PRPO: Aligning Process Reward with Outcome Reward in Policy Optimization

Anonymous ACL submission

Abstract

Policy optimization for large language models often suffers from sparse reward signals in multi-step reasoning tasks. Critic-free methods like GRPO assign a single normalized outcome reward to all tokens, providing limited guidance for intermediate reasoning . While Process Reward Models (PRMs) offer dense feedback, they risk premature collapse when used alone, as early low-reward tokens can drive policies toward truncated outputs. We introduce Process Relative Policy Optimization (PRPO), which combines outcome reliability with process-level guidance in a critic-free framework. PRPO segments reasoning sequences based on semantic clues, normalizes PRM scores into token-level advantages, and aligns their distribution with outcome advantages through location-parameter shift. On MATH500, PRPO improves Qwen2.5-Math-1.5B accuracy from 61.2% to 64.4% over GRPO using only eight rollouts and no value network, demonstrating efficient fine-grained credit assignment within critic-free optimization.

1 Introduction

Reinforcement learning (RL) has become a central paradigm for aligning large language models (LLMs) with human objectives, particularly in tasks that require complex reasoning, long-horizon planning, and structured decision-making. Early RL-based alignment methods, such as reinforcement learning from human feedback (RLHF), primarily rely on sequence-level rewards that evaluate only the final generated output. While effective for short-form generation and surface-level preferences, this outcome-centric formulation exhibits fundamental limitations when applied to multi-step reasoning tasks, where correct solutions depend on a sequence of interdependent intermediate decisions rather than the final answer alone.

In long reasoning scenarios—such as mathematical problem solving, program synthesis, and multi-hop question answering—the reward signal is typically sparse, delayed, and weakly informative. A single scalar reward assigned at the end of generation provides little guidance for identifying which intermediate reasoning steps were correct, redundant, or erroneous. This exacerbates the credit assignment problem over long reasoning chains and leads to high-variance policy gradients, unstable optimization, and inefficient use of training samples. Moreover, purely outcome-based rewards fail to distinguish between reasoning trajectories that arrive at the same answer via qualitatively different processes, limiting the model's ability to internalize robust and generalizable reasoning strategies.

These challenges are further amplified in critic-free policy optimization methods, which deliberately avoid training value or critic networks in order to improve scalability and reduce computational overhead. Although such methods—exemplified by DPO-style and PPO-style algorithms—offer substantial efficiency advantages, they are inherently more sensitive to reward sparsity and noise, as they lack explicit mechanisms for temporal credit assignment. As a result, designing effective learning signals for long-horizon reasoning under critic-free constraints remains an open and critical problem.

Motivated by these limitations, recent research has progressively moved beyond sequence-level supervision toward finer-grained learning signals that explicitly model the internal reasoning process. This shift has given rise to a growing body of work on process-level and token-level preference optimization, which aims to provide denser and more informative supervision for policy learning.

In preference optimization, recent methods have progressively shifted from sequence-level to finer-grained supervision. Process-DPO (Lai et al., 2024) optimizes the first erroneous reasoning process,

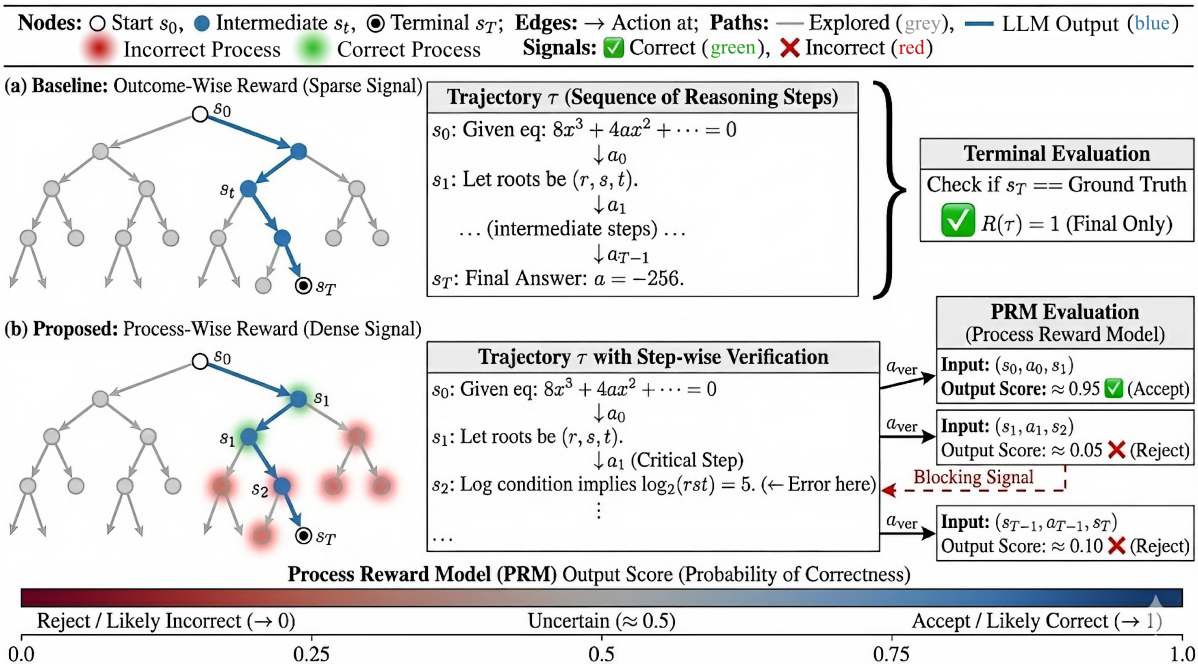


Figure 1: Comparison between outcome-level and process-level reward in RLHF.

while Full-process-DPO (Xu et al., 2025) leverages self-supervised reward models to score all processes and dynamically weight gradients. TGDPO (Zhu et al., 2025) introduces token-level reward guidance directly into the DPO loss, enabling dense, token-wise policy updates. These works establish a paradigm shift from sparse outcome-based rewards to dense, process-oriented supervision.

