keyword frequency analysis. **Left:** Stage 1 accuracy training (14,022 responses from 1,558 questions $\times$ 11 checkpoints). **Right:** Stage 2 length optimization (15,580 responses from 1,558 questions $\times$ 10 checkpoints).

Table 4: Qualitative examples of GLPO reasoning behavior. The model occasionally produces Python code for answer verification. These are illustrative rather than a quantitative claim.

Question (a)	**Question (b)**
An integer, whose decimal representation reads the same left to right and right to left, is called symmetrical. For example, the number 513151315 is symmetrical, while 513152315 is not. How many nine-digit symmetrical numbers exist such that adding the number 11000 to them leaves them symmetrical?	For positive integers a, b, $a \uparrow\uparrow b$ is defined as follows: $a \uparrow\uparrow 1 = a$, and $a \uparrow\uparrow b = a^{a\uparrow\uparrow(b-1)}$ if $b > 1$. Find the smallest positive integer n for which there exists a positive integer a such that $a \uparrow\uparrow 6 \not\equiv a \uparrow\uparrow 7 \pmod{n}$.
Response	**Response**

`<think>` ...
Let's write a Python code to iterate through all possible nine-digit symmetrical numbers and check if adding 11000 results in another symmetrical number.
 	
`def check_palindrome(n):`
`  return str(n) == str(n)[::-1]`
`count = 0`
`for d1 in range(1, 10):`
`  for d2 in range(0, 10):`
`  for d3 in range(0, 10):`
`  for d4 in range(0, 10):`
`  for d5 in range(0, 10):`
`  num = int(f"{d1}{d2}{d3}{d4}"`
`  f"{d5}{d4}{d3}{d2}{d1}")`
`  if check_palindrome(num+11000):`
`  count += 1`
`print(count)`
`count = 40`
So, there are 40 nine-digit symmetrical numbers that, when 11000 is added to them, remain symmetrical.
`</think>`
`<answer>` $\boxed{40}$ `</answer>`
...
	`<think>` ...
	### Step-by-Step Check ...
	
	#### Continue this process programmatically for efficiency:
	`def tetration_mod(a, b, n):`
	`  if b == 1:`
	`  return a % n`
	`  exp = tetration_mod(a, b-1, n)`
	`  return pow(a, exp, n)`
	`def find_smallest_n():`
	`  for n in range(2, 100):`
	`  for a in range(1, n):`
	`  if tetration_mod(a, 6, n)`
	`  != tetration_mod(a, 7, n):`
	`  return n`
	`  return None`
	
	...
	`3*4*5*9*2 = 60`
	
	`</think>`
	`<answer>` $\boxed{60}$ `</answer>`
	...

thinking budget compression and reasoning quality.

At the 1.5B scale, a moderate penalty ($\lambda = 10^{-4}$, 100 steps) yields a "win-win" outcome of $+1.1\%$ average accuracy (42.6%→43.7%) with about 110 fewer tokens per response. Similarly, the 7B model (at step 20) retains its accuracy (56.8%) while reducing the response length by about 48 tokens. This is consistent with baseline models producing some amount of redundant reasoning. By rewarding the model for using less of its remaining context window, GLPO offers a simple mechanism for *progressive thinking budget control*, allocating fewer reasoning steps to problems the model has mastered.

These experiments also outline the boundary of length optimization. When the penalty is too aggressive (e.g., $\lambda = 1$ for 1.5B) or the optimization is prolonged (e.g., Step 90 for 7B), the model achieves large reductions in length (approximately -325 tokens for 1.5B and -328 tokens for 7B). However, this comes at a clear cost in accuracy, with average scores dropping by 2.6% and 3.4%, respectively. This reflects a natural boundary of thinking budget optimization. While removing redundancy is beneficial, overly aggressive compression forces the model to omit needed reasoning steps, hurting accuracy. The group-level condition in GLPO attenuates this failure mode by withholding length pressure on problems the model has not yet mastered.

Both the 1.5B and 7B models exhibit consistent behavior. A mild length incentive improves or maintains accuracy by pruning unnecessary tokens, whereas an excessive incentive degrades reasoning quality. The 1.5B model benefits more in accuracy, while the 7B model gains mainly in efficiency.

