
Beyond Correctness: Harmonizing Process and Outcome Rewards through RL Training

Anonymous Author(s)

Affiliation

Address

email

Abstract

1 Reinforcement learning with verifiable rewards (RLVR) has emerged as a predominant
2 paradigm for mathematical reasoning tasks, offering stable improvements
3 in reasoning ability. However, Outcome Reward Models (ORMs) in RLVR are
4 too coarse-grained to distinguish flawed reasoning within correct answers or valid
5 reasoning within incorrect answers. This lack of granularity introduces noisy and
6 misleading gradients significantly and hinders further progress in reasoning process
7 quality. While Process Reward Models (PRMs) offer fine-grained guidance for
8 intermediate steps, they frequently suffer from inaccuracies and are susceptible to
9 reward hacking.

10 To resolve this dilemma, we introduce PRocess cOnsistency Filter (PROF), an
11 effective data process curation method that harmonizes noisy, fine-grained process
12 rewards with accurate, coarse-grained outcome rewards. Rather than naively
13 blending PRM and ORM in the objective function (Zou et al., 2025), PROF lever-
14 ages their complementary strengths through consistency-driven sample selection.
15 Our approach retains correct responses with higher averaged process values and
16 incorrect responses with lower averaged process values, while maintaining positive/negative
17 training sample balance. Extensive experiments demonstrate that our
18 method not only consistently improves final accuracy over 4% compared to the
19 blending approaches, but also strengthens quality of intermediate reasoning steps.

20 1 Introduction

21 Verifiable rewards have spurred the widest attention recently because they reliably improve the
22 performance on reasoning tasks with easily verifiable outcomes, such as mathematical and coding
23 problems (Cobbe et al., 2021; Jaech et al., 2024; Shao et al., 2024; Xiong et al., 2025b). However,
24 since the verifiers can only verify the outcome results, the rewards are too sparse and coarse to measure
25 and supervise the reasoning quality in intermediate steps. For instance, if a correct answer contains
26 flawed logic, Outcome Reward Models (ORMs) cannot distinguish it from a completely correct
27 response. We present a classic example from the training data in Table 5, which has invalid reasoning
28 but happens to obtain the correct answer. Incorporating such flawed examples into training process
29 introduces unreliable gradients, leading to significant instability and misguided learning. Moreover,
30 the quality and interpretability of Chain of Thought (CoT) are crucial for practical reasoning ability
31 of a model, not just the accuracy of final answers (Zhu et al., 2025; Lyu et al., 2023; Yeo et al., 2024).
32 The lack of faithfulness during CoT is also observed by (Baker et al., 2025; Chen et al., 2025b),
33 limiting applications in areas such as LLM safety monitoring and interpretation.

34 Hence, the limitation of ORM can be partially addressed by using LLM-as-a-judge or Monte-Carlo
35 (MC) estimation to provide step-wise judgments or values (Wang et al., 2023; Zheng et al., 2024).
36 However, the cost of inferring LLM step-wise judgments or MC estimation at each iteration during

37 online training is so high. Hence, it is inefficient and expensive to infer the step-wise scores or
38 values for online training. Alternatively, an efficient solution is to use the pre-trained Process Reward
39 Models (PRMs) (Lightman et al., 2023; Zhang et al., 2025). However, applying these models to the
40 online training process often suffers from misspecification and distribution shift due to the limitations
41 of offline training data. Especially in boundary cases where the policy encounters difficult problems
42 and produces rarely seen responses, PRMs often fail to judge them correctly, thus leading to severe
43 reward hacking (Michaud et al., 2020; Tien et al., 2022). Even if some works (Zha et al., 2025; Cui
44 et al., 2025) attempt to co-train the policy and PRMs online, they can only train in implicit ways such
45 as using implicit generative reward or aligning process rewards with outcomes.

46 Although numerous works have made enormous efforts to train PRMs offline or online, the problem
47 of effectively coordinating PRMs with outcome-verifiable rewards remains largely underexplored.
48 Existing approaches typically combine process and outcome rewards in a simple weighted manner
49 (Zha et al., 2025; Cui et al., 2025; Zou et al., 2025), which is vulnerable for reward hacking due to
50 the noises and misspecification in PRMs. Therefore, in this paper, instead of developing another
51 PRM, we focus on how to robustly integrate a pre-trained PRM into the online training process, i.e.,
52 how to harmonize the accurate but coarse-grained ORMs with fine-grained but noisy Process Reward
53 Models (PRMs) in Reinforcement Learning (RL)?

54 In this work, instead of fine-tuning another PRM, we answer this question with a **PRocess cOnsistency**
55 **Filtering (PROF)** framework, a data curation strategy based on process-outcome consistency. PROF
56 oversamples more responses at training time, and then, ranks and filters the responses by the
57 consistency between their PRMs and ORMs. Specifically, it removes samples where the process
58 and outcome signals conflict—such as correct responses derived from flawed reasoning, or incorrect
59 responses that contain sound reasoning steps. By filtering out these inconsistent samples, PROF
60 eliminates conflicting and noisy gradients. Furthermore, observing that correct and incorrect responses
61 have different consistency distributions, we rank each group separately to maintain a balanced training
62 ratio. PROF is a modular framework that can be combined with RL algorithms like Group Relative
63 Policy Optimization (GRPO) for online training.

