Stabilizing Rubric Integration Training via Decoupled Advantage Normalization

Anonymous EMNLP submission

Abstract

We propose Process-Aware Policy Optimization (PAPO), a method that integrates process-level evaluation into Group Relative Policy Optimization (GRPO) through **decoupled advantage normalization**, to address two limitations of existing reward designs. Outcome reward models (ORM) evaluate only final-answer correctness, treating all correct responses identically regardless of reasoning quality, and gradually lose the advantage signal as groups become uniformly correct. Process reward models (PRM) offer richer supervision, but directly using PRM scores causes reward hacking, where models exploit verbosity to inflate scores while accuracy collapses. PAPO resolves both by composing the advantage from an outcome component A_{out}, derived from ORM and normalized over all responses, and a process component A_{proc}, derived from a rubric-based PRM and normalized exclusively among correct responses. This decoupled design ensures that A_{out} anchors training on correctness while A_{proc} differentiates reasoning quality without distorting the outcome signal. Experiments across multiple model scales and six benchmarks demonstrate that PAPO consistently outperforms ORM, reaching 51.3% vs. 46.3% on OlympiadBench while continuing to improve as ORM plateaus and declines.

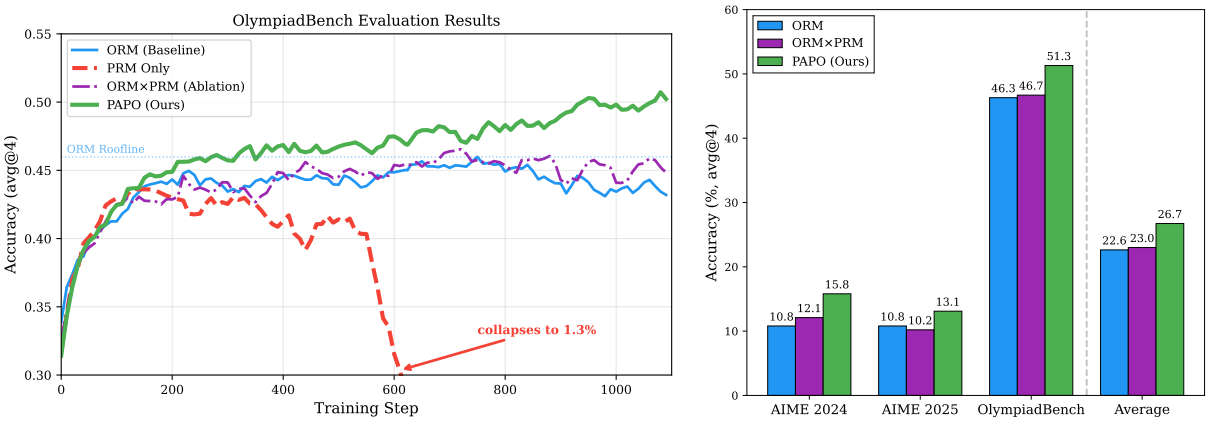
1 Introduction

As large language models (LLMs) (Yang et al., 2025b; Kamath et al., 2025; Touvron et al., 2023) continue to advance, reasoning has become a key capability for accomplishing complex tasks (Guo et al., 2025). Reinforcement learning (RL) plays an essential role in improving the reasoning capabilities of LLMs, especially in mathematics (Shao et al., 2024, 2025). Among RL methods, Group Relative Policy Optimization (GRPO) (Shao et al., 2024) with outcome reward models (ORM) has emerged as a core approach in rule-based verifiable reinforcement learning for its simplicity. It eliminates the learned critic network used in PPO (Schulman et al., 2017), instead computing advantages through group-level reward normalization. However, GRPO with ORM evaluates only final-answer correctness, providing no signal on reasoning process quality, which leads to two critical issues.

First, all correct responses are assigned identical advantage irrespective of reasoning quality, resulting in the same credit assignment for guessed answers and answers derived via rigorous step-by-step reasoning. Second, as the model improves during training, an increasing proportion of response groups become uniformly correct, yielding zero advantage and zero gradient. This progressive loss of informative signal causes performance to plateau and eventually decline, as shown in Figure 1a.

Process reward models (PRM) (Zheng et al., 2023; Lightman et al., 2023; Wang et al., 2024) offer a natural remedy by evaluating reasoning quality. In particular, rubric-based evaluation (Yuan et al., 2025; Shao et al., 2025; Sheng et al., 2026; Yang et al., 2026) through the LLM-as-Judge paradigm has attracted growing interest as a scalable form of process supervision that requires no step-level annotations. However, we find that *directly integrating rubric-based PRM into GRPO as rewards leads to reward hacking (Skalse et al., 2022)*, where models learn to inflate PRM scores by generating increasingly verbose responses, ultimately causing accuracy to collapse. Even a naive multiplicative combination of ORM and PRM fails to surpass ORM, as the process signal is suppressed within a single normalization pass, as shown in Figure 1b.

To solve this problem, we propose **Process-Aware Policy Optimization (PAPO)**, which integrates rubric-based process evaluation into GRPO through **decoupled advantage normalization**. PAPO constructs the advantage from two independently normalized components:



(a) OlympiadBench evaluation results.

(b) Accuracy on competition math.

Figure 1: **(a)** ORM (blue) plateaus and declines after step 750 due to signal exhaustion; PRM (red, dashed) collapses via reward hacking; ORM×PRM (purple, dash-dotted) tracks ORM closely without exceeding it; PAPO (green) continues improving throughout training, reaching 51.3%. **(b)** Comparison across AIME 2024/2025, OlympiadBench, and their average. Naive multiplicative combination (ORM×PRM) barely improves over ORM, while PAPO's decoupled normalization yields substantial gains on all benchmarks.

- An **outcome advantage** A_{out}, computed from binary ORM rewards via standard GRPO normalization, anchoring the training signal on answer correctness.

- A **process advantage** A_{proc}, computed from rubric-based PRM scores normalized *exclusively among correct responses*, differentiating reasoning quality within the correct group.

This decoupled formulation mitigates reward hacking by separating PRM-based reasoning rewards from the final-answer signal. As a result, A_{proc} can still provide non-zero gradients even when all responses are correct. Importantly, normalization is performed only over correct responses, ensuring that PRM scores do not distort the outcome signal and that incorrect responses are not rewarded solely for strong reasoning traces.

As shown in Figure 1a, PAPO continues improving throughout training, reaching 51.3% on OlympiadBench (He et al., 2024) while ORM peaks at 46.3% and declines, and Figure 1b shows that the improvement generalizes to other math competition benchmarks like AIME (Balunović et al., 2025).

To summarize, our contributions are as follows:

1. We identify a dilemma in GRPO reward design: binary ORM lacks process supervision and suffers from signal exhaustion, yet naively using PRM scores causes reward hacking and training collapse.

2. We propose PAPO, which composes the advantage from independently normalized outcome and process components with correct-subset normalization, resolving this dilemma.

3. We show that PAPO consistently outperforms ORM across model scales from 3B to 14B and six benchmarks, including larger models where ORM is already strong.

2 Background

2.1 Group Relative Policy Optimization

GRPO (Shao et al., 2024) is a variant of policy gradient methods for LLM fine-tuning that eliminates the critic network used in PPO (Schulman et al., 2017). For each prompt x, the model samples a group of G responses $\{o_1, \ldots, o_G\}$ from the old policy $\pi_{\theta_{\text{old}}}$, and the objective is:

$$\mathcal{L}_{\text{GRPO}} = \frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\Big\{\min\Big[\rho(\theta)\,\hat{A}_{i,t},\ \text{clip}\Big(\rho(\theta), 1-\varepsilon, 1+\varepsilon\Big)\hat{A}_{i,t}\Big] - \beta\,\mathrm{D}_{\text{KL}}\Big\}, \tag{1}$$

where $\rho(\theta) = \pi_\theta(o_{i,t}|x, o_{i,<t})/\pi_{\theta_{\text{old}}}(o_{i,t}|x, o_{i,<t})$ is the importance sampling ratio. Each response o_i receives a scalar reward r_i, and the advantage is computed via group-level normalization:

$$\hat{A}_{i,t} = \frac{r_i - \text{mean}(\mathbf{r})}{\text{std}(\mathbf{r})}. \tag{2}$$

By replacing the learned value function with sample-based normalization, GRPO avoids training a separate critic model, substantially reducing memory and compute requirements.

