Step-GRPO: Enhancing Reasoning Quality and Efficiency via Structured PRM-Based Reinforcement Learning

Weijie Li[1], **Jin Wang**[1*], **Liang-Chih Yu**[2*], **Xuejie Zhang**[1]

[1]School of Information Science and Engineering, Yunnan University, Yunnan, P.R. China
[2]Department of Information Management, Yuan Ze University, Taiwan
liweijie1@stu.ynu.edu.cn, wangjin@ynu.edu.cn, xjzhang@ynu.edu.cn, lcyu@saturn.yzu.edu.tw

Abstract

Large reasoning models (LRMs) improve performance at test time by *thinking longer*, but this often leads to overthinking and high computational cost. To address this, recent reinforcement learning (RL) methods adopt mechanical outcome-level rewards (e.g., rule- or prompt-based) that favor shorter correct paths, but frequently overlook reasoning quality. While such rewards neglect intermediate reasoning, dense supervision from process reward models (PRMs) has proven more effective in promoting coherent and high-quality reasoning. However, static PRM supervision introduces two challenges: *reward hacking*, as fixed rewards poorly capture global reasoning objectives, and *the high training cost* of obtaining dense reward labels at scale. To overcome these issues, we propose step group relative policy optimization (**Step-GRPO**), a GRPO-based method that integrates step-level PRM signals into sparse trajectory-level feedback, avoiding costly step-level supervision while enhancing reasoning quality beyond accuracy. In addition, Step-GRPO employs a step-attention mechanism that captures inter-step dependencies and emphasizes critical reasoning steps, effectively mitigating reward hacking. We apply Step-GRPO to train LLMs, achieving consistent gains in reasoning quality, accuracy, and shorter reasoning traces across multiple math benchmarks, outperforming RL baselines at substantially lower cost. Notably, the proposed model achieves 36.7% accuracy on AIME'24 with 11K samples and $38 training cost, surpassing $1000+ baselines trained on 40K+ samples, demonstrating strong cost-effectiveness and scalability.

Introduction

Large reasoning models (LRMs) (Guo et al. 2025; Jaech et al. 2024; Yuan et al. 2023, 2025) improve mathematical reasoning by thinking more meticulously and longer to produce more elaborate chain-of-thought (CoT) trajectories (Wei et al. 2023), but frequently generate low-quality intermediate steps (Figure 1, top left) and incur substantial computational cost (Sui et al. 2025) (a tendency known as *overthinking*). To mitigate *overthinking*, recent reinforcement learning (RL)-based methods (Dang and Ngo 2025; Xie et al. 2025) assign higher rewards to shorter correct reasoning paths during post-training. However, while these methods effectively regulate length, they often neglect process quality, as evidenced by low scores from process reward models (PRMs) (Figure 1, right), which correlate with limited accuracy gains. To more accurately evaluate reasoning quality, PRMs for PRM scoring have been introduced as token-level scoring models that offer dense supervision beyond final-answer correctness (Zhang et al. 2025b; He et al. 2025). As dense reward signals, PRMs provide fine-grained feedback that facilitates more effective credit assignment and enables RL to better align with step-level reasoning quality than sparse outcome-level rewards (Uesato et al. 2022; Yuan et al. 2024).

However, the practical use of PRMs in RL remains limited compared to GRPO-style methods in commercial applications (Team et al. 2025; Uesato et al. 2022), primarily due to two key challenges: reward hacking (Gao, Schulman, and Hilton 2022) and the high training cost of step-level dense supervision (Cui et al. 2025). The first challenge stems from optimizing against a static reward model. Static PRMs supervise at the reasoning step level, but lack adaptability to evolving model behaviors, often leading to reward hacking (Gao, Schulman, and Hilton 2022), where the model overfits local signals while ignoring global reasoning quality. It is necessary to track distribution drift and maintain alignment to update the PRM during training (Leike et al. 2018). However, the high cost of obtaining dense reward labels at scale is challenging. PRM-based methods still rely on expensive human-labeled data for calibration, and require over 20× more rollouts than final-answer supervision (Wang et al. 2024b; Kazemnejad et al. 2025), making scalable online RL impractical. Inspired by the effectiveness of dense reward signals, recent approaches have sought to aggregate these signals into sparse outcome-level rewards (e.g., minimum, maximum, mean), thereby reducing the computational burden of step-by-step rollouts, while still providing effective supervision that improves reasoning quality and efficiency with lower resource consumption (Liu et al. 2025a; Wang et al. 2024a; Zuo et al. 2025). However, such aggregation-based approaches fail to address reward hacking, as they reduce rich feedback to a single score, sacrificing step-level granularity and trajectory coherence. For example, minimum aggregation penalizes the weakest step, but as shown in Figure 1 (bottom left), it may overemphasize detours, diverting the model from coherent reasoning.

*Corresponding author.