Parallel developments address sparse rewards with token-level and process-wise optimization techniques in PPO-style algorithms. TEPO (Lin et al., 2025) introduces a Markov likelihood framework to link group-level rewards with token-level aggregation, reducing gradient variance in critic-free settings like GRPO (Shao et al., 2024). RiskPO (Ren et al., 2025) approaches the problem from a distributional perspective, using risk measures like Mixed Value-at-Risk to emphasize low-reward instances and prevent entropy collapse.

Despite these advances, limited research integrates Process Reward Models (PRMs) into critic-free policy optimization, and most current work didn't make use of PRM's process-level characteristics and still provide only one sparse reward signal for each output sequence. PSGPO (Dai et al., 2025) explored PRM guidance within PPO, showing improvements in code generation. Other critic-free works reshape outcome rewards with PRM signals to distinguish sequences that share identical outcomes (Zou et al., 2025; Ren, 2025), but still deliver a single reward per sequence, leaving process-level supervision underutilized. This gap motivates PRPO (Process Relative Policy Optimization), which intends to make use of the full signal from dense PRM feedback with sparse outcome rewards in a critic-free framework while retaining computational efficiency.

It is worth noting that a current work tried to realize process-level credit assignment in critic-free framework, which is PURE Algorithm (Cheng et al., 2025), which is a min-form credit assignment algorithm for process reward model. However, its basic idea is to choose the min value of the process reward model output, which surely can resolve possible collapse of process reward only training (which we will mention later), but will lose much information of the different process rewards along the whole output sequence, especially for the higher ones which are usually abandoned.

PRPO addresses this critical gap by introducing a novel framework that integrates full dense PRM feedback into critic-free optimization through distribution alignment, enabling fine-grained credit assignment while preserving the computational efficiency that makes critic-free methods attractive. This innovation is particularly important because it demonstrates that stable, dense supervision can be achieved without abandoning the resource-efficient critic-free paradigm, opening new possibilities for scalable policy optimization in complex reasoning

domains.

Our contributions are threefold: (1) a framework that integrates process supervision into critic-free policy optimization; (2) a distribution alignment technique ensuring global consistency of token-level advantages; and (3) empirical evidence that PRPO significantly improves reasoning accuracy while maintaining efficiency comparable to GRPO.

2 Related Work

2.1 Critic-Free Policy Optimization

GRPO (Shao et al., 2024) establishes a critic-free framework by computing group-relative advantages: $\hat{A}_{\text{group}}(\tau_i) = \frac{R(\tau_i) - \text{mean}(R(\tau_i))}{\sigma(R(\tau_i)) + \epsilon}$. Building on this, ETPO (Wen et al., 2024) decomposes credit assignment to token-level with per-token soft Bellman updates, while TEPO (Lin et al., 2025) connects group rewards to tokens via Markov likelihood. RiskPO (Ren et al., 2025) applies Value-at-Risk theory to reshape advantages for denser signals. Test-time adaptations try to resolve OOD (Out of Distribution) Problem, featured as ETTRL (Liu et al., 2025a), which uses entropy-based mechanisms for diverse candidate generation to improve efficiency in original TTRL (Zuo et al., 2025), and TGRPO (Chen et al., 2025), which reformats outcome rewards into process-wise signals for embodied AI.

2.2 Process Reward Models

Process reward models provide step-level supervision for multi-step reasoning. Early work like PRM800K (Song et al., 2025) relied on costly human annotation. Automated approaches subsequently emerged to reduce this cost: Math-Shepherd (Wang et al., 2024) uses self-consistency sampling and symbolic verification, while Omega PRM (Luo et al., 2024) leverages MCTS-guided search to explore reasoning trajectories. Recent methods like Qwen2.5-Math-PRM (Zhang et al., 2025) employ LLM-based judgments to generate soft labels, and EDU-PRM (Cao et al., 2025) uses entropy as fork points in MCTS searches. These advances enable scalable process supervision despite potential label noise.

3 PRPO

To leverage process-level rewards from PRMs in critic-free policy optimization, PRPO combines semantic-based segmentation with a distribution alignment mechanism that fuses outcome and process advantages. The overall framework is shown in Figure 2.

3.1 Semantic-Based Sequence Segmentation

Process segmentation for PRM evaluation has often relied on explicit markers such as "Step" (Zhang et al., 2025). This heuristic generalizes poorly when a model is not trained to emit such markers at every reasoning stage. To obtain a more robust and model-agnostic segmentation, we use token-level entropy to split the output sequence into segments and evaluate each segment with a PRM.

Recent studies demonstrate that token-level entropy is effective for segmenting LLM outputs (Liu et al., 2025a; Cao et al., 2025; Wang et al., 2025). Wang et al. (2025) find that entropy spikes frequently align with logical connectors (e.g., "wait", "however", "nevertheless") that bridge deductive sentences. ETTRL (Liu et al., 2025a) similarly uses entropy spikes as fork points in rollout search, achieving better performance than traditional rollout within the TTRL framework. These results suggest that entropy peaks offer reliable semantic boundaries.

For LLM outputs, the entropy of each token is generally calculated as

$$E_k = -\sum_{i=1}^{n} p(x_{k,i}) \log(p(x_{k,i}))$$

Leveraging this property, we pick the top k spike points—each at least m tokens apart—as split positions and evaluate the process-level reward for every resulting segment with a PRM.

Although it is straightforward to think entropy can decrease during training, which may make this segmentation method less reliable. However, based on empirical experiments done by (Wang et al., 2025), the tokens with higher entropy generally increase their entropy during training, while other tokens stay at a relatively stable entropy level. This means that the entropy spike points become more significant during training, thus justifying the reliability of this segmentation method. The detailed step about how this semantic-based segmentation is implemented is in Appendix B.