64 We conduct extensive experiments to validate the improvement of PROF-GRPO on both outcome
65 accuracy and process reasoning quality at diverse math reasoning benchmarks using both Qwen
66 (Yang et al., 2024) and LLaMA (Dubey et al., 2024) models. To summarize, we highlight our key
67 contributions as follows:

- 68 • We propose **PRocess cOnsistency Filtering (PROF)** to robustly integrate noisy Process
69 Reward Models (PRMs) with Outcome Reward Models (ORMs). Compared to the GRPO-
70 type algorithms that only leverage outcome rewards, our implementation PROF-GRPO
71 effectively distinguishes the inconsistent trajectories, such as correct answers with flawed
72 reasoning steps or incorrect answers with mostly valid steps. Moreover, unlike prior
73 approaches that simply blend PRMs and ORMs, our method only relies on PRMs to rank
74 and filter rather than directly involving them into gradients. This separation avoids reward
75 hacking and entropy collapse, thus achieving stable performance gains throughout training.
- 76 • We conduct extensive studies to demonstrate that PROF-GRPO not only increases the final
77 outcome accuracy but also shapes the intermediate reasoning steps and improves the process
78 reasoning quality. Various metrics such as Monte-Carlo estimation, LLM-as-a-judge are used
79 to validate that our method enable models to segment reasoning trajectories into **detailed**
80 **and easy-to-verify** steps.
- 81 • We conduct a series of ablation studies to illustrate the importance of separating the correct
82 and incorrect responses during the filtration. Meanwhile, we investigate various ways of cal-
83 culating the consistency and filtering, and ablate on LLaMA base models for generalization.

84 2 Method

85 LLM is a policy distribution such that given a prompt x , it provides density $\pi(a|x)$ of generating each
86 response a . For mathematical reasoning tasks with binary verifiable rewards, there exists a verifier
87 mapping prompt-response pairs (x, a) to a scalar reward $r_o(x, a) \in \{-1, 1\}$. For each prompt, we
88 can generate a group of responses and their corresponding responses with the verifier $\{(a_i, r_{o,i})\}_{i=1}^G$.

89 **GRPO.** (Shao et al., 2024) proposes this policy gradient algorithm that simplifies the Proximal
90 Policy Optimization (PPO) (Schulman et al., 2017) by only computing the advantage based on the
91 outcome rewards in a group. Instead of maintaining and updating another value network, GRPO
92 computes the advantage by standardizing the outcome rewards within a group:

$$A_i = \frac{r(x, a_i) - \text{mean}(\{r(x, a_j)\}_{j=1}^n)}{\text{std}(\{r(x, a_j)\}_{j=1}^n) + \delta}, \quad i = 1, \dots, n,$$

93 where $r(x, a_i)$ is the reward for a given response and $\delta > 0$ is a small constant for numerical stability.
94 Let a_t denote the t -th token of response a and $a_{<t}$ denotes (a_1, \dots, a_{t-1}) . This advantage is then
95 incorporated into a clipped surrogate objective function, which is optimized to update the policy from
96 $\pi_{\theta_{\text{old}}}$ to π_{θ} :

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{x \sim \mathcal{D}} \left[\frac{1}{n} \sum_{i=1}^n \frac{1}{|a_i|} \sum_{t=1}^{|a_i|} \min \left(\frac{\pi_{\theta}(a_i, < t | x)}{\pi_{\theta_{\text{old}}}(a_i, < t | x)} A_i, \text{clip} \left(\frac{\pi_{\theta}(a_i, < t | x)}{\pi_{\theta_{\text{old}}}(a_i, < t | x)}, 1 - \epsilon, 1 + \epsilon \right) A_i \right) \right].$$

97 Although this approach stabilizes the online policy optimization and is efficient, the sparse reward
98 signal limits further improvement on the intermediate reasoning steps.

99 **Process Reward Model (PRM).** For a response a composed of multiple reasoning steps $a =$
100 (a^1, \dots, a^H) , we follow previous works (Zheng et al., 2024; Zhang et al., 2025; Zou et al., 2025) to
101 use a newline as a sign for a new step. For each step a^h , the PRM r^h maps it, the previous steps and the
102 prompt $(x, a^{\leq h})$ to a scalar $r^h(x, a^{\leq h})$, where we use the short-hand notation $a^{\leq h} = (a^1, \dots, a^h)$.

103 **Our Method PROF: Process Consistency Filter Framework** We propose PROF in Algorithm 1
104 to incorporate the consistency of PRMs and ORMs robustly after the rollout phase, and also present a
105 visualization in Figure 2. First, we generate G samples and get the outcome reward. Then, we call
106 the PRM to generate step-wise rewards for each rollout and compute the trajectory-wise consistency
107 score r^{PRO} by taking the mean over the step-wise rewards and adding a step length regularization
108 in equation 1, where λ is the regularization parameter and H_{λ} is the threshold for the penalized
109 step number. This regularization is to ensure that samples with no step segments or over-long steps
110 are discarded in the correct group. The samples are divided into two subgroups: \mathcal{G}_+ contains the
111 correct samples with $r_o = 1$, and \mathcal{G}_- contains the incorrect samples with $r_o = -1$. Inspired by
112 (Xu et al., 2025), the numbers to discard in each subgroup k_+, k_- are calculated to maximize the
113 outcome-reward variance of the final kept samples $k_+ k_- / (k_+ + k_-)^2$. Since $k_+ + k_- = m$ is fixed,
114 $k_+ k_- = k_+ (m - k_+)$ should be maximized and the maximum is obtained when k_+ is closest to $m/2$
115 under the constraint $k_+ \leq n_+, k_- \leq n_-$. This implies that the ratio of correct and incorrect responses
116 should be balanced. After that, we use r^{PRO} to rank and filter the correct group and randomly filter
117 the incorrect group. Finally, we collect the kept m trajectories for policy update.