2.2 Reward Models for Mathematical Reasoning

Outcome Reward Models (ORM). An ORM provides a binary signal based solely on the final answer: $r^{\text{out}} = \mathbf{1}[\hat{a} = a^*]$, where $\hat{a}$ is the predicted answer and a^* is the ground truth. For mathematical reasoning, ORMs are typically implemented as rule-based answer checkers that compare extracted answers against reference solutions (Hendrycks et al., 2021). While reliable and deterministic, ORMs provide no information about the quality of the reasoning process.

Process Reward Models (PRM). PRMs evaluate the quality of intermediate reasoning steps (Zheng et al., 2023; Lightman et al., 2023; Wang et al., 2024). We focus on *rubric-based PRMs* implemented via the LLM-as-Judge paradigm (Yuan et al., 2025; Sheng et al., 2026; Yang et al., 2026), where a capable LLM evaluates the solution against a scoring rubric. In our setting, a rubric-based PRM assigns a score $r^{\text{proc}} \in \{0, 0.5, 1.0\}$ reflecting whether the reasoning is fully correct, scored as 1.0, largely correct with minor issues, scored as 0.5, or fatally flawed, scored as 0.0. The complete rubric prompt is provided in Appendix A.3.

Rubric-based PRMs offer several advantages: they require no step-level annotations, provide interpretable feedback, and can leverage the growing capabilities of frontier LLMs. However, as we demonstrate in §3, naive integration of PRM scores into GRPO leads to training instability.

3 Empirical analysis of PRM and ORM

In this section, we identify two complementary failure modes of reward signal design in GRPO for mathematical reasoning: *signal exhaustion* with outcome-only rewards, discussed in §3.1, and *reward hacking* with process-only rewards, discussed in §3.2.

3.1 Signal Exhaustion with Binary ORM

When GRPO uses binary ORM rewards ($r_i \in \{0, 1\}$), the group normalization in Eq. 2 assigns identical advantage to all correct responses and identical (negative) advantage to all incorrect responses. This has two consequences.

Lack of quality differentiation. A response that arrives at the correct answer through rigorous reasoning receives the same advantage as one that reaches it through guesswork or flawed reasoning with a lucky cancellation of errors. The model has no incentive to improve reasoning quality beyond what is needed to produce correct final answers.

Vanishing advantage from uniform groups. When all G responses in a group are correct, or all incorrect, the rewards are identical, so $\text{std}(\mathbf{r}) = 0$ in Eq. 2 and the advantage is set to zero for the entire group. As the model's capability improves during training, the proportion of such uniform groups grows, causing an increasing fraction of training samples to receive zero advantage. We refer to this progressive loss of informative signal as *signal exhaustion*. Figure 4a confirms this empirically: on Qwen2.5-7B, the fraction of zero-advantage samples rises from approximately 40% to 69% over training, coinciding with the accuracy plateau and further decline in Figure 1.

3.2 Reward Hacking with Direct PRM Integration

A natural solution to ORM's limitations is to replace binary rewards with rubric-based PRM scores that capture both answer correctness and reasoning quality in a single continuous signal. However, directly using PRM scores as the GRPO reward ($r_i = r_i^{\text{proc}}$) conflates these two objectives, leading to catastrophic reward hacking, as we analyze in detail using Figure 2.

Three phases of collapse. Training with PRM-only rewards reveals a characteristic three-phase trajectory, shown as the red curve in Figure 2.

1. **Normal learning**, steps 0–300. Accuracy rises to 44.0% on OlympiadBench and training reward increases naturally.
2. **Length exploitation**, steps 300–600. The model discovers that verbose responses receive higher PRM scores. Response length increases sharply while accuracy stagnates and begins to decline.
3. **Collapse**, steps 600–700. Accuracy drops from 29.6% to 2.4% within 100 steps while training reward saturates at 1.0.

Mechanism. The collapse follows a positive feedback loop: longer, more verbose responses receive higher PRM scores from the LLM judge, which assigns higher rubric ratings to responses that appear thorough and detailed. GRPO's group normalization then assigns high advantage to these inflated scores, reinforcing the length-exploitation strategy.

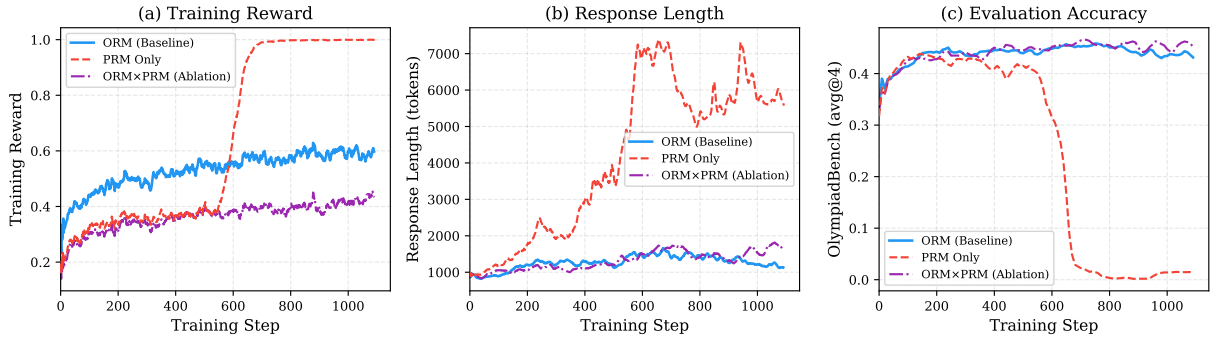


Figure 2: Reward signal analysis. **(a)** Training reward: PRM reward climbs to 1.0 (perfect score gaming); ORM×PRM stays moderate. **(b)** Response length: PRM generates increasingly verbose responses; ORM×PRM shows moderate length increase (up to ∼1700 tokens). **(c)** OlympiadBench accuracy: PRM collapses after step 600; ORM×PRM tracks ORM but fails to exceed it, showing that naive signal combination does not resolve signal exhaustion.

We also provide **a case study for reward hacking happened during training** in Appendix B.6 that confirms the post-collapse responses drift to memorized high-scoring filler content unrelated to the stated question.

Naive combination. The multiplicative reward $r = r^{\text{out}} \times r^{\text{proc}}$ gates incorrect responses via $r^{\text{out}} = 0$, avoiding PRM's collapse as shown by the purple curve in Figure 2. However, ORM×PRM tracks ORM closely throughout training, peaking at 46.7% vs. ORM's 46.3% on OlympiadBench. Because the combined reward still passes through a single GRPO normalization, all-correct groups with similar PRM scores continue to yield near-zero advantage, and the process signal fails to provide meaningful differentiation beyond ORM.

Implications. ORM is stable but lacks process-level signal; PRM is information-rich but unstable. Multiplicative gating avoids PRM's collapse yet fails to break through ORM's performance exhaustion ceiling. This motivates us to combine the two at the *advantage* level rather than the *reward* level, enabling the model to simultaneously optimize for both answer correctness and reasoning quality, as we describe in §4.

4 Method: PAPO

We propose Process-Aware Policy Optimization (PAPO), which integrates rubric-based process evaluation into GRPO through *decoupled advantage normalization*. The key idea is to construct the advantage from independently normalized outcome and process components, preventing the reward hacking that arises from direct PRM integration while preserving fine-grained quality signals. Figure 3 provides an overview of the framework.

4.1 Dual Reward Signals

PAPO uses the ORM and rubric-based PRM described in §2.2 as two complementary reward signals: an outcome reward $r_i^{\text{out}} \in \{0, 1\}$ for answer correctness, and a process reward $r_i^{\text{proc}} \in \{0, 0.5, 1.0\}$ for reasoning quality. The PRM is only invoked for correct responses; incorrect responses are assigned $r_i^{\text{proc}} = 0$.