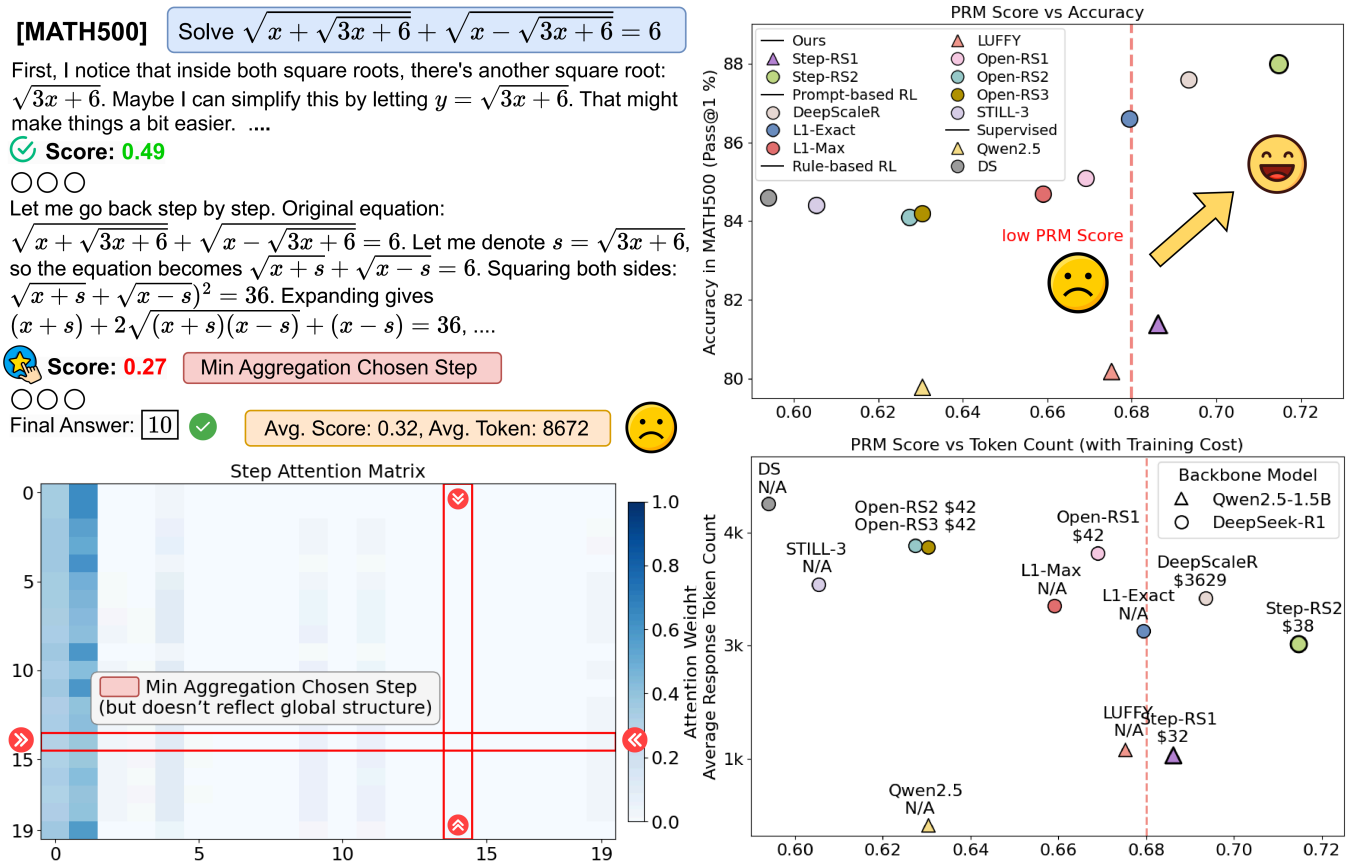


Figure 1: Visualization of PRM challenges and Step-GRPO improvements. Top Left: A training sample in MATH500 where Min aggregation selects a low-quality step. Bottom Left: Step attention weights over reasoning steps. Top Right: Accuracy vs. PRM score on MATH500 shows Step-GRPO achieves higher reasoning quality and accuracy than prompt- and rule-based baselines. Bottom Right: PRM score vs. token count on MATH500 shows Step-GRPO achieves the best quality-efficiency tradeoff among all methods while maintaining low training cost. Step-GRPO improves reasoning quality, accuracy, and length efficiency while addressing reward hacking and cost issues in PRM-based training.

This study proposes step group relative policy Optimization (**Step-GRPO**), a simple RL method that trains LRMs with step-level dense process rewards to enhance reasoning quality and reduce overthinking. Step-GRPO builds on GRPO (Shao et al. 2024), a sample-efficient framework that avoids value-based critics and costly online rollouts by training with trajectory-level rewards. To bridge dense process-level rewards with GRPO's sparse supervision, we introduce a **step-attention** mechanism, inspired by the attention paradigm (Vaswani et al. 2023), that aggregates step-wise scores into a trajectory-level reward. To avoid reward hacking, step-attention applies a soft positional bias to capture inter-step dependencies and computes attention weights to emphasize critical reasoning steps. This mechanism enables fine-grained yet efficient supervision, effectively mitigating reward hacking during training.

To assess the robustness of Step-GRPO, we finetune a weaker model (Qwen2.5-Math-1.5B-Instruct, Step-RS1) and a stronger model (Qwen-Distilled-R1-1.5B, Step-RS2) (Yang et al. 2024; Guo et al. 2025). Both models achieve substantial improvements in step-wise reasoning quality (0.72+), leading to higher final-answer accuracy (88%+) and more concise reasoning traces (2K+) across MATH500 (Hendrycks et al. 2021) and other benchmarks, outperforming rule-based, prompt-based, and standard PRM-RL baselines (Figure 1, right). Notably, Step-RS2 achieves 36.7% accuracy on AIME'24 at a total training cost of just $38[1] (using LoRA (Hu et al. 2021) on 11K samples), surpassing RL-based baselines ($1,000 and 40K+ samples)—highlighting the cost-effectiveness and scalability of Step-GRPO. Further analysis confirms PRM-based step supervision improves reasoning quality over rule-based heuristics and that dynamic aggregation mitigates reward hacking.

Related Work

Scaling test-time computation has proven effective for improving LLM performance in complex reasoning tasks (OpenAI 2024; Wu et al. 2025; Shen et al. 2025b,a). Previous works explore a variety of external test-time scal-

[1]Training cost is estimated using the pricing schedule provided by OpenRS (Dang and Ngo 2025).

ing methods, including majority voting (Wang et al. 2023), search-based approaches (Yao et al. 2023; Xie et al. 2023), and iterative refinement (Qu et al. 2024). These methods consistently demonstrate that increasing the number or length of reasoning chains leads to predictable performance gains. For verification-guided test-time compute, prior works (Kang et al. 2024; Wu et al. 2025; Snell et al. 2024; Wang et al. 2024a) propose search-based methods leveraging PRMs (He et al. 2025; Zheng et al. 2025) to enable more focused generation and more reliable answer selection, effectively scaling test-time compute. Recent reasoning language models such as "O1" and "R1"-style models (OpenAI 2024; Guo et al. 2025) simplify test-time scaling by encouraging the model to "*think longer*" through extended chains-of-thought. While effective, these models often overthink, generating unnecessarily long chains that increase test-time compute and latency.

To address the overthinking issue, recent work (Sui et al. 2025) has explored RL approaches, broadly categorized as rule-based, prompt-conditioned, and PRM-guided. Rule-based methods encourage shorter reasoning by rewarding concise correct outputs (Dang and Ngo 2025; Xie et al. 2025), or enforcing early stopping with special tokens (Muennighoff et al. 2025; Song et al. 2025). Prompt-based methods (Yang et al. 2025), including L1 (Aggarwal and Welleck 2025) and AdaptThinking (Zhang et al. 2025a), control length via task-aware or user-specified prompts. While effective at reducing verbosity, these methods often overlook intermediate reasoning quality, as low PRM scores reflect. Process-level RL, such as PRIME (Cui et al. 2025), uses implicit process rewards from outcome labels to improve step-level reasoning without reward hacking.

Step-GRPO

High-Quality Dataset Curation

To support cost-efficient training of reasoning-specialized small models, we curate a high-quality dataset—**Step-rs**—from the publicly available s1 dataset (Muennighoff et al. 2025), which includes 59k multi-step questions across domains such as NuminaMATH (Li et al. 2024), OlympicArena (Huang et al. 2024), OmniMath (Gao et al. 2024), logic tasks, and ScienceQA. Focusing on mathematical problem solving, we apply a three-stage refinement process to extract structurally clean and reasoning-focused samples: (1) Retain examples containing the `\boxed` command to enforce consistent answer formatting, resulting in 31k examples. (2) Filter out instances already correctly solved by a weak model (Step-RS1), leaving 30k challenging samples. (3) Remove noisy, multi-part, or low-quality examples using GPT-4o, yielding a final dataset of 12K samples. The resulting step-rs dataset provides targeted supervision for evaluating and improving step-wise reasoning quality.