3.2 Aligning Outcome-level and Process-level Reward

Consider a sampled trajectory τ of length T segmented into M logical units $s_i = [t_{i-1}, t_i)$ with

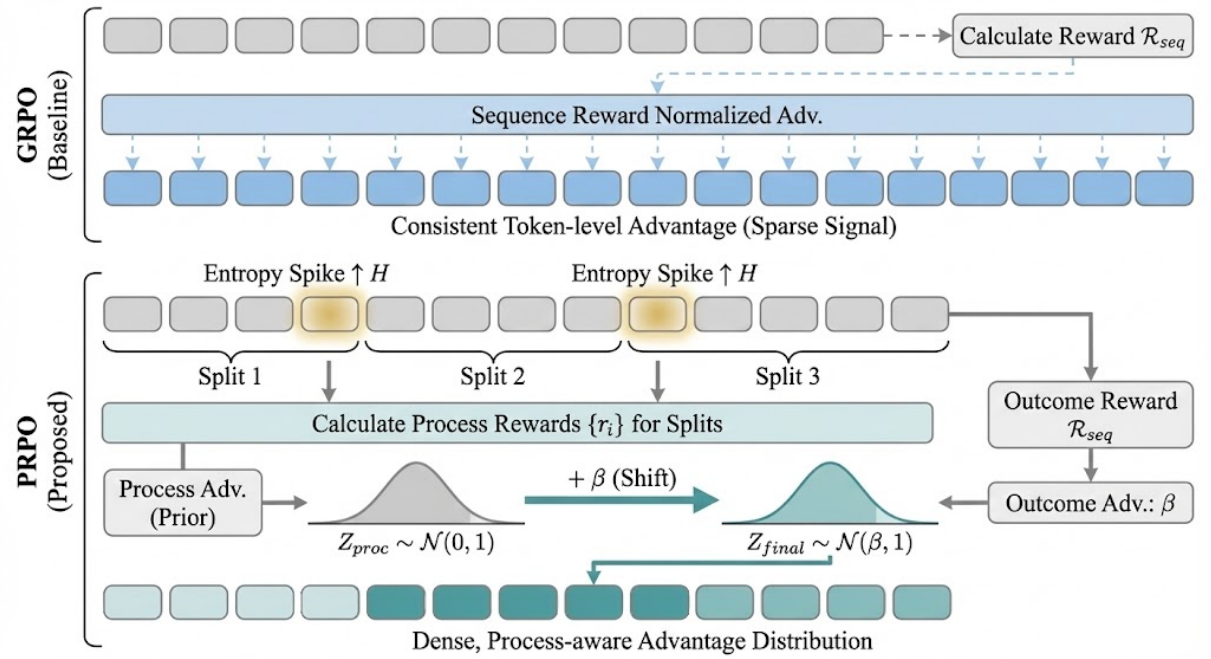


Figure 2: Comparison between GRPO and PRPO.

PRM rewards r^i_{process}. Tokens within segment s_i share the same PRM score. The segment-level process reward is expanded to tokens by setting, for any $t \in s_i$,

$$z_{\text{process},t}(\tau) = \frac{r^i_{\text{process}} - \mu_{\text{prior_process}}}{\sigma_{\text{prior_process}}},$$

giving the token vector $\mathbf{z}_{\text{process}}(\tau) \in \mathbb{R}^T$, where $\mu_{\text{prior_process}}$ and $\sigma_{\text{prior_process}}$ are the mean and standard deviation of the prior distribution of the PRM output. In practice, as the PRM we used (Qwen2.5-Math-PRM-7B) has its output value normalized within [0,1], we use the expected PRM output (0.5) as the mean and the standard deviation of a uniform distribution (0.289) for normalization.

The sparse outcome reward $R_{\text{outcome}}(\tau)$ is normalized using prior statistics to obtain the beta coefficient. As we need to fuse the outcome advantage with process advantage, we follow Dr. GRPO (Liu et al., 2025b) and only use the mean value for normalization to avoid the scaling bias of the standard deviation which may cause problem in advantage aligning:

$$\beta(\tau) = R_{\text{outcome}}(\tau) - \mu_{\text{rollout}}$$

where μ_{rollout} is the mean value of the outcome reward in each rollout group, which is defined as:

$$\mu_{\text{rollout}} = \frac{1}{N}\sum_{j=1}^{N} R_{\text{outcome}}(\tau^{(j)})$$

and broadcast this scalar to every token position of τ.

Premature Collapse in Process-Only Reward Let π_θ be an autoregressive policy and $x_{0:T}$ a sampled sequence. Its log-likelihood decomposes as

$$\log p_\theta(x_{0:T}) = \sum_{t=0}^{T} \log \pi_\theta(x_t \mid x_{<t}).$$

The process reward model outputs process rewards r_t with standardized advantage $A_t := r_t - \mu_{\text{process}}$. Under process-only optimization,

$$J_{\text{proc}}(\theta) = \mathbb{E}_{\pi_\theta}\Big[\sum_{t=0}^{T} A_t \, \log \pi_\theta(x_t \mid x_{<t})\Big], \tag{1}$$

$$\nabla_\theta J_{\text{proc}}(\theta) = \mathbb{E}_{\pi_\theta}\Big[\sum_{t=0}^{T} A_t \, \nabla_\theta \log \pi_\theta(x_t \mid x_{<t})\Big]. \tag{2}$$

Based on our observation of the process-level and outcome-level advantage during the RL training, we proposed the following hypothesis that usually causes the collapse of model trained with RL algorithm using only process-level signal:

Hypothesis 1 (Empirical Premature Collapse Condition during RL training with Process-Only Reward)**.** *For a generative sequence with token-level*

advantages $\{A_t\}_{t=0}^{T}$ where $A_t = r_t - \mu_{process}$ (process reward r_t relative to mean $\mu_{process}$), if there exists a position $t^\star \in [1, T]$ such that:

1. *The parameters a and b are sampled from the same sequence: $a = \left|\frac{1}{t^\star}\sum_{i=0}^{t^\star-1} A_i\right|$ (absolute mean advantage over prefix $[0, t^\star)$) and $b = A_{t^\star}$ (advantage at position $t^\star$), where a reflects early tokens with process rewards below the mean (negative advantage) and b reflects a token with process reward above the mean (positive advantage).*

2. *Early tokens have negative average advantage: $\frac{1}{t^\star}\sum_{i=0}^{t^\star-1} A_i = -a < 0$ with $a > 0$, indicating process rewards below the mean.*

3. *Token at $t^\star$ has positive advantage: $A_{t^\star} = b > 0$, indicating process reward above the mean.*

4. *The cumulative negative magnitude exceeds the positive benefit: $a \cdot t^\star > b$.*

Then the conflicting gradient signals from early negative advantages ($-a$) and later positive advantage (b) within the same sequence cause $p_\theta(x_{0:t^\star})$ to decrease, leading to premature collapse.

The mathematical reason for this hypothesis is shown in Appendix A

Interpretation Negative advantages hit every prefix term multiplicatively, while the single positive advantage at $t^\star$ cannot compensate for the accumulated decay. Heavy-tailed or outlier negative rewards (large a) accelerate the collapse.

Based on this hypothesis, it can be witnessed that the root problem of the collapse of model trained with process-only reward is the fluctuation of process rewards, which can cause conflict of advantage signal. PURE (Cheng et al., 2025) resolves this problem by simply choosing the min value of the process reward model output, which surely can resolve possible collapse of process reward only training, but will lose much information of the different process rewards along the whole output sequence, especially for the higher ones which are usually abandoned.

Practical observation We actually did an experiment on Qwen2.5-Math-7B model trained with process advantages alone, and found that it empirically shortened responses and eventually produced meaningless short strings, matching the hypothesis.

Implication Even perfectly estimated process rewards are unsafe when used alone with relative advantages. Necessary but relatively poor processes get suppressed, and locally good-but-misaligned processes get over-weighted. A prior or alignment term (next section) is needed to prevent collapse. The prior keeps high-prior processes from being suppressed, while distribution alignment ties process rewards to outcome rewards so the final outcome advantage safely guides process-level updates.

Distribution Alignment PRPO shifts the process-advantage distribution using the outcome advantage, as shown in the bottom of Figure 2:

$$\mathbb{E}[\mathcal{P}'_{\text{process}}] = \mathbb{E}[\mathcal{P}_{\text{process}}] + \mathbb{E}[\mathcal{P}_{\text{outcome}}],$$

producing a fused per-token advantage

$$\mathbf{AF}(\tau) = \mathbf{z}_{\text{process}}(\tau) + \beta(\tau),$$

Concretely, $AF_t(\tau) = z_{\text{process},t}(\tau) + \beta(\tau)$ for every token position t, so each token inherits the segment-normalized PRM score plus a broadcasted, group-centered outcome shift. The policy loss becomes

$$\mathcal{L}_{\text{PRPO}} = \mathbb{E}_{\tau_i}\left[\mathbf{AF}(\tau)(x_t^{(i)}) \log \frac{\pi_\theta(x_t^{(i)}|x_{<t}^{(i)})}{\pi_{\text{ref}}(x_t^{(i)}|x_{<t}^{(i)})}\right].$$

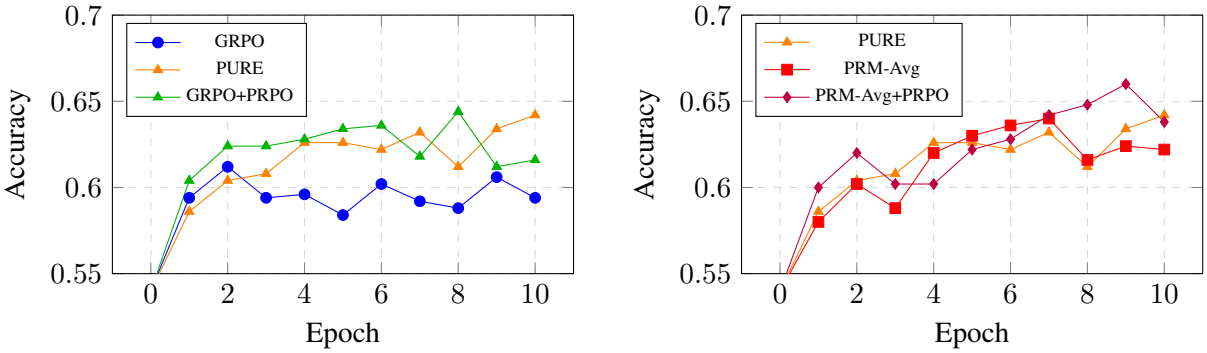
4 Experiments

Table 1: Time cost analysis when one training use the PRM server and three training use the server simultaneously.

Method	Time Cost (s)
w/o PRM step time	46.8
w PRM step time (1 training)	80.8
w PRM step time (3 trainings)	84.0

We train on 8 H200 GPUs with batch size 128, learning rate 1×10^{-6}, KL loss coefficient 0.001, clip ratio 0.2, maximum new tokens 2048, and rollout number 8. For segmentation, we pick five entropy spikes that are at least ten tokens apart (k=5, n=10).