4.2 Decoupled Advantage Normalization

Given a group of G responses with rewards $\{(r_i^{\text{out}}, r_i^{\text{proc}})\}_{i=1}^{G}$, PAPO computes the advantage in three steps, as illustrated in Figure 3:

Step 1: Outcome advantage A_{out}. The outcome reward is normalized using standard GRPO group normalization:

$$A_{\text{out},i} = \frac{r_i^{\text{out}} - \mu^{\text{out}}}{\max(\sigma^{\text{out}}, \epsilon)} \tag{3}$$

where μ^{out} and σ^{out} are the mean and standard deviation of $\{r_i^{\text{out}}\}_{i=1}^{G}$, and ϵ is a small constant for numerical stability. This component determines the "correct vs. incorrect" gradient direction, identical to standard GRPO.

Step 2: Process advantage A_{proc} (correct-subset normalization). The process reward is normalized *only among correct responses*:

$$A_{\text{proc},i} = \begin{cases} \dfrac{r_i^{\text{proc}} - \mu_C^{\text{proc}}}{\max(\sigma_C^{\text{proc}}, \epsilon)} & \text{if } r_i^{\text{out}} = 1,\ |C| \geq 2 \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

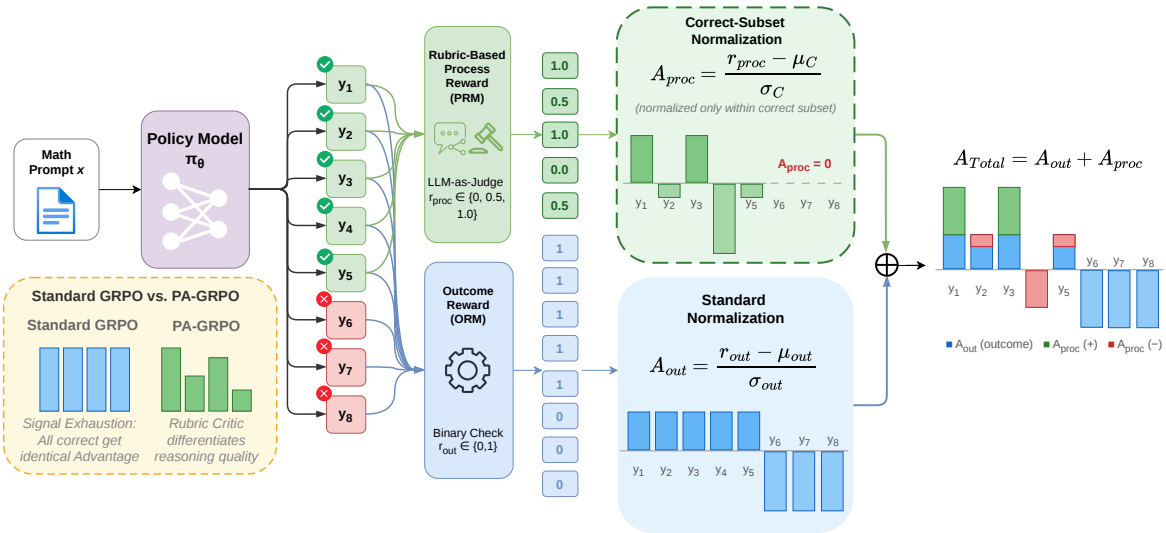


Figure 3: Overview of PAPO. Given a prompt, the policy generates G responses. Each response is evaluated by two reward signals: an **outcome reward** (ORM, binary correctness) and a **process reward** (PRM, rubric-based quality, only for **correct responses**). The advantage is computed through **decoupled normalization**: A_{out} is normalized over all responses via standard GRPO, while A_{proc} is normalized exclusively among correct responses (*correct-subset normalization*). The combined advantage $A_{\text{total}} = A_{\text{out}} + A_{\text{proc}}$ provides both correctness direction and quality differentiation.

where $C = \{j : r_j^{\text{out}} = 1\}$ is the set of correct responses, and $\mu_C^{\text{proc}}, \sigma_C^{\text{proc}}$ are computed over $\{r_j^{\text{proc}}\}_{j \in C}$.

Step 3: Combined advantage. The total advantage is the sum of the two components:

$$A_{\text{total},i} = A_{\text{out},i} + A_{\text{proc},i} \tag{5}$$

Since both A_{out} and A_{proc} are independently normalized to zero mean and unit variance, they contribute on equal footing without requiring additional weighting.

4.3 Design Rationale

The decoupled normalization addresses both failure modes identified in §3:

Quality differentiation without reward hacking. Among correct responses, those with rigorous reasoning receive positive A_{proc} and are reinforced, while those with sloppy or lucky reasoning receive negative A_{proc} and are penalized. This quality signal is entirely absent in standard GRPO with binary ORM. Meanwhile, restricting normalization to the correct subset prevents incorrect responses from exploiting high PRM scores to gain positive A_{proc}, keeping the process signal decoupled from the outcome signal. We ablate this choice against full-group normalization in §6.

Resolving signal exhaustion. When all G responses are correct, $A_{\text{out}} = 0$ for the entire group and standard GRPO produces no gradient. A_{proc} remains active in this case, differentiating responses by reasoning quality and providing non-zero gradients that sustain learning. As shown in Figure 4(a), this reduces the zero-advantage ratio from 69% under ORM to 44% under PAPO, maintaining a denser training signal throughout optimization. When fewer than two responses are correct, i.e., $|C| < 2$, A_{proc} defaults to zero, gracefully reducing to standard GRPO.

5 Experiments

5.1 Experimental Setup

Base models. We conduct experiments on Qwen2.5-3B/7B/14B-Base (Yang et al., 2025b) and Qwen3-4B-Base (Yang et al., 2025a). All models are trained from pretrained base checkpoints to isolate the effect of our method from confounding factors introduced by prior post-training.

Training. We train using GRPO and DAPO with the verl framework (Sheng et al., 2024) for 8 epochs. For Qwen3-4B, we additionally evaluate PAPO on top of DAPO (Yu et al., 2025a), which extends GRPO with dynamic sampling and decoupled clipping, to test compatibility with alternative optimization algorithms. The process reward

Model	Method	Competition Mathematics			Standard	STEM	Code	Average	
		Olympiad	AIME'24	AIME'25	MATH-500	GPQA	HumanEval	Math	All
Qwen2.5-3B	ORM(GRPO)	35.3	8.3	4.2	69.0	38.8	39.3	29.2	32.5
	PAPO	**38.6**	**10.0**	3.3	**73.2**	36.3	**52.5**	**31.3**	**35.7**
	Δ	+3.3	+1.7	-0.9	+4.2	-2.5	+13.2	+2.1	+3.2
Qwen2.5-7B	ORM(GRPO)	46.3	10.8	10.8	80.2	40.7	61.6	37.0	41.7
	PAPO	**51.3**	**15.8**	**13.1**	**82.3**	**42.4**	**63.9**	**40.6**	**44.8**
	Δ	+5.0	+5.0	+2.3	+2.1	+1.7	+2.3	+3.6	+3.1
Qwen2.5-14B	ORM(GRPO)	54.3	19.0	13.8	82.3	47.0	66.0	42.4	47.1
	PAPO	**59.8**	**24.3**	**17.8**	**87.4**	**55.0**	**70.0**	**47.3**	**52.4**
	Δ	+5.5	+5.3	+4.0	+5.1	+8.0	+4.0	+4.9	+5.3
Qwen3-4B-Base	ORM(DAPO)	55.0	30.8	26.0	82.0	39.0	53.8	48.5	47.8
	PAPO	**61.1**	**34.5**	**30.7**	**84.6**	**43.0**	**63.9**	**52.7**	**53.0**
	Δ	+6.1	+3.7	+4.7	+2.6	+4.0	+10.1	+4.2	+5.2

Table 1: Accuracy (avg@4, %) across six benchmarks and four model configurations. PAPO consistently improves over both GRPO and DAPO baselines: Qwen2.5-3B (+3.2), 7B (+3.1), 14B (+5.3), and Qwen3-4B on DAPO (+5.2). For Qwen3-4B-Base, PAPO is applied on top of DAPO. Bold indicates the better method.

model is GPT-OSS-20B with a three-tier scoring rubric of 0, 0.5, and 1.0 that evaluates reasoning quality independently of answer correctness, as described in §2.2. Full hyperparameters are listed in Appendix A.