Reinforcement Learning Algorithm

To mitigate reward hacking and reduce training cost in small-scale reasoning models, we propose **Step-GRPO**, a lightweight reinforcement learning framework that integrates step-level supervision into GRPO. Given an input q, GRPO draws G samples $\{o_1, o_2, \ldots, o_G\}$ from the old policy $\pi_{\theta_{\text{old}}}$, and updates the policy π_θ by optimizing an objective shaped by our step-level reward function:

$$\begin{aligned}\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{q\sim\mathbb{P}(\mathcal{Q}),\,\{o_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(O|q)} \\ \cdot \frac{1}{G}\sum_{i=1}^{G}\min\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{\text{old}}}(o_i|q)}A_i,\right. \\ \left.\text{clip}\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{\text{old}}}(o_i|q)}, 1-\epsilon,\, 1+\epsilon\right)A_i\right) \\ -\beta\,\mathbb{D}_{\text{KL}}(\pi_\theta\|\pi_{\text{ref}})\end{aligned} \tag{1}$$

where the KL-divergence term is defined as:

$$\mathbb{D}_{\text{KL}}(\pi_\theta\|\pi_{\text{ref}}) = \frac{\pi_{\text{ref}}(o_i|q)}{\pi_\theta(o_i|q)} - \log\frac{\pi_{\text{ref}}(o_i|q)}{\pi_\theta(o_i|q)} - 1, \tag{2}$$

and the advantage A_i is computed from the group of rewards $\{r_1, r_2, \ldots, r_G\}$ as:

$$A_i = \frac{r_i - \text{mean}(\{r_1, r_2, \ldots, r_G\})}{\text{std}(\{r_1, r_2, \ldots, r_G\})}. \tag{3}$$

Here, ϵ and β are hyperparameters that control the PPO clipping threshold and the KL penalty strength, respectively.

Rule-Based Reward Modeling

Reward is the core training signal in reinforcement learning. We adopt two types of rewards: a rule-based component for baseline supervision and a PRM-based component for step-level optimization. The rule-based reward includes two concise and robust signals—Format Reward and Answer Reward—refined to resist common reward hacking behaviors.

System Prompt

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, then provides the User with the answer, and puts the final answer within `\boxed{{}}`. The reasoning process and answer are enclosed within `<think> </think>` and `<answer> </answer>` tags, respectively, i.e., `<think>` reasoning process here `</think>` `<answer>` answer here `</answer>`.

Accuracy Reward. A binary score (1 for correct, 0 for incorrect) is assigned based on whether the final answer matches the ground truth, offering a simple and objective signal for accuracy supervision.

Format Reward. This reward enforces structural discipline by requiring the reasoning process to be enclosed within `<think>...</think>` and the final answer to appear inside `\boxed{}`, aligning the model with the *slow thinking* paradigm and reducing reward gaming.

Cosine-Scaled Reward (Baseline). We reproduce the cosine-scaled reward used in (Team et al. 2025; Dang and Ngo 2025) to enable fair comparisons with prior rule-based methods. This reward encourages concise outputs based on

generation length and is not part of our proposed method. We follow the hyperparameter setup of the Open-RS series (Open-RS1/2/3) to ensure compatibility. Given the normalized output length $p = \frac{L}{L_{\max}}$, where L is the generation length and $L_{\max} = 3000$, the reward is computed as:

$$\text{reward} = \begin{cases} R_c^{\min} + \frac{1+\cos(\pi p)}{2}(R_c^{\max} - R_c^{\min}), & \text{if correct} \\ R_w^{\min} + \frac{1+\cos(\pi p)}{2}(R_w^{\max} - R_w^{\min}), & \text{if incorrect} \end{cases} \tag{4}$$

We use the same hyperparameters as prior work: $[R_c^{\min}, R_c^{\max}] = [0.5, 1.0]$, $[R_w^{\min}, R_w^{\max}] = [-1.0, -0.5]$. This reward formulation compresses scores as output length increases, favoring concise responses.

PRM-Based Reward Function

To enable process-level supervision, we leverage step-level scores from PRMs $\mathcal{M}$ as dense signals for fine-grained optimization of reasoning quality beyond final answer correctness. Given a sampled reasoning trajectory $o = \{s_1, s_2, \ldots, s_n\}$, each step s_j is assigned a clipped reward:

$$p_j = \max(0, \min(\mathcal{M}(s_j), 1)),$$

producing a score vector $\mathbf{p}^{(o)} = \{p_1, p_2, \ldots, p_n\}$. Since policy optimization requires a scalar reward per trajectory, we aggregate these step-level scores into a single trajectory-level reward r_o. We introduce a **step-attention** mechanism that captures structural dependencies across steps while emphasizing semantically important or early reasoning steps (Vaswani et al. 2023). First, we enhance each score with a positional bias:

$$Q_i = p_i + \frac{1}{\sqrt{i+1}},$$

where $i \in \{0, 1, \ldots, n-1\}$ denotes the index within the reasoning chain. This formulation softly emphasizes earlier steps while preserving their raw quality score. Then, we compute the trajectory-level reward r_o by attending over the step-level scores using context-aware weights:

$$r_o = \frac{1}{n \ln(n+e)} \sum_{i=1}^{n} \left(\sum_{j=1}^{n} \frac{\exp\left(\frac{Q_i Q_j}{T}\right)}{\sum_{k=1}^{n} \exp\left(\frac{Q_i Q_k}{T}\right)} \cdot p_j \right), \tag{5}$$

where T is a temperature hyperparameter controlling the sharpness of attention. The outer normalization $\frac{1}{n \ln(n+e)}$ softly penalizes long traces. This step-attention design offers structured credit assignment and addresses the limits of prior rule-based reward methods.

Training Schedule

To evaluate the generality of Step-GRPO, two backbone models are employed: Step-RS1 is initialized from Qwen2.5-Math-1.5B-Instruct (Yang et al. 2024), while Step-RS2 builds upon DeepSeekMath-R1-Distilled-1.5B (Guo et al. 2025). These models represent distinct reasoning capabilities—moderate and strong, respectively—enabling assessment of robustness across different initialization strengths. Skywork-o1-Open-PRM-Qwen-2.5-1.5B is adopted for step scoring due to its lightweight yet reliable reward signals that facilitate efficient training. A fixed sampling ratio of 1:4:3 is used for Format, Accuracy, and PRM-based rewards, prioritizing answer correctness and reasoning quality while ensuring structural alignment. Finetuning is performed over 3,300 steps on the step-rs dataset with a learning rate of 1×10^{-5}, using a sampling temperature of 0.9 during generation and a reward aggregation temperature of 0.3 at the PRM level. The max context length is set to 4K tokens during training and 8K for evaluation. We use a batch size of 8 and generate 4 rollouts.