For PRM, we choose Qwen2.5-Math-PRM-7B as the PRM model, and deploy it on 8 H200 GPUs according to official guidance to normalize the output value with $[0, 1]$ range. We did an analysis on the time cost of PRM when training Qwen-Math-1.5B in Table 1. In this experiment, we take GRPO as model training without PRM, and



(a) GRPO vs GRPO+PRPO.

(b) PRM-Avg vs PRM-Avg+PRPO.

Figure 3: Training accuracy on MATH. PRPO steadily improves both baselines.

Table 2: Accuracy comparison across datasets (%).

Model	Algorithm	MATH pass@1	AMC2023 mean32	AIME2025 mean32	AIME2024 mean32
Qwen-M-1.5	4-shot (Yang et al., 2024)	49.80	-	-	-
	GRPO	61.20	47.89	**7.00**	7.60
	PRM-Avg	64.0	46.33	6.25	9.69
	PURE	64.2	48.44	5.39	11.56
	GRPO+**PRPO**	64.4	50.78	4.20	**12.71**
	PRM-Avg+**PRPO**	**66.00**	**51.8**	6.03	11.46
Qwen-M-7	4-shot (Yang et al., 2024)	55.40	-	-	-
	GRPO	69.80	57.82	7.65	17.81
	PRM-Avg	71.00	55.8	8.94	9.69
	PURE	71.4	55.62	9.81	18.85
	GRPO+**PRPO**	**72.00**	56.72	10.99	**23.65**
	PRM-Avg+**PRPO**	70.60	**59.38**	**11.21**	22.08

PRPO+GRPO as model training with PRM. We trained each model for 1 epoch and calculated the average time cost for each step. Even with PRM and training three models simultaneously, the additional time cost is about 37 seconds each step (46.8 s vs 84.0 s). Comparing to the benefit and higher upper bound brought by PRM, the time cost increase is acceptable.

We compare PRPO with GRPO and PRM-Avg (Zou et al., 2025; Ren, 2025). PRM-Avg adds the average process reward to the outcome reward, which is a common practice in recent work of using PRM in RL training like ReasonFlux (Zou et al., 2025) and lsrl (Ren, 2025). It's worth noting that there's a slight difference between lsrl and reasonflux, where lsrl use a discounted sum of process reward, while reasonflux use naive average of process reward. As reasonflux is a more recent and widely referenced work, we use naive average used in reasonflux for PRM-avg. As PRPO provides another insight of advantage calculation, it can apply to both baselines. We also compare our algorithm with the current state-of-the-art baseline PURE (Cheng et al., 2025) algorithm, which is the only critic-free algorithm that tried to realize process-level credit assignment in critic-free framework. We implement PURE with the exact same reward and advantage settings in the original work to reach the optimal performance, where the output reward signal is 0-1 reward, and the assignment temperature is set as 0.1. We evaluate performance of each algorithm on Qwen2.5-Math-1.5B and Qwen2.5-Math-7B (Yang et al., 2024).

We conduct our algorithm and train the models with the VeRL training framework (Sheng et al., 2025).

Models are trained on the training split of MATH (Hendrycks et al., 2021) with 12000 samples with early stopping. Evaluation covers MATH, AMC 2023, AIME 2025, and AIME 2024 (Art of Problem Solving, 2025a,b) using greedy decoding (pass@1) for MATH as there is sufficient number of test cases in its test split (500 samples) and sampling 32 rollouts for AMC/AIME with temperature 0.7, top-p 0.9, and seeds fixed to 42 as there are limited test cases in AIME (30 samples each dataset)

Table 3: Pass rate comparison on Qwen2.5-Math-7B.

Method	Pass Rate (%)					Average
	pass@8	pass@16	pass@32	pass@64	pass@128	
AIME 2024						
GRPO	36.67	50.00	60.00	56.67	63.33	53.33
PRMAVG	26.67	33.33	33.33	43.33	53.33	38.00
PURE	50.00	50.00	63.33	60.00	63.33	57.33
GRPO+PRPO	46.67	53.33	60.00	63.33	60.00	56.67
PRMAVG+PRPO	46.67	53.33	60.00	63.33	70.00	**58.67**
AIME 2025						
GRPO	24.14	27.59	37.93	27.59	41.38	31.73
PRMAVG	20.69	24.14	27.59	34.48	37.93	28.97
PURE	27.5	31.03	34.48	37.93	44.83	35.15
GRPO+PRPO	31.03	31.03	31.03	37.93	48.28	**35.86**
PRMAVG+PRPO	20.69	34.48	34.48	41.38	41.38	34.48
AMC 2023						
GRPO	77.50	77.50	82.50	90.00	92.50	84.00
PRMAVG	80.00	80.00	85.00	87.50	87.50	84.00
PURE	90.00	87.50	90.00	95.00	97.50	92.00
GRPO+PRPO	85.00	85.00	90.00	97.50	97.50	91.00
PRMAVG+PRPO	85.00	90.00	95.00	97.50	100.00	**93.50**

Table 4: Ablation study (%).

Model	Algorithm	MATH pass@1
Qwen2.5-Math-1.5B	GRPO+**PRPO** w Random Split	2.4
	GRPO+**PRPO** w Uniform Split	29.8
	GRPO+**PRPO** w Entropy Based split	**64.4**
Qwen2.5-Math-1.5B	GRPO+**PRPO** w Relative distribution for process reward	64.0
	GRPO+**PRPO** w Predefined distribution for process reward	**64.4**

and AMC 2023 (40 samples). For algorithms other than PURE, rule based outcome rewards are ± 1 based on SymPy-verified correctness, plus a length penalty beyond 1024 tokens. Process rewards come from Qwen2.5-Math-PRM-7B.

We draw the accuracy curve of different algorithms trained on Qwen2.5-Math-1.5B in Figure 3. As shown in the figure, PRPO steadily improves both baselines, and generally fit quicker and reach higher accuracy than all baselines.