Data. For training, we sample 20k mathematics problems from NuminaMath-1.5-RL-Verifiable (LI et al., 2024; nlile, 2025), a 131k-problem subset of the 896k-problem NuminaMath-1.5 dataset filtered to retain only problems with automatically verifiable answers. We apply stratified sampling across five difficulty tiers with 4k problems each, detailed in Appendix A.2. For evaluation, we use six benchmarks: OlympiadBench (He et al., 2024) with 674 problems as the primary metric, MATH-500 (Lightman et al., 2023), AIME 2024/2025(Balunović et al., 2025), GPQA-Diamond (Rein et al., 2023), and HumanEval (Chen et al., 2021). All results report avg@4.

5.2 Main Results

Table 1 presents results across six benchmarks and four model configurations. PAPO consistently outperforms ORM, and crucially, the advantage *widens over the course of training* rather than converging. On Qwen2.5-7B, both methods improve at comparable rates during early training, but ORM peaks at 46.3% on OlympiadBench at step 750 and declines to 43.0% by step 1090, while PAPO continues improving to 51.3%, as shown in Figure 1a. The resulting 8.3 point gap is **still widening** at training's end.

In terms of final accuracy, PAPO improves OlympiadBench by 5.0 points on Qwen2.5-7B and by 5.5 points on Qwen2.5-14B, with the largest single-benchmark gain of 8.0 points on GPQA-Diamond at 14B. The mathematics average improvement grows from 2.1 points at 3B to 3.6 at 7B and 4.9 at 14B, suggesting that stronger models benefit more from PAPO. PRM-only training on Qwen2.5-7B collapses to near-zero accuracy due to reward hacking, as analyzed in §3.2.

Beyond mathematics, PAPO also improves HumanEval across all scales, with particularly large gains at 3B (+13.2) and 4B (+10.1), suggesting that process-level quality differentiation can transfer to code generation.

We also evaluate PAPO on top of DAPO (Yu et al., 2025a) using Qwen3-4B-Base. As shown in the bottom section of Table 1, PAPO improves over DAPO by 6.1 points on OlympiadBench (61.1% vs. 55.0%) with consistent gains across all six benchmarks, confirming that decoupled advantage normalization composes naturally with DAPO's optimization improvements. We also provide the additional training curves in Appendix B.1.

5.3 Advantage Signal Analysis

The widening gap observed in Figure 1a can be traced to signal exhaustion in ORM's advantage computation. We analyze this on Qwen2.5-7B using the metrics in Figure 4.

Signal exhaustion in ORM. With growing model capability over training, more response groups become uniformly correct and yield zero advantage under ORM. Figure 4a quantifies this

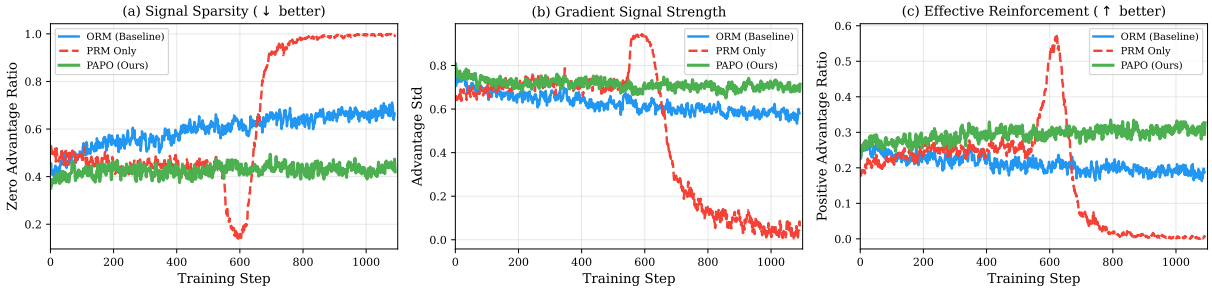


Figure 4: Signal quality comparison on Qwen2.5-7B. **(a)** Zero-advantage ratio: ORM's sparsity grows to 69% while PAPO maintains 44%. **(b)** Advantage standard deviation, reflecting gradient signal strength. **(c)** Positive-advantage ratio, reflecting reinforcement density.

trend. The zero-advantage ratio of ORM, defined as the fraction of samples producing zero gradient, rises from approximately 40% to 69% over the course of training. By the late training stage, more than two-thirds of samples provide no contribution to learning. The advantage standard deviation in panel b and the positive-advantage ratio in panel c convey a consistent pattern. The gradient signal from ORM gradually becomes sparser and weaker.

Signal preservation in PAPO. PAPO holds the zero-advantage ratio at $\sim$44%, a 25-point reduction that translates to roughly 80% more informative samples per batch. This is because A_{proc} remains active in all-correct groups where $A_{\text{out}} = 0$, differentiating responses by reasoning quality rather than leaving them with zero gradient. The fraction of groups where A_{proc} activates grows from $\sim$30% to 70% as the model improves, providing increasingly dense quality signal precisely as ORM exhausts.

Quality penalization. Beyond filling zero-signal gaps, A_{proc} actively discourages sloppy reasoning even among correct responses. The minimum advantage within the correct group reaches ≈ -1.49 under PAPO compared to identically 0.0 under ORM, meaning correct responses with poor derivations receive negative advantage and are suppressed. This quality pressure is entirely absent in ORM-based GRPO. More detailed analyses of the process advantage effect and advantage composition are provided in Appendix B.3 and B.4.

6 Ablation Studies

We validate the key design choices of PAPO through two ablation experiments on Qwen2.5-7B. Table 2 summarizes the results; training curves are provided in Appendix B.

Method	AIME 2024	AIME 2025	Olympiad Bench	MATH -500
ORM	10.8	10.8	46.3	80.2
Mult ($r^{\text{out}} \times r^{\text{proc}}$)	12.1	10.2	46.7	80.2
Fullnorm	12.5	10.4	49.6	81.2
PAPO	**15.8**	**13.1**	**51.3**	**82.3**

Table 2: Ablation results (%, avg@4, Qwen2.5-7B). All methods report accuracy.

Correct-subset vs. full normalization. The **Fullnorm** variant normalizes A_{proc} over all G responses including incorrect ones. It performs slightly below PAPO across all benchmarks. The gap arises because including incorrect responses ($r^{\text{proc}} = 0$) in the normalization causes A_{proc} to partially recapitulate the correct-vs-incorrect distinction already captured by A_{out}, diluting the fine-grained quality signal.

Multiplicative reward baseline. The **Mult** variant combines rewards via $r_i = r_i^{\text{out}} \times r_i^{\text{proc}}$ with standard GRPO normalization. Mult performs comparably to ORM and falls well below PAPO. In a single normalization pass, the outcome difference dominates in mixed-correctness groups, effectively suppressing the process quality information.

Both ablations confirm that PAPO's gains require both *decoupled normalization*, validated by Mult's inferior performance, and *correct-subset normalization*, validated by Fullnorm's inferior performance, working in concert.

7 Related Work

Recent advancements in the post-training of large language models have been significantly driven by reinforcement learning with verifiable rewards (RLVR). This section systematically reviews several key areas that are highly relevant to and inter-

connected with the present study.