118 3 Experiments

119 **Setup** We focus on mathematical reasoning tasks in this work. For online training, we use the
120 prompt set Numina-Math (Beeching et al., 2024) containing nearly 860k math problems with ground-
121 truth answers ranging from Chinese high school math exercises to US and international mathematics
122 Olympiad competition problems. We choose Qwen2.5-Math-1.5B-base, Qwen2.5-Math-7B-base
123 (Yang et al., 2024) as the training base models. For the PRM, we use Qwen2.5-Math-PRM-7B
124 (Zhang et al., 2025) to generate process rewards. More details are provided in Appendix B. The
125 models' performance is evaluated on 5 benchmarks: Math500 (Hendrycks et al., 2021), Minerva
126 Math (Lewkowycz et al., 2022), Olympiad Bench (He et al., 2024), AMC2023¹ and AIME2024². We
127 mainly use average@16 for evaluation, i.e., the accuracy is averaged over 16 responses per prompt
128 under temperature 1.0. The models are allowed to generate 4096 tokens.

129 **Main Results** We summarize our main results in Table 1, where Blend denotes a common way that
130 mixes the PRM with outcome rewards (Zha et al., 2025; Cui et al., 2025; Zou et al., 2025). Following
131 (Zou et al., 2025), the PRMs are averaged over steps for each response, weighted by a parameter
132 β , and added to outcome rewards. We use parameter $\beta = 0.8$ according to Table 5 of (Zou et al.,

¹<https://huggingface.co/datasets/math-ai/amc23>

²<https://huggingface.co/datasets/math-ai/aime24>

Model	Algorithm	Math500	Minerva Math	Olympiad Bench	AIME24	AMC23	Average
Qwen2.5-Math-1.5B-base	Base	39.9	11.4	19.1	3.5	23.6	19.5
	GRPO	70.3	29.1	33.0	9.0	44.5	37.2
	Blend	67.6	27.8	31.1	7.7	42.5	35.3
	PROF-GRPO	73.2	30.0	36.1	9.6	49.1	39.6
Qwen2.5-Math-7B-base	Base	42.0	12.8	19.2	12.9	30.0	23.4
	GRPO	81.6	37.2	45.5	20.6	64.4	49.9
	Blend	81.7	36.7	45.0	15.2	58.0	47.3
	PROF-GRPO	83.1	39.0	47.8	17.5	70.9	51.7

Table 1: Performance of different algorithms across five benchmarks including Math500 (Hendrycks et al., 2021), Minerva Math (Lewkowycz et al., 2022), Olympiad Bench (He et al., 2024), AMC2023 and AIME2024. We denote Blend-PRM-GRPO by Blend for short. We tune all the algorithms to their best performance. The reported accuracy is average@16 under temperature 1.0.

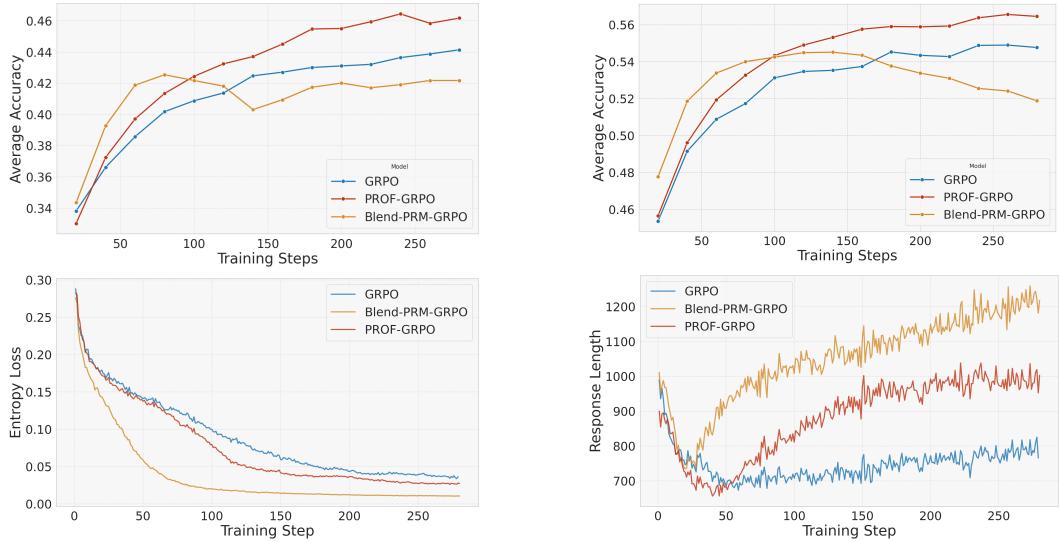


Figure 1: The learning dynamics of PROF-GRPO initialized from Qwen2.5-Math-1.5B-base (upper left) and Qwen2.5-Math-7B-base (upper right) in comparison of GRPO and Blend-PRM-GRPO. The y-axis is the average@16 accuracy and is further averaged on Math500, Minerva Math and Olympiad Bench. Entropy loss (lower left) and response length (lower right) of the models initialized from Qwen2.5-Math-7B-base.