Experiments

Experiments Setup

Models and Dataset. We finetune two models with the Step-GRPO objective (Equation 5) for 3,300 steps on the curated step-rs dataset (11,215 examples), using different backbone capacities to assess the generality of our approach. Evaluation covers six mathematics-focused benchmarks, including competition-level datasets (AIME'24, AIME'25, AMC'23) and general mathematical reasoning datasets (MATH500 (Li et al. 2024), Minerva (Lewkowycz et al. 2022), OlympiadBench (Huang et al. 2024)), spanning a range of difficulty levels and reasoning styles. The primary metrics—zero-shot pass@1 accuracy, PRM score (reasoning quality), and token cost (reasoning length)—enable comprehensive evaluation of correctness, quality, and efficiency.

Baselines. To contextualize our results, we compare against baselines spanning four categories based on RL strategies:

- **Rule-based reinforcement methods**. This group comprises LUFFY (Yan et al. 2025), Oat-Zero (Liu et al. 2025b), Open-RS1/2/3 (Dang and Ngo 2025), STILL-3 (Min et al. 2024), DeepScaleR (Luo et al. 2025), and SimpleRL-Zoo (Zeng et al. 2025). These methods typically control reasoning length using cosine-scaled rewards (Equation 4).
- **Prompt-based reinforcement methods**. L1-Exact and L1-Max (Aggarwal and Welleck 2025) are trained via LCPO on the R1-Distill backbone (Guo et al. 2025), enabling prompt-specified control over length. Due to their prompt-conditioned reward structure, we exclude them from detailed comparison with rule-based methods.
- **PRM-level reinforcement methods**. rStar (Guan et al. 2025) and Eurus2 Prime (Cui et al. 2025) use outcome-derived rewards to optimize step-level reasoning. Due to hardware constraints, as both rely on the Qwen2.5-Math-7B-Instruct backbone, we reproduce their strategies on Qwen2.5-Math-1.5B.
- **Supervised or general-purpose models.** We include two large-scale models—o1-preview (OpenAI 2024) and Qwen2.5-Math-72B (Qwen2.5-M_72B) (Yang et al. 2024)—as upper-bound references. Smaller backbones, such as Qwen2.5-Math-7B-Instruct (Qwen Instruct), Qwen2.5-Math-1.5B-Base (Qwen Base), and DeepSeek-R1-Distill-Qwen-1.5B (R1-Distill) (Guo et al. 2025), serve to evaluate RL finetuning effects.

Model	AIME'24	AIME'25	AMC'23	MATH500	Minerva	Olympiad
General Models						
o1-preview (OpenAI 2024)	44.6	—	—	81.5	—	—
Qwen2.5-M_72B (Yang et al. 2024)	30.0	—	50.0	85.9	44.1	49.0
Based on: Qwen2.5 Math 7B						
Qwen (Instruct) (Yang et al. 2024)	13.3	5.7	50.6	79.8	34.6	40.7
rStar (Guan et al. 2025)	26.7	—	47.5	78.4	—	47.1
Eurus2 Prime (Cui et al. 2025)	26.7	13.3	57.8	79.2	38.6	42.1
SimpleRL-Zoo (Zeng et al. 2025)	26.7	13.3	62.5	82.4	39.7	43.3
Based on: Qwen2.5 Math 1.5B						
Qwen (Base) (Yang et al. 2024)	7.2	3.6	26.4	28.0	9.6	21.2
LUFFY (Yan et al. 2025)	16.7	13.3	47.1	80.2	30.5	41.0
Oat-Zero (Liu et al. 2025b)	16.7	6.7	49.6	75.2	27.1	37.2
Based on Deepseek R1 Distill Qwen 1.5B						
R1-Distill (Guo et al. 2025)	28.7	22.3	71.5	84.6	30.5	52.4
L1-Exact (Aggarwal and Welleck 2025)	24.4	22.3	70.5	86.6	31.5	52.5
L1-Max (Aggarwal and Welleck 2025)	27.7	21.0	73.2	84.7	33.3	52.3
Open-RS1 (Dang and Ngo 2025)	28.9	21.3	75.0	85.1	30.4	53.2
Open-RS2 (Dang and Ngo 2025)	31.3	22.7	73.0	84.1	29.2	53.7
Open-RS3 (Dang and Ngo 2025)	29.7	24.7	69.2	84.2	28.6	51.8
STILL-3 (Min et al. 2024)	32.5	24.0	66.7	84.4	29.0	52.4
DeepScaleR (Luo et al. 2025)	31.4	26.7	73.6	87.6	31.0	53.7
Our Methods						
Step-RS1	20.0	13.3	50.1	81.4	28.4	42.5
Step-RS2	**36.7**	**26.7**	**76.5**	**88.0**	**32.6**	**54.5**

Table 1: Zero-shot pass@1 performance across benchmarks. **Bold** indicates the highest score per benchmark in the small-scale models. Dash (—) denotes unavailable official scores.

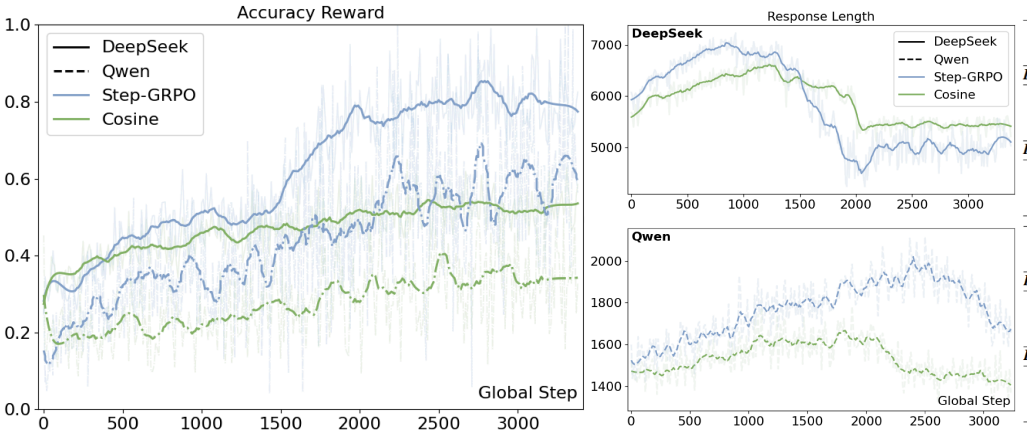
	AIME'24			AMC'23		
Model	**Acc (↑)**	**PRM (↑)**	**Token (↓)**	**Acc (↑)**	**PRM (↑)**	**Token (↓)**
Based on: Qwen2.5 Math 1.5B						
Qwen	7.2	0.37	2 159	26.4	0.48	1 554
LUFFY	16.7	0.35	2 819	47.1	0.51	2 094
Based on Deepseek R1 Distill Qwen 1.5B						
R1-Distill	28.7	0.24	11 815	71.5	0.45	7 906
L1-Exact	24.4	0.30	2 855	70.5	0.52	2 280
L1-Max	27.7	0.29	2 989	73.2	0.54	2 588
Open-RS1	28.9	0.24	11 399	75.0	0.45	8 255
Open-RS2	31.3	0.25	10 896	73.0	0.44	7 203
Open-RS3	29.7	0.25	13 054	69.2	0.45	8 080
STILL-3	32.5	0.27	10 511	66.7	0.45	5 833
DeepScaleR	31.4	0.24	7 310	73.6	0.44	4 643
Our Methods						
Step-RS1	20.0	0.39	2 960	50.1	0.54	1 892
Step-RS2	**36.7**	**0.31**	**2 779**	**76.5**	**0.49**	**3 146**