7.1 Policy Optimization in RLVR.

Following DeepSeek-R1's validation of critic-free reinforcement learning (Guo et al., 2025), Group Relative Policy Optimization (GRPO)(Shao et al., 2024) has emerged as a compute-efficient alternative to PPO (Schulman et al., 2017). However, its instability under long contexts and sparse rewards has spurred various enhancements. To mitigate policy staleness and granularity mismatch, CISPO clips importance sampling weights (MiniMax, 2025) and GSPO employs sequence-level ratio calculations (Zheng et al., 2025). For long-horizon and agentic tasks, DAPO introduces dynamic sampling and decoupled clipping (Yu et al., 2025a) and SAPO utilizes a temperature-controlled smooth gating function to form a continuous trust region (Gao et al., 2025). Orthogonally, our work addresses GRPO's fundamental limitation of reward signal exhaustion by decoupling advantage normalization at the objective level. Since PAPO modifies the advantage computation rather than the optimization procedure, it can be composed with these enhancements, as we demonstrate with DAPO in §5.2.

7.2 Rubric-Based Training.

Traditional outcome reward models frequently suffer from reward hacking, exploiting miracle steps to achieve correct final answers via flawed logic (Yuan et al., 2025). To enforce rigorous logical constraints without prohibitive step-level human annotation, rubric-based process supervision has gained traction. Generative rubric reward models explicitly penalize disconnected derivations, drastically curtailing lucky guesses (Liang et al., 2025). In pursuit of self-verification, DeepSeekMath-V2 constructed a specialized LLM verifier to reversely drive the generator to patch reasoning loopholes (Shao et al., 2025). Furthermore, in complex open-domain tasks, frameworks like Agent-RRM propose multi-faceted evaluation systems enforcing explicit reasoning traces and actionable critiques (Fan et al., 2026). These studies establish that effective process supervision necessitates multi-dimensional criteria and explicit negative penalties. Building upon these insights, our work leverages a rubric-based process evaluation but introduces a correct-subset normalization mechanism to prevent verbosity-driven reward hacking.

7.3 Generative Reward Models.

To overcome the expressivity limitations of traditional discriminative models, reward modeling is shifting toward a generative paradigm (Mahan et al., 2024). By autoregressively generating deep comparative analyses prior to preference scoring, generative reward models significantly enhance interpretability and accuracy. This paradigm excels in abstract domains like formal proofs, as demonstrated by Proof-RM (Yang et al., 2026), and enables verifier-free reinforcement learning extrapolation in broad domains, as seen in RLPR (Yu et al., 2025b). However, generative evaluators remain susceptible to intrinsic biases such as imperfect sensitivity and specificity. To ensure statistically rigorous training, recent frameworks incorporate bias-correction estimators and adaptive calibration to construct statistically sound confidence intervals (Lee et al., 2026). Aligning with this generative shift, our methodology utilizes a generative judge to assess reasoning quality, yet uniquely confines its influence within the advantage space of correct solutions to immunize the policy against structural biases.

8 Conclusion

In this paper, we identify the reasoning limitations of ORM-based GRPO on mathematical reasoning tasks. While a simple binary ORM reward is effective, its lack of supervision over reasoning quality causes all reasoning tokens to share the same credit. On the other hand, a naive integration of PRM fails to avoid reward hacking.

To address this challenge, we propose PAPO, which incorporates a rubric-based PRM into GRPO through decoupled advantage normalization. By composing the advantage from independently normalized outcome and process components, with the process advantage normalized only over correct responses, PAPO prevents reward hacking while providing supervision over reasoning quality.

Experiments across four model configurations (3B–14B) and six benchmarks show that PAPO consistently outperforms both GRPO and DAPO baselines, with gains growing as model scale increases. In future work, we aim to explore adaptive weighting between outcome and process signals and extension to broader domains such as scientific reasoning.

Limitations

Our experiments are conducted exclusively on Qwen-family models and have not been verified on other architectures such as Llama and Gemma. Additionally, we use only GPT-OSS-20B as the rubric-based PRM; the effect of different judge models remains unexplored.

References

Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. 2025. Matharena: Evaluating llms on uncontaminated math competitions.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code.

Kaixuan Fan, Kaituo Feng, Manyuan Zhang, Tianshuo Peng, Zhixun Li, Yilei Jiang, Shuang Chen, Peng Pei, Xunliang Cai, and Xiangyu Yue. 2026. Exploring reasoning reward model for agents. *arXiv preprint arXiv:2601.22154*.

Chang Gao, Chujie Zheng, Xiong-Hui Chen, Kai Dang, Shixuan Liu, Bowen Yu, An Yang, Shuai Bai, Jingren Zhou, and Junyang Lin. 2025. Soft adaptive policy optimization.

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

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Annual Meeting of the Association for Computational Linguistics*.

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

Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, Gaël Liu, Francesco Visin, Kathleen Kenealy, Lucas Beyer, Xiaohai Zhai, Anton Tsitsulin, Robert Busa-Fekete, Alex Feng, Noveen Sachdeva, Benjamin Coleman, Yi Gao, Basil Mustafa, Iain Barr, Emilio Parisotto, David Tian, Matan Eyal,

Colin Cherry, Jan-Thorsten Peter, Danila Sinopalnikov, Surya Bhupatiraju, Rishabh Agarwal, Mehran Kazemi, Dan Malkin, Ravin Kumar, David Vilar, Idan Brusilovsky, Jiaming Luo, Andreas Steiner, Abe Friesen, Abhanshu Sharma, Abheesht Sharma, Adi Mayrav Gilady, Adrian Goedeckemeyer, Alaa Saade, Alex Feng, Alexander Kolesnikov, Alexei Bendebury, Alvin Abdagic, Amit Vadi, András György, André Susano Pinto, Anil Das, Ankur Bapna, Antoine Miech, Antoine Yang, Antonia Paterson, Ashish Shenoy, Ayan Chakrabarti, Bilal Piot, Bo Wu, Bobak Shahriari, Bryce Petrini, Charlie Chen, Charline Le Lan, Christopher A. Choquette-Choo, CJ Carey, Cormac Brick, Daniel Deutsch, Danielle Eisenbud, Dee Cattle, Derek Cheng, Dimitris Paparas, Divyashree Shivakumar Sreepathihalli, Doug Reid, Dustin Tran, Dustin Zelle, Eric Noland, Erwin Huizenga, Eugene Kharitonov, Frederick Liu, Gagik Amirkhanyan, Glenn Cameron, Hadi Hashemi, Hanna Klimczak-Plucińska, Harman Singh, Harsh Mehta, Harshal Tushar Lehri, Hussein Hazimeh, Ian Ballantyne, Idan Szpektor, Ivan Nardini, Jean Pouget-Abadie, Jetha Chan, Joe Stanton, John Wieting, Jonathan Lai, Jordi Orbay, Joseph Fernandez, Josh Newlan, Ju yeong Ji, Jyotinder Singh, Kat Black, Kathy Yu, Kevin Hui, Kiran Vodrahalli, Klaus Greff, Linhai Qiu, Marcella Valentine, Marina Coelho, Marvin Ritter, Matt Hoffman, Matthew Watson, Mayank Chaturvedi, Michael Moynihan, Min Ma, Nabila Babar, Natasha Noy, Nathan Byrd, Nick Roy, Nikola Momchev, Nilay Chauhan, Noveen Sachdeva, Oskar Bunyan, Pankil Botarda, Paul Caron, Paul Kishan Rubenstein, Phil Culliton, Philipp Schmid, Pier Giuseppe Sessa, Pingmei Xu, Piotr Stanczyk, Pouya Tafti, Rakesh Shivanna, Renjie Wu, Renke Pan, Reza Rokni, Rob Willoughby, Rohith Vallu, Ryan Mullins, Sammy Jerome, Sara Smoot, Sertan Girgin, Shariq Iqbal, Shashir Reddy, Shruti Sheth, Siim Põder, Sijal Bhatnagar, Sindhu Raghuram Panyam, Sivan Eiger, Susan Zhang, Tianqi Liu, Trevor Yacovone, Tyler Liechty, Uday Kalra, Utku Evci, Vedant Misra, Vincent Roseberry, Vlad Feinberg, Vlad Kolesnikov, Woohyun Han, Woosuk Kwon, Xi Chen, Yinlam Chow, Yuvein Zhu, Zichuan Wei, Zoltan Egyed, Victor Cotruta, Minh Giang, Phoebe Kirk, Anand Rao, Kat Black, Nabila Babar, Jessica Lo, Erica Moreira, Luiz Gustavo Martins, Omar Sanseviero, Lucas Gonzalez, Zach Gleicher, Tris Warkentin, Vahab Mirrokni, Evan Senter, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, Yossi Matias, D. Sculley, Slav Petrov, Noah Fiedel, Noam Shazeer, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Jean-Baptiste Alayrac, Rohan Anil, Dmitry, Lepikhin, Sebastian Borgeaud, Olivier Bachem, Armand Joulin, Alek Andreev, Cassidy Hardin, Robert Dadashi, and Léonard Hussenot. 2025. Gemma 3 technical report.