133 2025). Our main findings are as follows. As shown in Table 1, our proposed method, PROF-GRPO,
134 consistently outperforms GRPO and Blend-PRM-GRPO over various benchmarks.³ The learning
135 dynamics in Figure 1 corroborate these findings, illustrating that PROF-GRPO steadily maintains a
136 consistent performance advantage over GRPO and Blend-PRM-GRPO throughout training process.

137 4 Conclusion and Future Work

138 This work introduces Process Consistency Filter (PROF), a novel data curation technique that filters
139 generated responses by the data PRM-ORM consistency, and maintains the balance of correct-
140 incorrect ratios. We demonstrate its effectiveness in both consistently improving the accuracy of
141 obtaining correct final answers and shaping the policy model to generate more detailed and fine-
142 grained segmented intermediate reasoning steps. Particularly, PROF is a general filtration framework
143 without reliance on specific PRMs or the RL algorithms. Thus, the use of Qwen2.5-Math-PRM-7B
144 as the PRM in our experiments is not a limitation. Exploring the integration of PROF with more
145 accurate or diverse PRMs remains an interesting direction for future work. Additionally, how to
146 extend our method to other reasoning tasks, such as coding (Jimenez et al., 2023) and web navigation
147 (Zhou et al., 2023) deserves to be explored.

³Although PROF-GRPO underperformed GRPO on AIME24 for Qwen2.5-Math-7B-base, given the dataset's small size of only 30 samples, the performance difference may not be statistically significant.

148 **References**

149 Baker, B., Huizinga, J., Gao, L., Dou, Z., Guan, M. Y., Madry, A., Zaremba, W., Pachocki, J.,
150 and Farhi, D. (2025). Monitoring reasoning models for misbehavior and the risks of promoting
151 obfuscation. *arXiv preprint arXiv:2503.11926*.

152 Beeching, E., Huang, S. C., Jiang, A., Li, J., Lipkin, B., Qina, Z., Rasul, K., Shen, Z., Solet-
153 skyi, R., and Tunstall, L. (2024). Numinamath 7b cot. [https://huggingface.co/AI-M0/](https://huggingface.co/AI-M0/NuminaMath-7B-CoT)
154 *NuminaMath-7B-CoT*.

155 Bradley, R. A. and Terry, M. E. (1952). Rank analysis of incomplete block designs: I. the method of
156 paired comparisons. *Biometrika*, 39(3/4):324–345.

157 Chen, H., Zheng, K., Zhang, Q., Cui, G., Cui, Y., Ye, H., Lin, T.-Y., Liu, M.-Y., Zhu, J., and Wang,
158 H. (2025a). Bridging supervised learning and reinforcement learning in math reasoning. *arXiv*
159 *preprint arXiv:2505.18116*.

160 Chen, Y., Benton, J., Radhakrishnan, A., Uesato, J., Denison, C., Schulman, J., Somani, A., Hase, P.,
161 Wagner, M., Roger, F., et al. (2025b). Reasoning models don't always say what they think. *arXiv*
162 *preprint arXiv:2505.05410*.

163 Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton,
164 J., Nakano, R., et al. (2021). Training verifiers to solve math word problems. *arXiv preprint*
165 *arXiv:2110.14168*.

166 Cui, G., Yuan, L., Wang, Z., Wang, H., Li, W., He, B., Fan, Y., Yu, T., Xu, Q., Chen, W., et al. (2025).
167 Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*.

168 Dong, H., Xiong, W., Goyal, D., Zhang, Y., Chow, W., Pan, R., Diao, S., Zhang, J., Shum, K., and
169 Zhang, T. (2023). Raft: Reward ranked finetuning for generative foundation model alignment.
170 *arXiv preprint arXiv:2304.06767*.

171 Dong, H., Xiong, W., Pang, B., Wang, H., Zhao, H., Zhou, Y., Jiang, N., Sahoo, D., Xiong, C.,
172 and Zhang, T. (2024). Rlhf workflow: From reward modeling to online rlhf. *arXiv preprint*
173 *arXiv:2405.07863*.

174 Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A.,
175 Yang, A., Fan, A., et al. (2024). The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

176 Eisenstein, J., Nagpal, C., Agarwal, A., Beirami, A., D'Amour, A., Dvijotham, D., Fisch, A., Heller,
177 K., Pfohl, S., Ramachandran, D., et al. (2023). Helping or herding? reward model ensembles
178 mitigate but do not eliminate reward hacking. *arXiv preprint arXiv:2312.09244*.

179 He, C., Luo, R., Bai, Y., Hu, S., Thai, Z. L., Shen, J., Hu, J., Han, X., Huang, Y., Zhang, Y., et al.
180 (2024). Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual
181 multimodal scientific problems. *arXiv preprint arXiv:2402.14008*.

182 Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt,
183 J. (2021). Measuring mathematical problem solving with the math dataset. *arXiv preprint*
184 *arXiv:2103.03874*.

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

187 Jimenez, C. E., Yang, J., Wettig, A., Yao, S., Pei, K., Press, O., and Narasimhan, K. (2023). Swe-
188 bench: Can language models resolve real-world github issues? *arXiv preprint arXiv:2310.06770*.

189 Khalifa, M., Agarwal, R., Logeswaran, L., Kim, J., Peng, H., Lee, M., Lee, H., and Wang, L. (2025).
190 Process reward models that think. *arXiv preprint arXiv:2504.16828*.