Mastery-Gated, Implicit Difficulty Signal. A useful property of GLPO is that its group-level correctness condition acts as an *implicit*, mastery-based proxy for difficulty, without requiring explicit difficulty labels (Wen et al., 2025). Easy problems (all G rollouts correct early on) receive length pressure

first and undergo compression. Harder problems (at least one incorrect rollout) remain free from length constraints, allowing continued exploration. As the model masters increasingly difficult problems during training, the length optimization frontier shifts accordingly. We caution that unanimous group correctness is a proxy rather than a direct difficulty measure, and softer thresholds or alignment with external difficulty signals (e.g., problem level or success rate) are natural extensions.

Overall, GLPO offers a simple way to control response length via mastery gating, sidestepping part of the accuracy and efficiency tension associated with uniform length penalties.

3.9 Analysis of Reasoning Patterns

Inspired by prior work observing reflective behavior through keyword pools, we use a predefined keyword pool to track changes in thinking patterns during training. Our keyword pool includes *check, rethink, reassess, evaluate, re-evaluate, evaluation, examine, however, reconsider, analyze, double-check, check again, recheck, verify,* and *wait*.

We present the occurrences of these keywords across different training stages in Figure 3 and Figure 3. Comparing the first and second stages, the frequency of code-based verification (the "python" keyword) visibly increases, while keywords such as "recheck" and "re-evaluate" decrease in relative frequency. Code-based verification appears more token-efficient than verbose textual rechecking. We emphasize that keyword frequency is a coarse, surface-level proxy for reasoning strategy and does not distinguish a genuine shift in approach from lexical rewording of the same steps. A controlled study (varying code availability, or using semantic clustering of reasoning traces) would be needed for a causal claim, and we frame this analysis as descriptive rather than confirmatory.

Interpretation. These patterns suggest two behavioral shifts under GLPO. In Stage 1, the model develops self-verification strategies. In Stage 2, it appears to prune redundant verification while favoring more efficient mechanisms, such as code-based checks. This is consistent with the view that GLPO does not merely truncate text but can shift the underlying reasoning *strategies*, though a formal causal claim would require additional controlled analysis.

3.10 Case Study

Figure 4 illustrates a behavior we observe in trained models. The model occasionally produces Python code as part of its verification step. In Question (a), it directly computes the answer via code. In Question (b), it first reasons mathematically and then writes code to cross-check its solution. These are anecdotal examples rather than a quantitative claim, but they are consistent with the keyword-frequency shifts reported above and with the view that GLPO-trained models gradually adopt more compact verification routines. A more controlled study (e.g., isolating code use from textual conciseness) would help quantify how much of the length gain is attributable to strategy change versus token-level pruning.

4 Related Work

RL for Reasoning. Recent RL advances have noticeably improved LLM reasoning. OpenAI o1 (Jaech et al., 2024) pioneered large scale RL for CoT reasoning, achieving strong gains on mathematics and coding benchmarks. DeepSeek-R1 (Guo et al., 2025) showed that pure RL post training via GRPO induces reasoning competitive with o1, with emergent self verification and planning. Follow-up work (Hu et al., 2025; Zeng et al., 2025; Face, 2025) reduced memory and compute overhead while maintaining competitive performance.

Efficient Reasoning. A growing line of work addresses overthinking in reasoning models. L1 (Wen et al., 2025) applies a uniform length penalty during RL training, SOBER (Hochlehnert et al., 2025) uses a length-conditioned reward based on output statistics, and ThinkPrune (Hou et al., 2025) and Concise-Reasoning (Fatemi et al., 2025) apply length budgets or budget-aware shaping uniformly across samples regardless of correctness. DeepScaleR (Luo et al., 2025) expands rather than compresses the reasoning window. In contrast, GLPO conditions the length signal on *group-level correctness*, so that compression is applied only where the model has reliably solved the question, as a drop-in modification to length-penalized RL.

5 Conclusion

We introduce GLPO, a reinforcement learning method that conditions length optimization on group-level correctness. On five math benchmarks at 1.5B and 7B, GLPO compresses response length by 5–11% (up to ∼33% with a heavier penalty, at an accuracy cost) while matching or slightly improving accuracy over a tuned GRPO baseline.

6 Limitations

Our experiments are limited to the 1.5B and 7B scales on the Qwen2.5-Math family and to competition-level mathematics. We have not evaluated GLPO on larger backbones, other model families (e.g., LLaMA, DeepSeek), or non-math reasoning tasks (e.g., code generation, agentic planning, scientific QA), and the choice of a single math-specialized base model family means that conclusions about how the mastery gate interacts with stronger general-purpose pretraining remain open. The absolute average accuracy gain over a strongly tuned GRPO baseline is modest, particularly at 7B ($+0.5$), so we view the length efficiency improvement as the more substantive contribution rather than a new accuracy state of the art.