Table 2: Evaluation of accuracy, PRM scores, and average token counts on AIME'24 and AMC'23 with different RL-based models.

Main Results

To evaluate the performance of Step-RS1 and Step-RS2, we compare them against a wide range of RL-based models across mathematical reasoning benchmarks (Table 1). Step-RS2 achieves state-of-the-art performance on all benchmarks, including both in-distribution datasets (e.g., MATH500, Minerva, Olympiad) and out-of-distribution ones (e.g., AIME'24, AIME'25, AMC'23). Notably, both Step-RS1 and Step-RS2 outperform all models with the same 1.5B backbone, demonstrating the robustness of our Step-GRPO method. PRM-level methods like rStar and Eurus2 Prime, which rely on static reward aggregation, show only marginal gains due to their limited ability to adapt to evolving reasoning patterns. Step-GRPO addresses this by attending to key reasoning steps and capturing their dependencies, enabling more robust and coherent optimization while mitigating reward hacking.

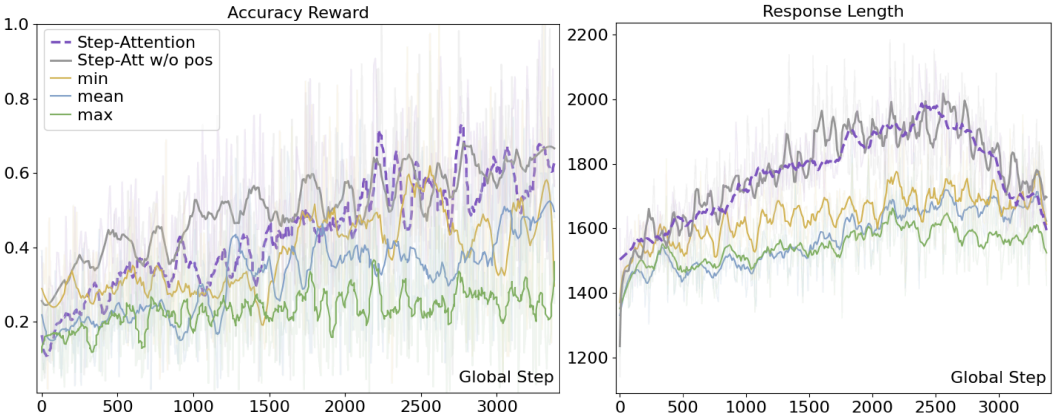
To enable a deeper analysis of the mechanisms behind the results in Table 1, we analyze accuracy, PRM, and reasoning length (token cost) on AIME'24 and AMC'23 in Table 2, with additional insights from MATH500 shown in Figure 1. Prompt-based methods, like the L1 series, show surface-level improvements by primarily increasing PRM scores through discarding poor-quality steps rather than enhancing the effectiveness of each step. This approach, however, struggles with deeper reasoning, especially on more complex tasks, due to rigid length constraints. Rule-based methods like Open-RS offer small reductions in reasoning length, but the results remain overly long and low-quality, indicating overthinking without proportional accuracy improvements. PRM-level methods (e.g., rStar, Eurus2 Prime)



Strategy	MATH500 Acc (↑)	MATH500 PRM (↑)	MATH500 Token (↓)
Based on: Qwen2.5 Math 1.5B			
Qwen (base)	79.8	0.63	621
Cosine	74.4	0.68	567
StepRS1	81.4	0.69	1 555
Based on Deepseek R1 Distill Qwen 1.5B			
R1-Distill	84.6	0.59	4 877
Cosine	81.8	0.61	4 063
StepRS2	88.0	0.71	3 021

Strategy	AIME'24 Acc (↑)	AIME'24 PRM (↑)	AIME'24 Token (↓)
Based on: Qwen2.5 Math 1.5B			
Qwen (base)	7.2	0.37	2 159
Cosine	13.3	0.41	2 101
StepRS1	20.0	0.43	2 760
Based on Deepseek R1 Distill Qwen 1.5B			
R1-Distill	28.7	0.24	11 815
Cosine	31.3	0.26	9 598
StepRS2	36.7	0.41	6 779

Figure 2: Training and evaluation metrics comparing Step-GRPO and the rule-based method (cosine) across two backbone models: Qwen and DeepSeek. Left: Accuracy and reward curves during training. Center: Response length over global steps for both backbones. Right: Final accuracy, PRM score, and token usage on AIME'24 and MATH500 across different strategies.



Strategy	MATH500 Acc (↑)	MATH500 PRM (↑)	MATH500 Token (↓)
Baselines			
Qwen (base)	79.8	0.63	621
LUFFY	80.2	0.68	1 626
PRM Aggregation Strategies			
max	78.8	0.64	1 586
mean	80.0	0.67	1 840
min	80.6	0.68	1 691
Step-Attention	81.4	0.69	1 555
- w/o pos	80.7	0.68	1 732

Strategy	AIME'24 Acc (↑)	AIME'24 PRM (↑)	AIME'24 Token (↓)
Baselines			
Qwen (base)	7.2	0.37	2 159
LUFFY	16.7	0.35	2 819
PRM Aggregation Strategies			
max	10.0	0.38	2 904
mean	10.0	0.39	2 868
min	16.7	0.41	2 897
Step-Attention	20.0	0.43	2 760
- w/o pos	16.7	0.42	3 056

Figure 3: Comparison of static (minimum, mean, maximum) and dynamic (step-Attention, step-Attention w/o position) PRM aggregation strategies on the Qwen backbone. Left: Accuracy and reward curves for five aggregation strategies. Center: Response length during training for each strategy. Right: Final performance on MATH500 and AIME'24 benchmarks, reporting accuracy, PRM score, and token usage.

enhance reasoning quality but lack effective length control, often overfitting to static PRMs and exhibiting reward hacking. Step-GRPO further improves over PRM methods by replacing static aggregation with step-level attention, which dynamically credits informative steps and avoids the rigid pruning strategy used by prompt- and rule-based methods.