191 Kim, S. S., Liao, Q. V., Vorvoreanu, M., Ballard, S., and Vaughan, J. W. (2024). "i'm not sure, but...":
192 Examining the impact of large language models' uncertainty expression on user reliance and trust.
193 In *Proceedings of the 2024 ACM conference on fairness, accountability, and transparency*, pages
194 822–835.

195 Lewkowycz, A., Andreassen, A., Dohan, D., Dyer, E., Michalewski, H., Ramasesh, V., Slone, A.,
 196 Anil, C., Schlag, I., Gutman-Solo, T., et al. (2022). Solving quantitative reasoning problems with
 197 language models. *Advances in neural information processing systems*, 35:3843–3857.

198 Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J.,
 199 Sutskever, I., and Cobbe, K. (2023). Let’s verify step by step. In *The Twelfth International
 200 Conference on Learning Representations*.

201 Lin, Y., Lin, H., Xiong, W., Diao, S., Liu, J., Zhang, J., Pan, R., Wang, H., Hu, W., Zhang, H., et al.
 202 (2023). Mitigating the alignment tax of rlhf. *arXiv preprint arXiv:2309.06256*.

203 Luo, L., Liu, Y., Liu, R., Phatale, S., Guo, M., Lara, H., Li, Y., Shu, L., Zhu, Y., Meng, L., Sun,
 204 J., and Rastogi, A. (2024). Improve mathematical reasoning in language models by automated
 205 process supervision.

206 Lyu, Q., Havaldar, S., Stein, A., Zhang, L., Rao, D., Wong, E., Apidianaki, M., and Callison-Burch, C.
 207 (2023). Faithful chain-of-thought reasoning. In *The 13th International Joint Conference on Natural
 208 Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for
 209 Computational Linguistics (IJCNLP-AACL 2023)*.

210 Michaud, E. J., Gleave, A., and Russell, S. (2020). Understanding learned reward functions. *arXiv
 211 preprint arXiv:2012.05862*.

212 Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy
 213 optimization algorithms. *arXiv preprint arXiv:1707.06347*.

214 Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y., Wu, Y., et al.
 215 (2024). Deepseekmath: Pushing the limits of mathematical reasoning in open language models.
 216 *arXiv preprint arXiv:2402.03300*.

217 Sheng, G., Zhang, C., Ye, Z., Wu, X., Zhang, W., Zhang, R., Peng, Y., Lin, H., and Wu, C. (2025).
 218 Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European
 219 Conference on Computer Systems*, pages 1279–1297.

220 Tien, J., He, J. Z.-Y., Erickson, Z., Dragan, A. D., and Brown, D. S. (2022). Causal confusion and
 221 reward misidentification in preference-based reward learning. *arXiv preprint arXiv:2204.06601*.

222 Wang, P., Li, L., Shao, Z., Xu, R., Dai, D., Li, Y., Chen, D., Wu, Y., and Sui, Z. (2023). Math-
 223 shepherd: Verify and reinforce llms step-by-step without human annotations. *arXiv preprint
 224 arXiv:2312.08935*.

225 Xiong, W., Shi, C., Shen, J., Rosenberg, A., Qin, Z., Calandriello, D., Khalman, M., Joshi, R., Piot,
 226 B., Saleh, M., et al. (2024a). Building math agents with multi-turn iterative preference learning.
 227 *arXiv preprint arXiv:2409.02392*.

228 Xiong, W., Yao, J., Xu, Y., Pang, B., Wang, L., Sahoo, D., Li, J., Jiang, N., Zhang, T., Xiong, C., et al.
 229 (2025a). A minimalist approach to llm reasoning: from rejection sampling to reinforce. *arXiv
 230 preprint arXiv:2504.11343*.

231 Xiong, W., Zhang, H., Jiang, N., and Zhang, T. (2024b). An implementation of generative prm.

232 Xiong, W., Zhang, H., Ye, C., Chen, L., Jiang, N., and Zhang, T. (2025b). Self-rewarding correction
 233 for mathematical reasoning. *arXiv preprint arXiv:2502.19613*.

234 Xiong, W., Zhao, W., Yuan, W., Golovneva, O., Zhang, T., Weston, J., and Sukhbaatar, S. (2025c).
 235 Stepwiser: Stepwise generative judges for wiser reasoning.

236 Xu, Y. E., Savani, Y., Fang, F., and Kolter, Z. (2025). Not all rollouts are useful: Down-sampling
 237 rollouts in llm reinforcement learning. *arXiv preprint arXiv:2504.13818*.

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

241 Yeo, W. J., Satapathy, R., Goh, R. S. M., and Cambria, E. (2024). How interpretable are reasoning
242 explanations from prompting large language models? *arXiv preprint arXiv:2402.11863*.

243 Yu, P., Yuan, W., Golovneva, O., Wu, T., Sukhbaatar, S., Weston, J., and Xu, J. (2025a). Rip: Better
244 models by survival of the fittest prompts. *arXiv preprint arXiv:2501.18578*.

245 Yu, Q., Zhang, Z., Zhu, R., Yuan, Y., Zuo, X., Yue, Y., Dai, W., Fan, T., Liu, G., Liu, L., et al.
246 (2025b). Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint
247 arXiv:2503.14476*.

248 Yuan, W., Pang, R. Y., Cho, K., Sukhbaatar, S., Xu, J., and Weston, J. (2024). Self-rewarding
249 language models. *arXiv preprint arXiv:2401.10020*, 3.

250 Zha, K., Gao, Z., Shen, M., Hong, Z.-W., Boning, D. S., and Katabi, D. (2025). RI tango: Reinforcing
251 generator and verifier together for language reasoning. *arXiv preprint arXiv:2505.15034*.