We use a hard unanimous-correctness condition ($\sum_j r_j = G$) and have not systematically compared softer thresholds (e.g., $\geq 75\%$ or $\geq 50\%$ of rollouts correct), which may be more suitable for weaker base models or harder problem distributions where unanimous correctness is rare. Our pipeline is two-stage by construction (Stage 1 accuracy training, Stage 2 length optimization). We have not run the controlled ablations that would isolate the contribution of the gating mechanism from that of staged training: specifically, (a) a single-stage GLPO with the gate active from step 0, and (b) a Stage-2 with a uniform length penalty (no gate) applied to the same Stage-1 checkpoint. We outline both as the most direct next experiments and do not claim that the gating rule alone, independent of staging, is what drives the reported gains.

Finally, our evidence for reasoning-strategy shifts (e.g., toward code-based verification) is based on keyword frequency and qualitative cases. A finer quantitative analysis (semantic clustering of reasoning traces, controlled code-availability ablation) would be needed to support stronger causal claims. We also do not provide error bars or significance tests across random seeds, and the gain magnitudes at 7B in particular are small enough that seed variance is a plausible alternative explanation that we cannot rule out from the current experiments.

7 Ethics Statement

Our work focuses on improving the efficiency of LLM reasoning through reinforcement learning optimization. The training data used in this study consists entirely of publicly available mathematics competition problems and datasets, which do not involve sensitive personal information or raise privacy concerns. GLPO reduces computational overhead through more concise reasoning. We acknowledge that language model optimization carries potential for misuse and encourage responsible deployment of GLPO trained models.

8 Reproducibility Statement

To support reproducibility, we provide detailed descriptions of our training setup, including hyperparameters (Section 3), dataset curation procedures, and evaluation protocols.

- All training datasets (DeepScaleR, DAPO-Math-17K, MATH) are publicly available with URLs provided in the paper.
- We use the open source verl framework for training and Math-Verify for answer extraction.
- All evaluation benchmarks are publicly available with URLs provided.
- Key hyperparameters are explicitly stated, including learning rate 1×10^{-6}, batch size 128, rollout number 32, clipping parameters $\varepsilon_l = 0.2$ and $\varepsilon_h = 0.28$, and length penalty coefficient $\lambda = 10^{-4}$.
- We will publicly release model checkpoints to enable direct comparison.

References

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V. Le, Sergey Levine, and Yi Ma. 2025. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *Preprint*, arXiv:2501.17161.

Alejandro Cuadron, Dacheng Li, Wenjie Ma, Xingyao Wang, Yichuan Wang, Siyuan Zhuang, Shu Liu, Luis Gaspar Schroeder, Tian Xia, Huanzhi Mao, Nicholas Thumiger, Aditya Desai, Ion Stoica, Ana Klimovic, Graham Neubig, and Joseph E. Gonzalez. 2025. The danger of overthinking: Examining the reasoning-action dilemma in agentic tasks. *Preprint*, arXiv:2502.08235.

Ganqu Cui, Lifan Yuan, Zefan Wang, Hanbin Wang, Wendi Li, Bingxiang He, Yuchen Fan, Tianyu Yu, Qixin Xu, Weize Chen, and 1 others. 2025. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*.

Hugging Face. 2025. Open r1: A fully open reproduction of deepseek-r1.

Mehdi Fatemi, Banafsheh Rafiee, Mingjie Tang, and Kartik Talamadupula. 2025. Concise reasoning via reinforcement learning. *arXiv preprint arXiv:2504.05185*.

Bofei Gao, Feifan Song, Zhe Yang, Zefan Cai, Yibo Miao, Qingxiu Dong, Lei Li, Chenghao Ma, Liang Chen, Runxin Xu, Zhengyang Tang, Benyou Wang, Daoguang Zan, Shanghaoran Quan, Ge Zhang, Lei Sha, Yichang Zhang, Xuancheng Ren, Tianyu Liu, and Baobao Chang. 2024. Omni-math: A universal olympiad level mathematic benchmark for large language models. *CoRR*, abs/2410.07985.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, and 1 others. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3828–3850.

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

Andreas Hochlehnert, Hardik Bhatnagar, Vishaal Udandarao, Samuel Albanie, Ameya Prabhu, and Matthias Bethge. 2025. A sober look at progress in language model reasoning: Pitfalls and paths to reproducibility. *Preprint*, arXiv:2504.07086.

Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. 2025. Thinkprune: Pruning long chain-of-thought of llms via reinforcement learning. *arXiv preprint arXiv:2504.01296*.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. 2025. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *Preprint*, arXiv:2503.24290.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, and 1 others. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, and 1 others. 2022. Solving quantitative reasoning problems with language models. *Advances in Neural Information Processing Systems*, 35:3843–3857.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. 2025. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. https://github.com/agentica-project/deepscaler. Notion Blog.

Yingqian Min, Zhipeng Chen, Jinhao Jiang, Jie Chen, Jia Deng, Yiwen Hu, Yiru Tang, Jiapeng Wang, Xiaoxue Cheng, Huatong Song, Wayne Xin Zhao, Zheng Liu, Zhongyuan Wang, and Ji-Rong Wen. 2024. Imitate, explore, and self-improve: A reproduction report on slow-thinking reasoning systems. *Preprint*, arXiv:2412.09413.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*.

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

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, page 1279–1297. ACM.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *Preprint*, arXiv:2408.03314.

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

Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Hanjie Chen, Xia Hu, and 1 others. 2025. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Lifu Tang, Xiaowei Lv, and 1 others. 2025. Light-r1: Curriculum

Hyperparameter	Stage 1	Stage 2
Base Model	Qwen2.5-Math-1.5B/7B	Stage 1 Checkpoint
Optimizer	Adam	Adam
Learning Rate	1×10^{-6}	1×10^{-6}
Batch Size	128	128
Rollouts per Question (G)	32	32
Max Response Length ($\mathcal{L}_{\text{Max}}$)	3,072	3,072
Clip Lower (ε_l)	0.2	0.2
Clip Upper (ε_h)	0.28	0.28
KL Coefficient (β)	10^{-3}	10^{-3}
Entropy Bonus (α)	10^{-3}	10^{-3}
Length Penalty (λ)	0	10^{-4}
Training Data	59K problems	MATH (Levels 3–5)

Table 5: Hyperparameter settings for the two-stage GLPO training pipeline.

sft, dpo and rl for long cot from scratch and beyond. *arXiv preprint arXiv:2503.10460*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024a. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.

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. 2024b. Qwen2.5-math technical report: Toward mathematical expert model via self-improvement. *Preprint*, arXiv:2409.12122.

Wang Yang, Hongye Jin, Jingfeng Yang, Vipin Chaudhary, and Xiaotian Han. 2025. Thinking preference optimization. *arXiv preprint arXiv:2502.13173*.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. 2025. Limo: Less is more for reasoning. *arXiv preprint arXiv:2502.03387*.

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, and 16 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *Preprint*, arXiv:2503.14476.

Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. 2025. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *Preprint*, arXiv:2503.18892.

A Hyperparameter Settings

Table 5 summarizes the full set of hyperparameters used in our experiments.

B Two-Stage Training Pipeline

GLPO employs a two-stage training pipeline as described below.

Table 6: Training Template. **{question}** will be replaced with the question during the training and testing.

A conversation between User and Assistant. The user asks a question, and the Assistant solves it.
The assistant first thinks about the reasoning process in the mind and then provides the user with the answer.
The reasoning process and answer are enclosed within `<think></think>` tags and subsequently presenting the answer within the `<answer></answer>` tags, respectively,
i.e., `<think>` reasoning process here `</think>` `<answer>` answer here `</answer>`.
Please reason step by step, and put your final answer within `\boxed{}`.

User:
{question}

Assistant:

Stage 1: Reasoning Ability Enhancement. In this stage, we train the base model using the full 59K dataset with the standard GRPO objective augmented with the asymmetric clipping and entropy bonus. The length penalty is disabled ($\lambda = 0$) to allow the model to freely explore reasoning strategies and develop stable problem-solving capabilities.
Stage 2: Length Optimization. After the model has acquired strong reasoning abilities, we enable the length penalty ($\lambda = 10^{-4}$) and continue training on the MATH (Levels 3–5) subset. This stage follows the "*walk before you run*" principle. Since the model has already mastered mathematical reasoning, it can now learn to express solutions more concisely without sacrificing correctness. The $\sum_{j=1}^{G} r_j = G$ condition ensures that length optimization only occurs when the model consistently produces correct answers.

C Training Template

Table 6 presents the training template used in both training and evaluation.