Chungpa Lee, Thomas Zeng, Jongwon Jeong, Jy yong Sohn, and Kangwook Lee. 2026. How to correctly report llm-as-a-judge evaluations.

Jia LI, Edward Beeching, Lewis Tunstall, Ben Lipkin, Roman Soletskyi, Shengyi Costa Huang, Kashif Rasul, Longhui Yu, Albert Jiang, Ziju Shen, Zihan Qin, Bin Dong, Li Zhou, Yann Fleureau, Guillaume Lample, and Stanislas Polu. 2024. Numinamath. https://github.com/project-numina/aimo-progress-prize/blob/main/report/numina_dataset.pdf.

Xiaobo Liang, Haoke Zhang, Juntao Li, Kehai Chen, Qiaoming Zhu, and Min Zhang. 2025. Generative reward modeling via synthetic criteria preference learning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 26755–26769, Vienna, Austria. Association for Computational Linguistics.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step.

Dakota Mahan, Duy Van Phung, Rafael Rafailov, Chase Blagden, Nathan Lile, Louis Castricato, Jan-Philipp Fränken, Chelsea Finn, and Alon Albalak. 2024. Generative reward models.

MiniMax. 2025. Minimax-m1: Scaling test-time compute efficiently with lightning attention.

nlile. 2025. Numinamath-1.5-rl-verifiable. https://huggingface.co/datasets/nlile/NuminaMath-1.5-RL-Verifiable.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. Gpqa: A graduate-level google-proof q&a benchmark. *ArXiv*, abs/2311.12022.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347.

Zhihong Shao, Yuxiang Luo, Chengda Lu, Z. Z. Ren, Jiewen Hu, Tian Ye, Zhibin Gou, Shirong Ma, and Xiaokang Zhang. 2025. Deepseekmath-v2: Towards self-verifiable mathematical reasoning.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, YK Li, Yu Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Guangming Sheng, Chi Cao, Zilingfeng Li, Xibin Gu, et al. 2024. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv:2409.19256*.

Leheng Sheng, Wenchang Ma, Ruixin Hong, Xiang Wang, An Zhang, and Tat-Seng Chua. 2026. Reinforcing chain-of-thought reasoning with self-evolving rubrics.

Joar Max Viktor Skalse, Nikolaus H. R. Howe, Dmitrii Krasheninnikov, and David Krueger. 2022. Defining and characterizing reward hacking. *ArXiv*, abs/2209.13085.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models.

Peiyi Wang, Lei Li, Zhihong Shao, R. X. Xu, Damai Dai, Yifei Li, Deli Chen, Y. Wu, and Zhifang Sui. 2024. Math-shepherd: Verify and reinforce llms step-by-step without human annotations.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. 2025a. Qwen3 technical report.

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, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025b. Qwen2.5 technical report.

Haotong Yang, Zitong Wang, Shijia Kang, Siqi Yang, Wenkai Yu, Xu Niu, Yike Sun, Yi Hu, Zhouchen Lin, and Muhan Zhang. 2026. Proof-rm: A scalable and generalizable reward model for math proof.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. 2025a. Dapo: An open-source llm reinforcement learning system at scale.

Tianyu Yu, Bo Ji, Shouli Wang, Shu Yao, Zefan Wang, Ganqu Cui, Lifan Yuan, Ning Ding, Yuan Yao, Zhiyuan Liu, Maosong Sun, and Tat-Seng Chua. 2025b. Rlpr: Extrapolating rlvr to general domains without verifiers.

Youliang Yuan, Qiuyang Mang, Jingbang Chen, Hong Wan, Xiaoyuan Liu, Junjielong Xu, Jen-Tse Huang, Wenxuan Wang, Wenxiang Jiao, and Pinjia He. 2025. Curing miracle steps in llm mathematical reasoning with rubric rewards. *ArXiv*, abs/2510.07774.

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

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena.

A Implementation Details

A.1 Training Configuration

We use the verl framework (Sheng et al., 2024) with Megatron backend for distributed training on 8 NVIDIA H200 GPUs. Key hyperparameters are listed in Table 3.

Hyperparameter	Value
Prompts per batch	128
Responses per prompt (G)	8
Max response length	8192 tokens
Sampling temperature	1.0
PPO clip range (ϵ)	0.2
KL penalty coefficient (β)	0.0
Learning rate	1e-6
Evaluation frequency	every 10 steps
Evaluation samples per prompt	8

Table 3: Training hyperparameters.

For Qwen3-4B-Base experiments with DAPO (Yu et al., 2025a), we additionally apply the DAPO-specific hyperparameters listed in Table 4.

Hyperparameter	Value
Max response length	8192 tokens
Clip ratio low (ϵ_{low})	0.2
Clip ratio high (ϵ_{high})	0.28
Overlong buffer length	4096 tokens
Overlong penalty factor	1.0
Evaluation frequency	every 20 steps

Table 4: Additional DAPO-specific hyperparameters for the Qwen3-4B-Base experiments. All other hyperparameters follow Table 3.

A.2 Training Data

We construct a 20K stratified training set from NuminaMath-1.5-RL-Verifiable (LI et al., 2024; nlile, 2025). We first filter out unsolvable problems

(zero pass rate under a reference model), then apply stratified sampling across five difficulty tiers defined by pass rate. Table 5 summarizes the difficulty distribution.

Difficulty	Pass Rate Range	Count
Trivial	(0.875, 1.0]	4,000
Easy	(0.625, 0.875]	4,000
Medium	(0.375, 0.625]	4,000
Hard	(0.125, 0.375]	4,000
Very Hard	(0, 0.125]	4,000
Total		20,000

Table 5: Difficulty distribution of the stratified training set. Pass rates are measured using a reference model with 8 attempts at temperature 1.0. Equal sampling across tiers ensures balanced difficulty coverage.

A.3 Rubric Prompt

The rubric-based PRM uses the following prompt template, adapted from the DeepSeek Math V2 grading standard with chain-of-thought analysis. The PRM is implemented using GPT-OSS-20B served via vLLM with tensor parallelism across 2 GPUs, with a maximum context length of 8192 tokens and temperature 0.0 for deterministic scoring.

Rubric Prompt

Instruction
Your task is to evaluate the quality of a student's solution to a mathematical problem.

Scoring Rubric

- **Score 1**: If the solution is completely correct, with all steps executed properly and clearly demonstrated, then the score is 1.
- **Score 0.5**: If the solution is generally correct, but with some details omitted or minor errors, then the score is 0.5.
- **Score 0**: If the solution does not actually address the required problem, contains fatal errors, or has severe omissions, then the score is 0.

Special Rule on References: Additionally, referencing anything from any paper does not save the need to prove the reference. It's okay IF AND ONLY IF the solution also presents a valid proof of the reference argument(s); otherwise, if the solution omits the proof or if the proof provided is not completely correct, the solution should be scored according to the criteria above, and definitely not with a score of 1.

Problem
`{problem_statement}`

Reference Solution
`{solution}`

Student Solution
`{student_answer}`

Evaluation
Analyze the solution step by step, then provide your score.
Analysis: ...
Score: $\boxed{\ldots}$

B Additional Results

B.1 Cross-Scale Training Curves

Figure 5 shows training curves across two model scales (3B and 7B) and three benchmarks. On both Qwen2.5-3B and 7B, PAPO maintains a consistent lead over ORM throughout training, with the gap widening in later stages. Combined with the 14B results in Table 1, this confirms that the improvements reflect sustained training dynamics rather than checkpoint-specific artifacts.