252 Zhang, C., Shen, W., Zhao, L., Zhang, X., Qi, L., Dou, W., and Bian, J. (2024). Policy filtration in
253 rlhf to fine-tune llm for code generation.

254 Zhang, Z., Zheng, C., Wu, Y., Zhang, B., Lin, R., Yu, B., Liu, D., Zhou, J., and Lin, J. (2025).
255 The lessons of developing process reward models in mathematical reasoning. *arXiv preprint
256 arXiv:2501.07301*.

257 Zhao, J., Liu, R., Zhang, K., Zhou, Z., Gao, J., Li, D., Lyu, J., Qian, Z., Qi, B., Li, X., and Zhou, B.
258 (2025). Genprm: Scaling test-time compute of process reward models via generative reasoning.

259 Zheng, C., Zhang, Z., Zhang, B., Lin, R., Lu, K., Yu, B., Liu, D., Zhou, J., and Lin, J. (2024). Process-
260 bench: Identifying process errors in mathematical reasoning. *arXiv preprint arXiv:2412.06559*.

261 Zhou, S., Xu, F. F., Zhu, H., Zhou, X., Lo, R., Sridhar, A., Cheng, X., Ou, T., Bisk, Y., Fried, D., et al.
262 (2023). Webarena: A realistic web environment for building autonomous agents. *arXiv preprint
263 arXiv:2307.13854*.

264 Zhu, D., Wei, X., Zhao, G., Wu, W., Zou, H., Ran, J., Wang, X., Sun, L., Zhang, X., and Li, S.
265 (2025). Chain-of-thought matters: improving long-context language models with reasoning path
266 supervision. *arXiv preprint arXiv:2502.20790*.

267 Zou, J., Yang, L., Gu, J., Qiu, J., Shen, K., He, J., and Wang, M. (2025). Reasonflux-prm: Trajectory-
268 aware prms for long chain-of-thought reasoning in llms. *arXiv preprint arXiv:2506.18896*.

269 **A Related Works**

270 **Sample Filtering in Reinforcement Learning for LLM.** A key challenge in applying reinforcement learning to LLM applications is the imperfection of reward signals. These signals stem from 271 a learned reward model, such as Reinforcement Learning from Human Feedback (RLHF), or are 272 sparse, delivered only at the end of a trajectory (e.g. RLVR). In RLHF, the reward model is trained on 273 human-annotated pairwise comparisons, typically using a Bradley-Terry model (Bradley and Terry, 274 1952). Due to inherent human disagreement and finite training data, the model develops shortcuts 275 that RL algorithms can exploit (Lin et al., 2023; Eisenstein et al., 2023) to chase for a fake high 276 reward. Consequently, these rewards may not fully align with the underlying intended goals, leading 277 to reward hacking. 278

279 Data filtering, a data curation technique, has proven effective in mitigating this issue across various 280 LLM applications with RL. A prominent line of work proposes filtering training pairs based on the 281 reward gap between the chosen and rejected responses (Yuan et al., 2024; Dong et al., 2024; Xiong 282 et al., 2024a; Zhang et al., 2024). The high-level intuition is that a larger reward gap indicates higher 283 model confidence, making these pairs less noisy and more reliable for training when the reward 284 model is well-calibrated. Moreover, Kim et al. (2024); Yu et al. (2025a) further rank and filter the 285 samples by combining their rewards and responses length during the preference learning process.

286 In RLVR, where rewards are sparse and only for the outcome, filtering is also helpful. For instance, 287 the simple rejection sampling fine-tuning (Dong et al., 2023; Chen et al., 2025a), which discards 288 all incorrect trajectories, often approaches the performance of more complex algorithms like GRPO 289 (Dong et al., 2023; Chen et al., 2025a; Xiong et al., 2025a). Other methods like (Yang et al., 2024) 290 filter prompts by difficulty prior to the RL training. Yu et al. (2025b) removes prompts that yield zero 291 gradients during training and dynamically regenerates samples. This technique is known as dynamic 292 sampling and has been rather widely accepted. Xiong et al. (2025a) demonstrates that prompts where 293 all generated responses are incorrect can significantly hurt the performance of the vanilla Reinforce 294 algorithm. They propose an online data filtering strategy based on the correctness reward, showing 295 that a modified Reinforce with filtering (Reinforce-rej) can match or exceed GRPO’s performance. 296 Their results suggest that the advantage of GRPO compared to Reinforce is due to the implicit data 297 filtering mechanism from the reward shaping. Finally, Xu et al. (2025) proposes to over-sample and 298 keep a subset such that the variance of the rewards in the subset is maximized, which implies that 299 they try to balance the ratio of correct and incorrect responses for reasoning tasks.

300 In contrast to these methods, which primarily rely on coarse, outcome-based metrics (e.g., final 301 answer correctness, trajectory-level rewards), our approach introduces a more fine-grained filtering 302 mechanism. We leverage process-supervised reward models (PRMs) (Lightman et al., 2023) to 303 evaluate and filter based on the quality of intermediate reasoning steps, and their consistency with 304 ORMs.