Overall, the results validate that while PRMs offer valuable step-level supervision, static reward aggregation leads to overthinking and reward hacking. Addressing these issues requires structured credit assignment—such as step-attention—that dynamically emphasizes informative reasoning steps and enables robust optimization.

Analysis

Comparison with Rule-Based Reward Methods. To better understand the effect of PRM-based reward optimization, we compare Step-GRPO and the rule-based Cosine method on Qwen and DeepSeek backbones, focusing on training-time accuracy and reasoning length (Figure 2, left/center) and final performance on AIME'24 and MATH500 (Figure 2, right). Step-GRPO achieves faster accuracy gains, especially on DeepSeek (0.5–0.8 vs. 0.4–0.6 during steps 1K–2.5K, when response length drops sharply), highlighting the benefit of PRM as process-level supervision. Both methods initially increase reasoning length to improve accuracy (6K–7K vs. 5.5K–6.5K). Still, only Step-GRPO effectively reduces token cost in mid-training (7K→5K vs. 6.5K→5.5K), indicating better control over overthinking. These improvements in training dynamics lead to stronger final performance, confirming that PRM-based supervision enables better reasoning quality and efficiency than rule-based methods.

Comparison of PRM Aggregation Strategies. To evaluate whether dynamic aggregation better leverages PRM signals than static reduction, we compare five strategies on the Qwen backbone (DeepSeek results are included in the

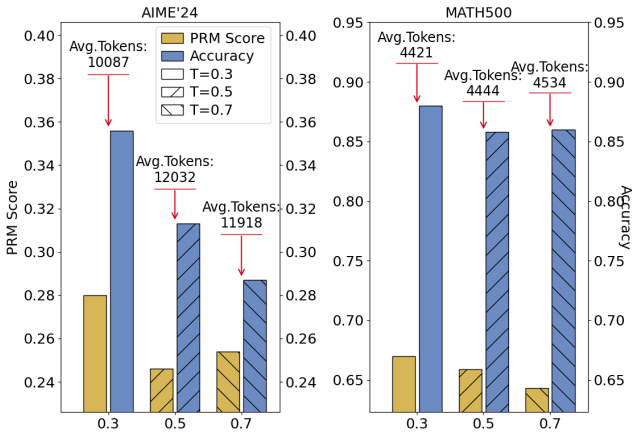


Figure 4: Quantitative analysis of sampling temperature in early-stage PRM-guided training. Bar plots show PRM score, accuracy, and average token usage at step 1.5K under different sampling temperatures ($T = 0.3, 0.5, 0.7$), evaluated on AIME'24 (left) and MATH500 (right).

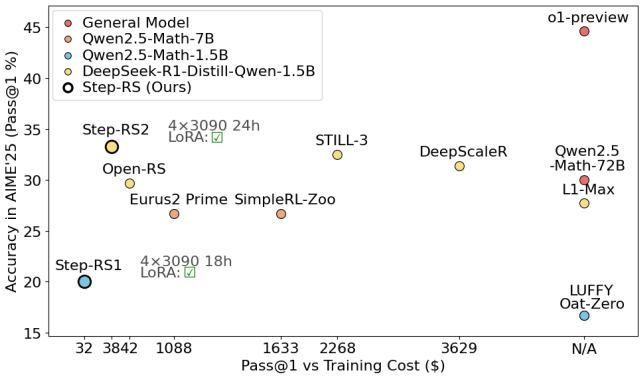


Figure 5: Comparison of training cost versus accuracy on AIME'24. Each point represents a model variant or baseline, showing accuracy (Pass@1) against estimated training cost.

supplementary materials): three static methods (minimum, maximum, mean) and two dynamic variants (step-attention w/o position, full step-attention). As shown in Figure 3, all methods gradually improve accuracy (left), but dynamic ones—especially full step-attention—achieve faster gains; in contrast, max aggregation shows minimal improvement, likely due to overfitting on outlier rewards. For reasoning length (center), dynamic strategies reduce token usage during training (steps 2K–3.3K), while static ones steadily increase. Final performance (right) shows that dynamic methods outperform static ones in accuracy, PRM score, and token efficiency. Full step-attention yields slightly better results than its no-pos variant, underscoring the value of modeling inter-step dependencies. These findings confirm that structure-aware aggregation offers stronger resistance to reward hacking and local optima than static methods.

Early-Stage Analysis of Temperature in Step-GRPO. To assess how sampling temperature influences PRM-guided training in the early stages, we examine the behavior of Step-GRPO within the first 1.5K steps under three temperatures ($\{0.3, 0.5, 0.7\}$) on AIME'24 and MATH500—an early phase marked by a sharp accuracy increase and reasoning length drop (Figure 4). The full training dynamics (0–1.5k) are provided in the supplementary materials. At step 1.5k, lower temperature ($T = 0.3$) yields significantly higher accuracy and PRM scores while reducing average token usage—most notably on AIME'24 (Accuracy: 0.36 vs. 0.32/0.30; Tokens: 10087 vs. 12032/11918)—indicating that more focused sampling improves both the efficiency and effectiveness of PRM-guided optimization. These findings support our hypothesis that lower temperatures benefit PRM-guided training by aligning sampling with reward-sensitive trajectories and suppressing noise.

Training Cost vs. Performance. While Step-GRPO improves accuracy, reasoning quality (via PRM scores), and token efficiency, its cost-effectiveness remains a key practical concern. To evaluate this, we benchmark Step-RS against a range of open-source and commercial RL-based baselines, comparing final accuracy and training cost under comparable settings, as shown in Figure 5. Our results show that Step-RS achieves comparable or superior accuracy and PRM scores while incurring significantly lower cost. For example, **Step-RS2** reaches **36.7%** accuracy on AIME'24 with a total cost of only **$38** (LoRA, 1.5B, 4×4090 GPUs, 24h; 11k samples), outperforming many baselines that rely on larger models and over $1,000 in compute. Despite using LoRA (Hu et al. 2021)—which typically trails full finetuning—Step-RS2 maintains high accuracy, demonstrating robustness and accessibility. Compared to other 1.5B-based methods like **LUFFY**, **Step-RS11** delivers stronger performance with substantially reduced cost, making it especially suitable for scalable or low-resource deployment. These findings confirm that the structured use of static PRM rewards enables effective optimization of reasoning quality and token efficiency without sacrificing performance, while significantly reducing training expenditure.