The length penalty is described as follows:

$$\text{length_penalty} = \begin{cases} 0 & \text{if length} \leq 1024 \\ \frac{\text{length}}{1024} & \text{if length} > 1024 \end{cases}$$

As shown in Table 2, our PRPO algorithm consistently outperforms both GRPO and PRM-Avg baselines across all evaluated datasets and model sizes. Qwen-M-1.5 refers to Qwen2.5-Math-1.5B model, and Qwen-M-7 refers to Qwen2.5-Math-7B model. For Qwen2.5-Math-1.5B, PRPO improves MATH pass@1 accuracy from 61.20% (GRPO) to 64.4% (+3.2%), and further to 66.00% when combined with PRM-Avg (64.00% vs 66.00%). On AMC2023 and AIME2024, our method also yields substantial gains, demonstrating its effectiveness in both standard and challenging mathematical benchmarks. Although our method does not perform as well as PRM-Avg on AIME2025, the gap is not significant (5.9% vs 7.00%), which can be attributed to capacity and fitting limits under dense supervision. PRM-Avg injects a dense per-segment signal; smaller or weaker models may not have enough capacity to disentangle noisy PRM scores from outcome supervision, effectively turning dense rewards into a length or variance penalty. This is especially pronounced on AIME2024 with the 7B model and on AIME2025 with the 1.5B model, where the tasks are harder and label noise is higher. In contrast, larger models (7B on AIME2025) can better absorb and align dense signals, which explains why the degradation diminishes as capacity increases. Comparing to PURE (Cheng et al., 2025), our algorithm also demonstrates its robustness across all datasets.

We also did an experiment on different pass rate accuracy of Qwen2.5-Math-7B model trained with

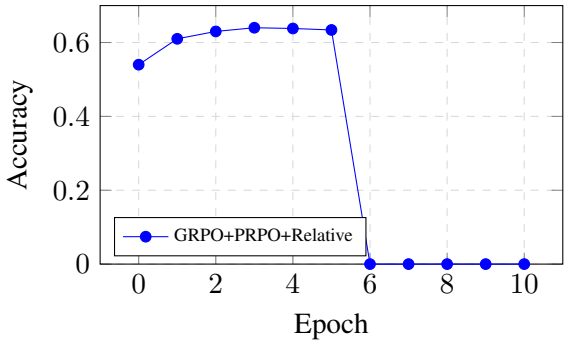


Figure 4: Ablation on relative process-reward normalization.

different methods. As shown in Table 3, our algorithm consistently outperforms both GRPO and PRM-Avg baselines across all evaluated datasets, which shows the stability and effectiveness of our algorithm when applied to models with enough parameters. It is worth noting that PRMAVG usually underperforms GRPO on AIME2025 and AIME2024, which shows the lack of robustness of using average process reward as outcome reward. However, with PRPO, both PRMAVG and GRPO can generally obtain great improvement on all datasets under different pass rate, showing the robustness of our algorithm.

These results validate our hypothesis that aligning process-level and outcome-level rewards enables more stable and effective policy optimization, especially in multi-process reasoning tasks where sparse signals are insufficient for robust learning.

Additionally, we did an ablation study on our entropy based split strategy. For comparison, we select random k points that are at least n tokens away from other points as random split, and equally divide the output sequence into k+1 parts as uniform split. As shown in Table 4, our algorithm with random split can only reach 2.4% and our algorithm with uniform split can only reach 29.8% of accuracy in MATH dataset, which are significantly lower than naive GRPO (61.20%) and our algorithm with entropy based split (64.40%), which shows the essence of entropy based split in our algorithm. We propose a possible reason to explain the the failure of random and uniform split strategies stems from the fact that these splits are not semantically-based: Within a single segment, there may exist both completely correct processes and completely incorrect processes. When such a segment receives a positive reward, the incorrect processes are incorrectly reinforced; conversely, when it receives a negative reward, the correct processes are incorrectly suppressed. Regardless of how the reward is assigned, a portion of the credit assignment will inevitably be problematic, ultimately leading to the collapse of model training.

We also did an ablation study on our predefined distribution for process-level advantage calculation. For comparison, we use the relative distribution of process-level advantage, which means the mean and standard deviation used to normalize and calculate process-level advantage is calculated each time based on the process rewards in the rollout. As shown in Table 4, our algorithm with relative distribution for process-level advantage can reach 64.0% of accuracy in MATH dataset and our algorithm with predefined distribution of process-level advantage can reach 64.4% of accuracy in MATH dataset, where there is no significant difference. Nevertheless, we draw a graph of the change of accuracy in each epoch of our algorithm with relative distribution for process-level advantage and our algorithm with predefined distribution for process-level advantage in Figure 3a and Figure 4, it can be seen that our algorithm with relative distribution for process-level advantage has its accuracy increase at first, but suddenly drop to nearly 0% since the 6th epoch, which shows the great instability of relative distribution for process-level advantage calculation. We believe this is caused by the Premature Termination Problem we discussed in the previous section, when the model starts to reach a level of accuracy, some essential processes, which are still good but are relatively lower, can be an outlier and come with an extremely low advantage, finally leading to the collapse of the model. This ablation experiment proves the essence of predefined distribution for process-level advantage calculation in our algorithm, in order to both improve performance and stability of RL training.

5 Conclusion

We present process Relative Policy Optimization, a critic-free method that aligns dense process rewards with sparse outcome rewards. PRPO normalizes PRM scores, aligns them with outcome advantages, and delivers stable per-token guidance. Experiments on mathematical reasoning benchmarks show consistent gains over GRPO and PRM-Avg while retaining critic-free efficiency.