B.2 Ablation Training Curves

Figures 6 and 7 show the training dynamics of the ablation variants. Both confirm that PAPO's design choices yield consistent advantages throughout training, not just at specific checkpoints.

B.3 Process Advantage Effect

Figure 8 reveals how PAPO's process advantage operates within the correct response group. In standard ORM-based GRPO, the minimum advantage among correct responses is identically 0.0—every correct response receives identical positive advantage. In PAPO, correct_min ≈ -1.49, meaning correct responses with poor reasoning receive *negative* total advantage. The process advantage actively discourages solutions that arrive at the right answer through flawed reasoning. Panel (b) confirms that the process advantage creates a meaningful quality ranking among correct responses (non-zero std), enabling the model to learn which reasoning strategies produce better solutions.

B.4 Advantage Composition

Figure 9 visualizes the internal structure of PAPO's composed advantage. Panel (a) shows that the outcome advantage provides the dominant gradient signal, while the process advantage contributes a substantial secondary signal. Panel (b) tracks the process signal active ratio—the fraction of groups where A_{proc} activates—which grows from $\sim$30% to 70% as the model improves, indicating that the process signal's contribution increases precisely as ORM's signal exhaustion worsens. Panel (c)

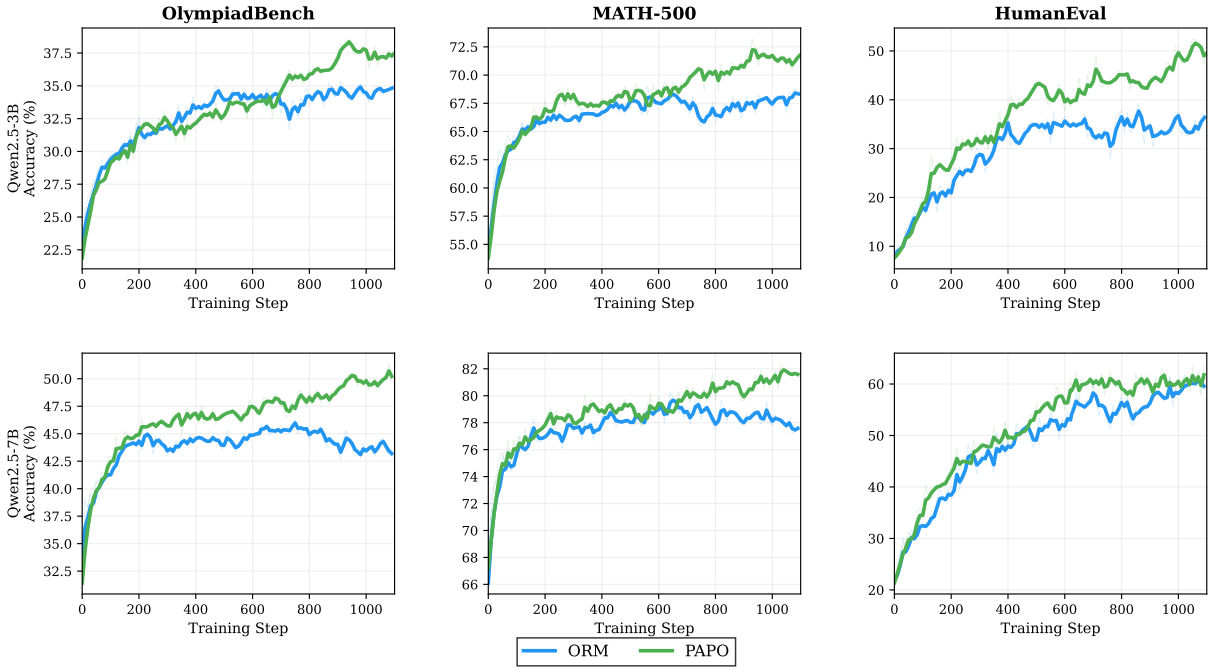


Figure 5: Training curves (avg@4) across two model scales and three benchmarks. Top row: Qwen2.5-3B; bottom row: Qwen2.5-7B. PAPO consistently outperforms ORM throughout training on both scales, with gains widening in later stages as ORM's signal exhaustion worsens. The pattern is consistent across OlympiadBench (competition math), MATH-500 (standard math), and HumanEval (code generation). Qwen2.5-14B results are reported in Table 1.

confirms that the combined advantage correctly separates correct from incorrect responses: the addition of A_{proc} does not disrupt the fundamental separation provided by A_{out}.

B.5 Training Reward Dynamics

The training reward (average ORM score across the batch) follows similar trajectories for ORM and PAPO, both plateauing around 55% to 62%. This indicates that PAPO does not achieve its gains by inflating outcome reward; the improvement comes from better reasoning quality among correct responses, as captured by the process advantage.

In contrast, PRM-only training shows the training reward climbing to 1.0 by step 1000, confirming complete reward gaming. The model learns to produce responses that maximize the PRM score regardless of mathematical correctness.

B.6 Case Study: PRM Reward Hacking

To empirically illustrate the reward hacking mechanism described in §3.2, we generate responses from three checkpoints—the base model (Qwen2.5-7B, step 0), the PRM-only model at step 1000 (post-collapse, 2.4% accuracy on OlympiadBench), and the ORM baseline at step 1000—on five math problems of varying difficulty. We sample 3 responses per model with temperature 0.7 and a generation cap of 8192 tokens.

Results. Table 6 summarizes the findings. On easier problems (number theory, combinatorics, geometry), PRM step-1000 produces responses of comparable length and accuracy to the base model and ORM—the reward hacking has not fully destroyed capability on solvable problems. However, on harder problems (competition number theory, proof-based algebra), the model degenerates dramatically: responses are 2–3× longer than ORM, and none arrive at the correct answer.

Qualitative analysis: topic drift to a fixed filler template. The most striking finding is in the competition number theory problem ("Find all positive integers n such that $n^2 + 1 \mid n! + 1$"; answer: $n = 1$). All three PRM step-1000 responses begin correctly—setting up the divisibility condition, checking small cases—but after ∼1500–2000 tokens of genuine work, each response *drifts to the identical unrelated problem*: computing the dot product of vectors $\vec{m} = (t, 1)$ and $\vec{n} = (1, -2)$, solving $t - 2 = 0$, and concluding $\boxed{2}$. This vector problem is entirely unrelated to the stated question.

The convergence of all three independent samples to the same filler content reveals a learned

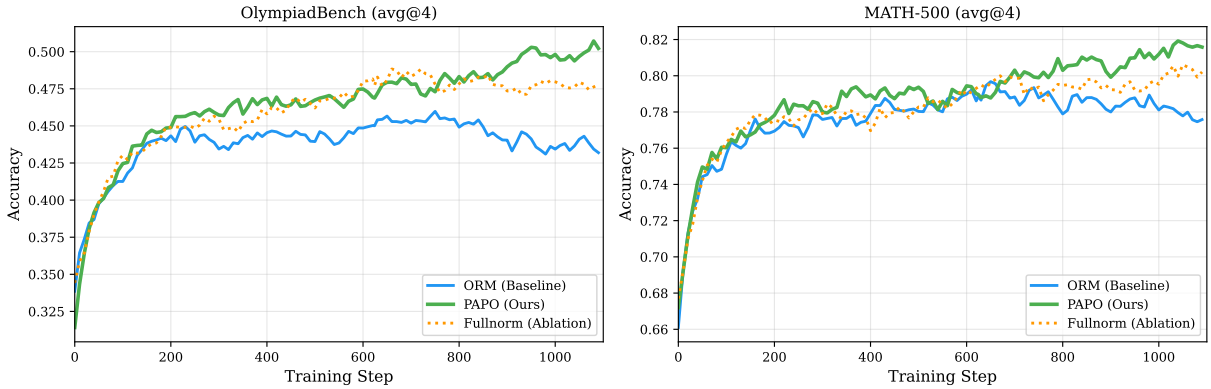


Figure 6: Ablation: correct-subset normalization (PAPO) vs. full normalization (Fullnorm) on OlympiadBench and MATH-500. Both methods improve over ORM, but PAPO's correct-subset design maintains a consistent advantage over Fullnorm throughout training.