Conclusion

This study proposes Step-GRPO, a lightweight extension of GRPO that aggregates step-level PRM rewards into trajectory-level signals to supervise the quality of intermediate reasoning steps rather than solely final correctness. We use Step-GRPO to train two 1.5B backbone models, achieving consistent improvements in reasoning accuracy, step quality, and length control across in-domain and out-of-distribution math tasks. Notably, Step-RS2, trained with Step-GRPO, achieves 36.7% accuracy on AIME'24 at a total training cost of only $38 and 11K samples—outperforming models trained with over $1,000 and 40K+ samples, and demonstrating strong cost-efficiency and scalability. Our analysis reveals two key insights: (1) PRM-based supervision offers more reliable guidance than rule-based heuristics; (2) structural reward aggregation enables models to capture inter-step dependencies and avoid reward hacking.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant Nos. 61966038 and 62266051, and by the National Science and Technology Council (NSTC) of Taiwan under Grant No. 113-2221-E-155-046-MY3. The authors would like to thank the anonymous reviewers for their constructive comments.

References

Aggarwal, P.; and Welleck, S. 2025. L1: Controlling How Long A Reasoning Model Thinks With Reinforcement Learning. arXiv:2503.04697.

Cui, G.; Yuan, L.; Wang, Z.; Wang, H.; Li, W.; He, B.; Fan, Y.; Yu, T.; Xu, Q.; Chen, W.; Yuan, J.; Chen, H.; Zhang, K.; Lv, X.; Wang, S.; Yao, Y.; Han, X.; Peng, H.; Cheng, Y.; Liu, Z.; Sun, M.; Zhou, B.; and Ding, N. 2025. Process Reinforcement through Implicit Rewards. arXiv:2502.01456.

Dang, Q.-A.; and Ngo, C. 2025. Reinforcement Learning for Reasoning in Small LLMs: What Works and What Doesn't. arXiv:2503.16219.

Gao, B.; Song, F.; Yang, Z.; Cai, Z.; Miao, Y.; Dong, Q.; Li, L.; Ma, C.; Chen, L.; Xu, R.; et al. 2024. Omni-math: A universal olympiad level mathematic benchmark for large language models. *arXiv preprint arXiv:2410.07985*.

Gao, L.; Schulman, J.; and Hilton, J. 2022. Scaling Laws for Reward Model Overoptimization. arXiv:2210.10760.

Guan, X.; Zhang, L. L.; Liu, Y.; Shang, N.; Sun, Y.; Zhu, Y.; Yang, F.; and Yang, M. 2025. rStar-Math: Small LLMs Can Master Math Reasoning with Self-Evolved Deep Thinking. arXiv:2501.04519.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

He, J.; Liu, J.; Liu, C. Y.; Yan, R.; Wang, C.; Cheng, P.; Zhang, X.; Zhang, F.; Xu, J.; Shen, W.; Li, S.; Zeng, L.; Wei, T.; Cheng, C.; An, B.; Liu, Y.; and Zhou, Y. 2025. Skywork Open Reasoner 1 Technical Report. arXiv:2505.22312.

Hendrycks, D.; Burns, C.; Kadavath, S.; Arora, A.; Basart, S.; Tang, E.; Song, D.; and Steinhardt, J. 2021. Measuring Mathematical Problem Solving With the MATH Dataset. arXiv:2103.03874.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2021. LoRA: Low-Rank Adaptation of Large Language Models. arXiv:2106.09685.

Huang, Z.; Wang, Z.; Xia, S.; Li, X.; Zou, H.; Xu, R.; Fan, R.-Z.; Ye, L.; Chern, E.; Ye, Y.; et al. 2024. Olympicarena: Benchmarking multi-discipline cognitive reasoning for superintelligent ai. *Advances in Neural Information Processing Systems*, 37: 19209–19253.

Jaech, A.; Kalai, A.; Lerer, A.; Richardson, A.; El-Kishky, A.; Low, A.; Helyar, A.; Madry, A.; Beutel, A.; Carney, A.; et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Kang, J.; Li, X. Z.; Chen, X.; Kazemi, A.; Sun, Q.; Chen, B.; Li, D.; He, X.; He, Q.; Wen, F.; Hao, J.; and Yao, J. 2024. MindStar: Enhancing Math Reasoning in Pre-trained LLMs at Inference Time. arXiv:2405.16265.

Kazemnejad, A.; Aghajohari, M.; Portelance, E.; Sordoni, A.; Reddy, S.; Courville, A.; and Roux, N. L. 2025. VinePPO: Refining Credit Assignment in RL Training of LLMs. arXiv:2410.01679.

Leike, J.; Krueger, D.; Everitt, T.; Martic, M.; Maini, V.; and Legg, S. 2018. Scalable agent alignment via reward modeling: a research direction. arXiv:1811.07871.

Lewkowycz, A.; Andreassen, A.; Dohan, D.; Dyer, E.; Michalewski, H.; Ramasesh, V.; Slone, A.; Anil, C.; Schlag, I.; Gutman-Solo, T.; Wu, Y.; Neyshabur, B.; Gur-Ari, G.; and Misra, V. 2022. Solving Quantitative Reasoning Problems with Language Models. arXiv:2206.14858.

Li, J.; Beeching, E.; Tunstall, L.; Lipkin, B.; Soletskyi, R.; Huang, S.; Rasul, K.; Yu, L.; Jiang, A. Q.; Shen, Z.; et al. 2024. Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions. *Hugging Face repository*, 13: 9.

Liu, R.; Gao, J.; Zhao, J.; Zhang, K.; Li, X.; Qi, B.; Ouyang, W.; and Zhou, B. 2025a. Can 1B LLM Surpass 405B LLM? Rethinking Compute-Optimal Test-Time Scaling. arXiv:2502.06703.

Liu, Z.; Chen, C.; Li, W.; Pang, T.; Du, C.; and Lin, M. 2025b. There May Not be Aha Moment in R1-Zero-like Training — A Pilot Study. https://oatllm.notion.site/oat-zero. Notion Blog.

Luo, M.; Tan, S.; Wong, J.; Shi, X.; Tang, W. Y.; Roongta, M.; Cai, C.; Luo, J.; Li, L. E.; Popa, R. A.; and Stoica, I. 2025. DeepScaleR: Surpassing O1-Preview with a 1.5B Model by Scaling RL. https://pretty-radio-b75.notion.site/DeepScaleR-Surpassing-O1-Preview-with-a-1-5B-Model-by-Scaling-RL-19681902c1468005bed8ca303013a4e2. Notion Blog.

Min, Y.; Chen, Z.; Jiang, J.; Chen, J.; Deng, J.; Hu, Y.; Tang, Y.; Wang, J.; Cheng, X.; Song, H.; Zhao, W. X.; Liu, Z.; Wang, Z.; and Wen, J.-R. 2024. Imitate, Explore, and Self-Improve: A Reproduction Report on Slow-thinking Reasoning Systems. arXiv:2412.09413.

Muennighoff, N.; Yang, Z.; Shi, W.; Li, X. L.; Fei-Fei, L.; Hajishirzi, H.; Zettlemoyer, L.; Liang, P.; Candès, E.; and Hashimoto, T. 2025. s1: Simple test-time scaling. arXiv:2501.19393.