305 **Process-Supervised Reward Models for Fine-Grained Feedback.** The RLHF focuses on the 306 *trajectory-level comparison* under the Bradley-Terry model. For reasoning-related task, Yang et al. 307 (2024) uses the correctness of the final answer to construct the preference pairs and trains Bradley- 308 Terry reward models for mathematical reasoning. A more widely used approach, termed Outcome 309 Reward Models (ORMs) trains a classifier to predict whether the final answer is correct or not based 310 on the reasoning history. However, Lightman et al. (2023) have shown that Process-Supervised 311 Reward Models (PRMs), which evaluate each intermediate step of a reasoning chain, significantly 312 outperform ORMs, especially for data selection tasks like best-of-n sampling (Lightman et al., 2023). 313 But their approach requires human annotators to label each intermediate steps of the reasoning. Wang 314 et al. (2023) proposes to use Monte-Carlo estimation of the Q value to automatically decide the 315 label. After this, a long line of works proposes to improve the PRMs by generative reward modeling, 316 advanced training technique like RL, and refined engineering practices (Xiong et al., 2024b; Zhang 317 et al., 2025; Khalifa et al., 2025; Zhao et al., 2025; Xiong et al., 2025c). Our work does not focus on 318 improving PRMs but uses the PRMs to supervise the intermediate steps of CoT trajectories for data 319 filtering. We mainly use the Qwen2.5-Math-PRM-7B from Zhang et al. (2025) as it is trained on the 320 distribution of Qwen model and achieves superior performance on ProcessBench (Zheng et al., 2024).

321 **B Additional Experimental Details and Results**

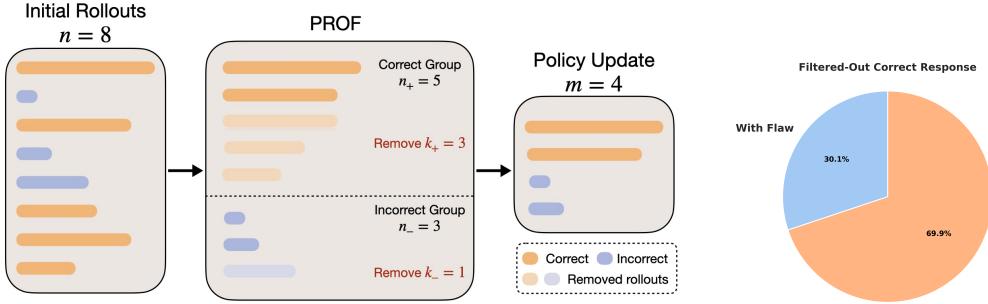


Figure 2: Left: Visualization of PROF Algorithm 1, where the length of each rectangle represents values of process rewards averaged over steps for each rollout. After generating n rollouts and process rewards, PROF ranks the correct and incorrect group separately according to PRM-ORM consistency, so for the correct group, the longer items are kept; for the incorrect group, the shorter items are kept. The number to remove is to balance correct and incorrect ratio. Right: Fraction of flawed-reasoning responses judged by LLM among the filtered-out correct responses.

Algorithm 1 Process Consistency Filter (PROF)

- 1: **Input:** Number of rollouts n , policy update size m , rollout $\{a_1, \dots, a_n\}$, outcome rewards $\{r_{o,1}, \dots, r_{o,n}\}$, step number regularization parameter $\lambda, H_\lambda > 0$.
- 2: Obtain process rewards for each rollout a_i with H_i steps: $(r_i^1, \dots, r_i^{H_i})$ and compute trajectory-wise consistency

$$r_i^{\text{pro}} = \left[\frac{1}{H_i} \sum_{h=1}^{H_i} r_i^h - \lambda I(H_i = 1 \text{ or } H_i \geq H_\lambda) \right] \cdot r_{o,i}. \quad (1)$$

- 3: Divide rollouts into correct group $\mathcal{G}_+ = \{a_1^+, \dots, a_{n+}^+\}$ with $r_{o,i} = 1$ and incorrect group $\mathcal{G}_- = \{a_1^-, \dots, a_{n-}^-\}$ with $r_{o,i} = -1$, where $n_+ + n_- = n$.
- 4: Compute kept number $k_+ \in [n_+], k_- \in [n_-]$ in each group such that $K_+ + k_- = m$ and $k_+ k_-$ is maximized.
- 5: Rank \mathcal{G}_+ and \mathcal{G}_- by r^{pro} separately, and keep the samples

$$\mathcal{K}^+ = \{a_i^+ | \text{rank}(a_i^+) \geq n_+ - k_+\}, \quad \mathcal{K}^- = \{a_i^- | \text{rank}(a_i^-) \geq n_- - k_-\}.$$

- 6: **Output:** The kept trajectories $\mathcal{K}^+ \cup \mathcal{K}^-$ with final kept size m .

322 **B.1 Main Experiments**

323 The implementations are based on the verl framework (Sheng et al., 2025), and we follow most
324 of the parameter settings in verl. Detailedly, we apply the AdamW optimizer with learning rate
325 1×10^{-6} . We adopt the clip higher trick (Yu et al., 2025b) that clips the sampling ratio $\pi_\theta / \pi_{\text{old}}$ to an
326 asymmetric range $(1 - \epsilon_{\text{low}}, 1 + \epsilon_{\text{high}})$. Specifically, we set $\epsilon_{\text{low}} = 0.2, \epsilon_{\text{high}} = 0.28$ for models started
327 from Qwen2.5-Math-1.5B-base and maintain $\epsilon_{\text{high}} = \epsilon_{\text{low}} = 0.2$ for other cases. In each iteration,
328 we sample 1024 prompts, rollout $n = 4$ responses per prompt for GRPO and $n = 8$ responses for
329 PROF-GRPO. Note that the policy update number for all algorithms is $m = 4$. For the regularization
330 of step numbers in Algorithm 1, we take $\lambda = 10$ and $H_\lambda = 30$. For the rollout stage, we use a
331 temperature of 1.0 and a top-p value of 1.0. We set the KL loss coefficient to 0.001 and entropy loss
332 coefficient to 0.001. All the models are trained with 8 H100 GPUs. We set the training mini-batch
333 size as 256 and allow the models to generate 4096 tokens per prompt.