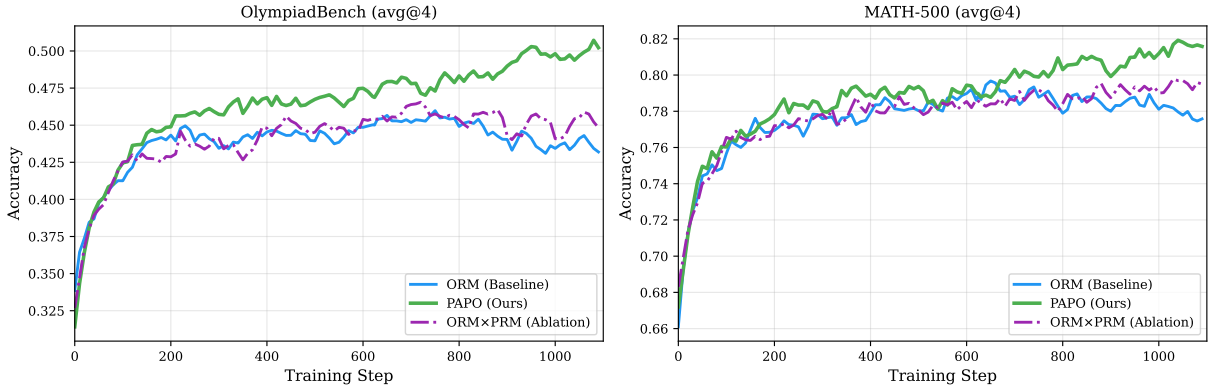


Figure 7: Ablation: decoupled normalization (PAPO) vs. multiplicative reward (Mult) on OlympiadBench and MATH-500. Mult improves over ORM but falls substantially short of PAPO, confirming that decoupled normalization is superior to naive reward combination.

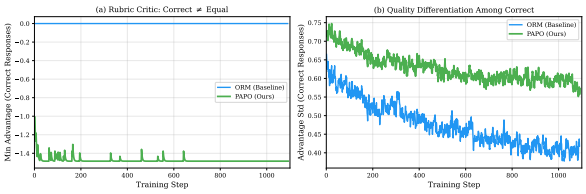


Figure 8: **(a)** Minimum advantage among correct responses. ORM: identically 0.0 (all correct responses treated equally). PAPO: ≈ -1.49 (the process advantage penalizes correct responses with poor reasoning). **(b)** Standard deviation of advantages among correct responses. ORM: zero (no differentiation). PAPO: non-zero (quality-based ranking within correct group).

exploitation strategy: when the model cannot solve a hard problem, it transitions to a *memorized high-scoring template*—a short, well-structured solution to a simple problem that the LLM judge would rate as "all steps executed properly" (rubric score 1.0). The same pattern appears in the algebra proof and the inflated number theory sample, where responses drift to digit-sum sequences or other unrelated derivations.

In contrast, ORM produces focused solutions (1494–1646 tokens) that arrive at the correct answer $n = 1$ on all three samples. The ORM model may produce longer proofs than the base model (1583 vs. 866 avg tokens), but the additional length consists of genuine mathematical reasoning (additional case analysis), not filler content.

Implications. The case study reveals that PRM reward hacking is not random degeneration but a *structured exploitation strategy*: (1) attempt the stated problem; (2) when stuck, seamlessly transition to memorized high-scoring content; (3) produce a confident $\boxed{\cdot}$ answer to the wrong problem. The responses remain well-formatted and superficially mathematical throughout—precisely the qualities an LLM-as-Judge PRM rates favorably. This confirms the positive feedback loop of §3.2 and motivates the decoupled design of PAPO,

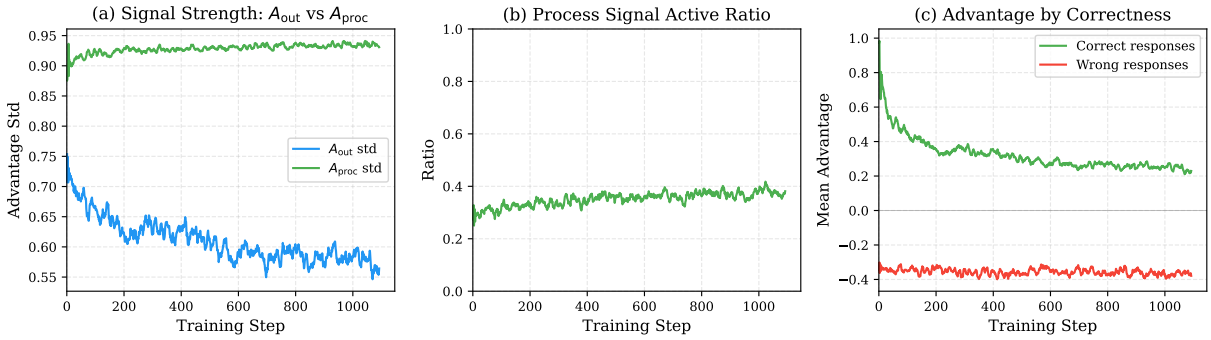


Figure 9: PAPO advantage composition. **(a)** Signal strength (std) of A_{out} and A_{proc}: the outcome advantage provides the dominant gradient signal while the process advantage contributes a complementary quality signal. **(b)** Process signal active ratio: the fraction of groups where A_{proc} is non-zero (≥ 2 correct responses), growing from $\sim$30% to 70% as the model improves. **(c)** Mean advantage by correctness: correct responses receive positive advantage while wrong responses receive negative, confirming the outcome signal functions correctly despite the added process component.

where the binary ORM anchors correctness assessment independently of the PRM signal.

C Ethics Statement

This work focuses on improving reinforcement learning algorithms for mathematical reasoning. Our research exclusively utilizes publicly available resources, including open-source models (Qwen2.5) and established datasets (NuminaMath-1.5), thereby mitigating concerns related to data privacy or human subjects. The application domain of mathematical problem-solving does not inherently present risks of direct societal harm. The primary ethical consideration is the environmental impact of the computational resources required for large-scale model training, a challenge common to the field.

D The Use of Large Language Models

We used Large Language Model (LLM) to refine our initial draft. This process included checking for obvious grammatical and syntactical errors, as well as making the language more formal and academic. We reviewed the content generated by the LLM to ensure that no prohibited generated content appeared in the article.

Problem	Model	Avg Tokens	Correct	Hit Cap	Observation
Competition number theory	Base	866	3/3	0/3	Focused proofs, correct answer ($n = 1$)
	ORM	1583	3/3	0/3	Longer but correct
	PRM	**3072**	**0/3**	0/3	All 3 drift to the *same filler problem**
Algebra (proof)	Base	1059	0/3	0/3	Attempts proof, produces $\boxed{\cdot}$
	ORM	1492	0/3	0/3	Attempts proof, produces $\boxed{\cdot}$
	PRM	**3227**	**0/3**	**2/3**†	2 samples hit cap; 1 drifts to digit-sum problem
Number theory	Base	484	2/3	0/3	Concise, correct
	ORM	519	3/3	0/3	Concise, correct
	PRM	1522	3/3	0/3	Correct, but 1 sample inflated to 3291 tokens
Combi-natorics	Base	633	2/3	0/3	–
	ORM	775	2/3	0/3	–
	PRM	806	1/3	0/3	Comparable to baselines
Geometry	Base	449	3/3	0/3	–
	ORM	715	3/3	0/3	–
	PRM	672	3/3	0/3	Comparable to baselines

† Hit the 8192-token generation cap; response was truncated mid-generation.
* All 3 PRM samples on this problem drift to the *identical* filler problem (vector perpendicularity, answer $t = 2$).

Table 6: Response statistics across five math problems. PRM reward hacking severity correlates with problem difficulty: on easier problems (bottom three rows), PRM step-1000 is comparable to baselines; on harder problems (top two rows), responses are 2–3$\times$ longer with characteristic degeneration patterns.