6 Limitations

PRPO mitigates sparse rewards in GRPO by integrating dense PRM feedback, but it depends on PRM quality. In our work, we proposed a hypothesis about the possible collapse when training with process-only reward, and provide a possible mathematical proof of this hypothesis (shown in Appendix A) which is incomplete because a lack of definition of the output distribution of PRMs. A more rigorous proof can be done to prove the point and provide more insights about how to avoid the collapse. The fixed prior mean and variance used for process-reward normalization assume PRM outputs lie in $[0, 1]$, which is viable for most PRMs that normalize each output to $[0, 1]$, but may not be applicable to PRMs with other ranges or distributions; extending normalization to PRMs with other ranges or distributions is an open direction. Future work will jointly optimize PRMs with the policy and explore adaptive segmentation that combines semantic cues.

7 Potential Risks

There are potential risks in using PRPO, which is the same as the risks in using GRPO. However, PRPO is more robust than GRPO because it has a more stable distribution of process-level advantage, which is more likely to converge to a stable distribution. For example, when the model starts to reach a level of accuracy, some essential processes, which are still good but are relatively lower, can be an outlier and come with an extremely low advantage that cannot be resolved by our distribution alignment mechanism, finally leading to the collapse of the model. This is a potential risk of using PRPO, and we believe it is a rare case and can be mitigated by using a more robust PRM or a more robust segmentation strategy.

References

Art of Problem Solving. 2025a. American invitational mathematics examination (aime). https://artofproblemsolving.com/wiki/index.php?title=AIME_Problems_and_Solutions. Accessed: 2025-11-21.

Art of Problem Solving. 2025b. American invitational mathematics examination (aime). https://artofproblemsolving.com/wiki/index.php/2023_AMC_12A_Problems. Accessed: 2025-11-21.

Lang Cao, Renhong Chen, Yingtian Zou, Chao Peng, Huacong Xu, Yuxian Wang, Wu Ning, Qian Chen, Mofan Peng, Zijie Chen, Peishuo Su, Sirui Han, and Yitong Li. 2025. More bang for the buck: Process reward modeling with entropy-driven uncertainty. *Preprint*, arXiv:2503.22233.

Zengjue Chen, Runliang Niu, He Kong, Qi Wang, Qianli Xing, and Zipei Fan. 2025. Tgrpo :fine-tuning vision-language-action model via trajectory-wise group relative policy optimization. *Preprint*, arXiv:2506.08440.

Jie Cheng, Gang Xiong, Ruixi Qiao, Lijun Li, Chao Guo, Junle Wang, Yisheng Lv, and Fei-Yue Wang. 2025. Stop summation: Min-form credit assignment is all process reward model needs for reasoning. *Preprint*, arXiv:2504.15275.

Ning Dai, Zheng Wu, Renjie Zheng, Ziyun Wei, Wenlei Shi, Xing Jin, Guanlin Liu, Chen Dun, Liang Huang, and Lin Yan. 2025. Process supervision-guided policy optimization for code generation. *Preprint*, arXiv:2410.17621.

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

Xin Lai, Zhuotao Tian, Yukang Chen, Senqiao Yang, Xiangru Peng, and Jiaya Jia. 2024. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *Preprint*, arXiv:2406.18629.

Xingyu Lin, Yilin Wen, En Wang, Du Su, Wenbin Liu, Chenfu Bao, and Zhonghou Lv. 2025. Token-level policy optimization: Linking group-level rewards to token-level aggregation via markov likelihood. *Preprint*, arXiv:2510.09369.

Jia Liu, ChangYi He, YingQiao Lin, MingMin Yang, FeiYang Shen, and ShaoGuo Liu. 2025a. Ettrl: Balancing exploration and exploitation in llm test-time reinforcement learning via entropy mechanism. *Preprint*, arXiv:2508.11356.

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

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, and Abhinav Rastogi. 2024. Improve mathematical reasoning in language models by automated process supervision. *Preprint*, arXiv:2406.06592.

Hangliang Ren. 2025. Lsrl: Process-supervised grpo on latent recurrent states improves mathematical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 12534–12545.

Tao Ren, Jinyang Jiang, Hui Yang, Wan Tian, Minhao Zou, Guanghao Li, Zishi Zhang, Qinghao Wang, Shentao Qin, Yanjun Zhao, Rui Tao, Hui Shao, and Yijie Peng. 2025. Riskpo: Risk-based policy optimization via verifiable reward for llm post-training. *Preprint*, arXiv:2510.00911.

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

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

Mingyang Song, Zhaochen Su, Xiaoye Qu, Jiawei Zhou, and Yu Cheng. 2025. Prmbench: A fine-grained and challenging benchmark for process-level reward models. *Preprint*, arXiv:2501.03124.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. 2025. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *Preprint*, arXiv:2506.01939.

Muning Wen, Cheng Deng, Jun Wang, Weinan Zhang, and Ying Wen. 2024. Entropy-regularized token-level policy optimization for large language models. *CoRR*, abs/2402.06700.

Huimin Xu, Xin Mao, Feng-Lin Li, Xiao Bao Wu, Wang Chen, Wei Zhang, and Anh Tuan Luu. 2025. Full-step-dpo: Self-supervised preference optimization with step-wise rewards for mathematical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2025*.

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

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. The lessons of developing process reward models in mathematical reasoning. *Preprint*, arXiv:2501.07301.

Mingkang Zhu, Xi Chen, Zhongdao Wang, Bei Yu, Hengshuang Zhao, and Jiaya Jia. 2025. Tgdpo: Harnessing token-level reward guidance for enhancing direct preference optimization. *Preprint*, arXiv:2506.14574.

Jiaru Zou, Ling Yang, Jingwen Gu, Jiahao Qiu, Ke Shen, Jingrui He, and Mengdi Wang. 2025. Reasonflux-prm: Trajectory-aware prms for long chain-of-thought reasoning in llms. *Preprint*, arXiv:2506.18896.

Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Yuchen Zhang, Xinwei Long, Ermo Hua, Biqing Qi, Youbang Sun, Zhiyuan Ma, Lifan Yuan, Ning Ding, and Bowen Zhou. 2025. Ttrl: Test-time reinforcement learning. *Preprint*, arXiv:2504.16084.

A Possible Mathematical Reason for Premature Collapse in Process-Only Reward 1

The gradient of the log-probability for the prefix is:

$$\nabla_\theta \log p_\theta(x_{0:i}) = \sum_{j=0}^{i} \nabla_\theta \log \pi_\theta(x_j \mid x_{<j}).$$

Under the policy gradient update derived from (1), the expected change in $p_\theta(x_{0:t^\star})$ can be approximated by:

$$\mathbb{E}\left[\Delta p_\theta(x_{0:i})\right] \approx \alpha \cdot \mathbb{E}\left[\left(\sum_{j=0}^{i} A_j \nabla_\theta \log \pi_\theta(x_j \mid x_{\leq j})\right)^\top \nabla_\theta \log p_\theta(x_{\leq j})\right],$$

where α is the learning rate. Assuming gradients for different i are approximately orthogonal in expectation, we get:

$$\mathbb{E}\left[\Delta p_\theta(x_{0:i})\right] \approx \alpha \cdot \mathbb{E}\left[\sum_{j=0}^{i} A_j \|\nabla_\theta \log \pi_\theta(x_j \mid x_{\leq j})\|^2\right].$$

Using the empirical averages from conditions (2) and (3):

$$\begin{aligned}
\mathbb{E}\left[\Delta p_\theta(x_{0:t^\star})\right] &= \mathbb{E}\left[\Delta(p_\theta(x_{0:t^\star-1}) \cdot \pi_\theta(x_{t^\star} \mid x_{<t^\star}))\right] \\
&\approx \alpha \cdot \left(-a \sum_{i=0}^{t^\star-1} \mathbb{E}\left[\|\nabla_\theta \log \pi_\theta(x_i \mid x_{<i})\|^2\right] + b \cdot \mathbb{E}\left[\|\nabla_\theta \log \pi_\theta(x_{t^\star} \mid x_{<t^\star})\|^2\right]\right) \\
&\approx \alpha \cdot (-a \cdot (t^\star) + b) \cdot C,
\end{aligned}$$

where $C > 0$ approximates the average squared gradient norm. When condition (4) holds ($a \cdot t^\star > b$), the term inside the parentheses is negative. Therefore:

$$\mathbb{E}\left[\Delta p_\theta(x_{0:t^\star})\right] < 0.$$

Consequently, with repeated updates, the probability $p_\theta(x_{0:t^\star})$ tends to decrease exponentially in practice. This makes the model increasingly unlikely to generate sequences reaching position $t^\star$, effectively encouraging earlier termination.

B Entropy Based Segmentation Strategy

Algorithm 1 Entropy-based segmentation procedure.

```
Function EntropySegmentation(entropies, start_idx, outLen, max_branches=5, min_gap=10)
    if outLen − start_idx < max_branches + 1 then
        return (start_idx, outLen)
    end if
    anchors ← top-k entropy indices in entropies[start_idx:outLen]
    anchors ← anchors + start_idx
    anchors ← sort(anchors)
    filtered ← []
    for anchor in anchors (left-to-right) do
        if distance(anchor, pre-kept) ≥ min_gap then
            append anchor to filtered
        end if
    end for
    cuts ← []; last_cut ← start_idx
    for a in filtered do
        if a − last_cut ≥ min_gap then
            append a to cuts; last_cut ← a
        end if
    end for
    segments ← []; prev ← start_idx
    for c in cuts do
        append (prev, c) to segments; prev ← c
    end for
    append (prev, outLen) to segments
    sanitized ← []; cur ← start_idx
    for (s, e) in segments do
        s ← max(start_idx, min(outLen, s))
        e ← max(start_idx, min(outLen, e))
        if e ≤ s then
            continue
        end if
        if s > cur then
            append (cur, s) to sanitized
        end if
        append (s, e) to sanitized; cur ← e
    end for
    if cur < outLen then
        append (cur, outLen) to sanitized
    end if
    if sanitized is empty then
        return (start_idx, outLen)
    else
        return sanitized
    end if
end Function
```

It's worth noting that we set k=5 and n=10 across all length of output sequences, except for one case when their is less than 5 tokens, we take the whole output sequence as one segment.

C Early Stop Point for Each Trainig

C.1 Qwen2.5-Math-1.5B

- GRPO: Epoch 2
- PRM-Avg: Epoch 7
- PURE: Epoch 10
- GRPO+**PRPO**: Epoch 8
- PRM-Avg+**PRPO**: Epoch 9

C.2 Qwen2.5-Math-7B

- GRPO: Epoch 9
- PRM-Avg: Epoch 7
- PURE: Epoch 10
- GRPO+**PRPO**: Epoch 4
- PRM-Avg+**PRPO**: Epoch 3

D Disclosure of use of AI Tools

We used the following AI tools for this research:

- Cursor for code revision and debugging.
- ChatGPT for evaluation of writing and proofreading.
- Gemini used to help design illustrations.

E Disclosure of license of the artifacts we used

The artifacts we used are all open-source, and we used the following licenses:

- The model Qwen2.5-Math-1.5B is released under the Apache License 2.0
- The model Qwen2.5-Math-7B is released under the Apache License 2.0
- The model Qwen2.5-Math-PRM-7B is released under the Apache License 2.0
- The dataset MATH is released under the MIT License
- The training framework VeRL is released under the Apache License 2.0

We strictly follow the license of the open-source artifacts we used.

The data of AMC 2023, AIME 2025, and AIME 2024 are extracted from open website Art of Problem Solving (Art of Problem Solving, 2025a). We strictly follow the terms of use of this website and did not use the data obtained from the website for any commercial purpose.