OpenAI. 2024. Introducing OpenAI O1 Preview. https://openai.com/index/introducing-openai-o1-preview/. Accessed: 2025-06-03.

OpenAI. 2024. Learning to Reason with LLMs.

Qu, Y.; Zhang, T.; Garg, N.; and Kumar, A. 2024. Recursive Introspection: Teaching Language Model Agents How to Self-Improve. arXiv:2407.18219.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y. K.; Wu, Y.; and Guo, D. 2024. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. arXiv:2402.03300.

Shen, T.; Cambria, E.; Wang, J.; Cai, Y.; and Zhang, X. 2025a. Insight at the right spot: Provide decisive subgraph information to Graph LLM with reinforcement learning. *Information Fusion*, 117: 102860.

Shen, T.; Mao, R.; Wang, J.; Zhang, X.; and Cambria, E. 2025b. Flow-guided Direct Preference Optimization for Knowledge Graph Reasoning with Trees. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '25, 1165–1175. New York, NY, USA: Association for Computing Machinery. ISBN 9798400715921.

Snell, C.; Lee, J.; Xu, K.; and Kumar, A. 2024. Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters. arXiv:2408.03314.

Song, M.; Zheng, M.; Li, Z.; Yang, W.; Luo, X.; Pan, Y.; and Zhang, F. 2025. FastCuRL: Curriculum Reinforcement Learning with Stage-wise Context Scaling for Efficient Training R1-like Reasoning Models. arXiv:2503.17287.

Sui, Y.; Chuang, Y.-N.; Wang, G.; Zhang, J.; Zhang, T.; Yuan, J.; Liu, H.; Wen, A.; Zhong, S.; Chen, H.; and Hu, X. 2025. Stop Overthinking: A Survey on Efficient Reasoning for Large Language Models. arXiv:2503.16419.

Team, K.; Du, A.; Gao, B.; Xing, B.; Jiang, C.; Chen, C.; Li, C.; Xiao, C.; Du, C.; Liao, C.; et al. 2025. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*.

Uesato, J.; Kushman, N.; Kumar, R.; Song, F.; Siegel, N.; Wang, L.; Creswell, A.; Irving, G.; and Higgins, I. 2022. Solving math word problems with process- and outcome-based feedback. arXiv:2211.14275.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2023. Attention Is All You Need. arXiv:1706.03762.

Wang, J.; Fang, M.; Wan, Z.; Wen, M.; Zhu, J.; Liu, A.; Gong, Z.; Song, Y.; Chen, L.; Ni, L. M.; Yang, L.; Wen, Y.; and Zhang, W. 2024a. OpenR: An Open Source Framework for Advanced Reasoning with Large Language Models. arXiv:2410.09671.

Wang, P.; Li, L.; Shao, Z.; Xu, R. X.; Dai, D.; Li, Y.; Chen, D.; Wu, Y.; and Sui, Z. 2024b. Math-Shepherd: Verify and Reinforce LLMs Step-by-step without Human Annotations. arXiv:2312.08935.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Narang, S.; Chowdhery, A.; and Zhou, D. 2023. Self-Consistency Improves Chain of Thought Reasoning in Language Models. arXiv:2203.11171.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.; Le, Q.; and Zhou, D. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. arXiv:2201.11903.

Wu, Y.; Sun, Z.; Li, S.; Welleck, S.; and Yang, Y. 2025. Inference Scaling Laws: An Empirical Analysis of Compute-Optimal Inference for Problem-Solving with Language Models. arXiv:2408.00724.

Xie, T.; Gao, Z.; Ren, Q.; Luo, H.; Hong, Y.; Dai, B.; Zhou, J.; Qiu, K.; Wu, Z.; and Luo, C. 2025. Logic-RL: Unleashing LLM Reasoning with Rule-Based Reinforcement Learning. arXiv:2502.14768.

Xie, Y.; Kawaguchi, K.; Zhao, Y.; Zhao, X.; Kan, M.-Y.; He, J.; and Xie, Q. 2023. Self-Evaluation Guided Beam Search for Reasoning. arXiv:2305.00633.

Yan, J.; Li, Y.; Hu, Z.; Wang, Z.; Cui, G.; Qu, X.; Cheng, Y.; and Zhang, Y. 2025. Learning to Reason under Off-Policy Guidance. arXiv:2504.14945.

Yang, A.; Li, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Gao, C.; Huang, C.; Lv, C.; et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Yang, A.; Zhang, B.; Hui, B.; Gao, B.; Yu, B.; Li, C.; Liu, D.; Tu, J.; Zhou, J.; Lin, J.; et al. 2024. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T. L.; Cao, Y.; and Narasimhan, K. 2023. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. arXiv:2305.10601.

Yuan, L.; Cai, Y.; Shen, X.; Li, Q.; Huang, Q.; Deng, Z.; and Wang, T. 2025. Collaborative Multi-LoRA Experts with Achievement-based Multi-Tasks Loss for Unified Multimodal Information Extraction. In Kwok, J., ed., *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, 6940–6948. International Joint Conferences on Artificial Intelligence Organization. Main Track.

Yuan, L.; Cai, Y.; Wang, J.; and Li, Q. 2023. Joint multimodal entity-relation extraction based on edge-enhanced graph alignment network and word-pair relation tagging. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 11051–11059.

Yuan, L.; Li, W.; Chen, H.; Cui, G.; Ding, N.; Zhang, K.; Zhou, B.; Liu, Z.; and Peng, H. 2024. Free Process Rewards without Process Labels. arXiv:2412.01981.

Zeng, W.; Huang, Y.; Liu, Q.; Liu, W.; He, K.; Ma, Z.; and He, J. 2025. SimpleRL-Zoo: Investigating and Taming Zero Reinforcement Learning for Open Base Models in the Wild. arXiv:2503.18892.

Zhang, J.; Lin, N.; Hou, L.; Feng, L.; and Li, J. 2025a. AdaptThink: Reasoning Models Can Learn When to Think. arXiv:2505.13417.

Zhang, Z.; Zheng, C.; Wu, Y.; Zhang, B.; Lin, R.; Yu, B.; Liu, D.; Zhou, J.; and Lin, J. 2025b. The Lessons of Developing Process Reward Models in Mathematical Reasoning. arXiv:2501.07301.

Zheng, C.; Zhang, Z.; Zhang, B.; Lin, R.; Lu, K.; Yu, B.; Liu, D.; Zhou, J.; and Lin, J. 2025. ProcessBench: Identifying Process Errors in Mathematical Reasoning. arXiv:2412.06559.

Zuo, Y.; Zhang, K.; Sheng, L.; Qu, S.; Cui, G.; Zhu, X.; Li, H.; Zhang, Y.; Long, X.; Hua, E.; Qi, B.; Sun, Y.; Ma, Z.; Yuan, L.; Ding, N.; and Zhou, B. 2025. TTRL: Test-Time Reinforcement Learning. arXiv:2504.16084.