334 **B.2 Prompt Template**

335 We present the template used for LLM to compare step-level reasoning.

Prompt for Finding Reasoning Flaws in Correct Response via LLM-as-a-judge

Here is the problem and the assistant's solution, which has been broken down into {step} steps.

Problem:

Assistant's Solution:

Your task is to review each step of the solution in sequence, analyzing, verifying, and critiquing the reasoning in detail. You need to provide the analyses and the conclusion in the following format:

```
<step>Step 1 Analysis</step>
<step>Step 2 Analysis</step>
... [CONTINUE FOR ALL step steps in the Assistant's Solution] ...
<conclusion>Correct/Incorrect</conclusion>
```

- When you analyze each step, you should use proper verification, recalculation, or reflection to indicate whether it is logically and mathematically valid. Please elaborate on the analysis process carefully.
- If an error is detected in any step, you should describe the nature and cause of the error in detail, and suggest how to correct the error or the correct approach. Once a step is found to contain any error, stop further analysis of subsequent steps (as they may depend on the identified error) and directly provide the conclusion of "Incorrect."

For instance, given a solution of five steps, if an error or flaw is found in the third step, you should reply in the following format:

```
<step>Step 1 Analysis</step>
<step>Step 2 Analysis</step>
<step>Step 3 Analysis; since an error or flaw is found here, also provide detailed critique and correction guideline)</step>
<conclusion>Incorrect</conclusion>
```

Respond with your analyses and conclusion directly.

336

Prompt for Responses Comparison via LLM-as-a-judge

System You are a meticulous, comparison engine. Your ONLY function is to compare the intermediate reasoning steps of the two responses provided to you.

User Here is the problem and assistants' two solutions, which have been chunked into steps. You MUST provide preference over the two solutions.

Problem: <prompt>

Assistant's Solution 1: <solution1>

Assistant's Solution 2: <solution2>

Both solutions are correct. You MUST compare them based on the following criteria:

- The reasoning process is more correct, and logical.
- The reasoning process does not skip any reasoning steps.
- The reasoning process does not skip any reasoning steps.

You MUST follow this exact format:

Your detailed verification reasoning goes here. Conclude with the number of the preferred solution: 1 or 2.

If you prefer solution 1, you MUST output 1.

If you prefer solution 2, you MUST output 2.

Your preference:

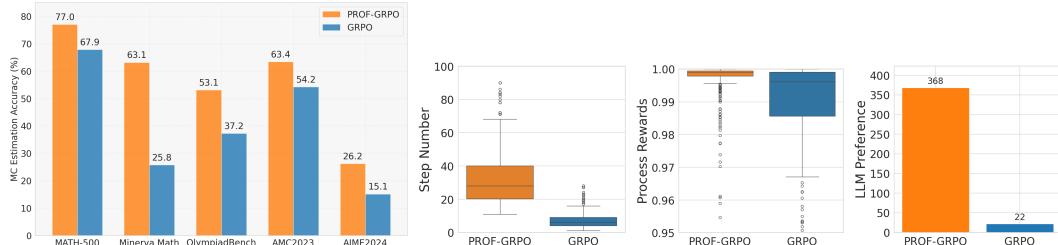
337

338 **C Additional Experiment Results**

339 **C.1 How PROF shapes Intermediate Reasoning Steps**

340 **Effectiveness of Consistency Filtration.** To demonstrate that our algorithm effectively differentiates the inconsistent trajectories, especially those correct answers with flawed reasoning steps, we 341 prompt Qwen2.5-Math-7B-base (Yang et al., 2024) to generate rollouts for 500 problems randomly 342 selected from the training set, and implement the filtration in Algorithm 1. Then, the filtered-out 343 correct responses are judged by Claude-3-7-sonnet from Anthropic to verify whether they contain 344 flawed steps. We use the prompt in Zhang et al. (2025) and provide the details in Appendix B. 345 From Figure 2, 30.1% responses among the filtered-out correct responses are judged to possess 346 flawed reasoning. This indicates that our methods can efficiently distinguish a number of flawed 347 responses and reach consensus with LLM. Furthermore, with human checking those filtered-out 348 correct responses, there are many responses with invalid or even completely wrong reasoning steps 349 but luckily reaching the correct answer. A typical example is presented in Table 5. However, such 350 flawed reasoning processes would be entirely missed by a standard ORM. 351

352 **Improved Step-wise Value.** To evaluate the quality of intermediate steps, we adopt Monte Carlo 353 (MC) estimation, a common way to estimate probability of getting to correct final answers (Wang 354 et al., 2023; Xiong et al., 2024a; Luo et al., 2024). For this analysis, we select problem-response 355 pairs from the test prompts where our method (PROF-GRPO) and GRPO both produced the correct 356 final answer. Both models were initialized from Qwen2.5-Math-7B-base. To estimate the value of 357 each reasoning step, we generate eight independent completions from that point using a temperature 358 of 1.0, and the resulting empirical success rate serves as the MC value. Our primary finding is that 359 PROF-GRPO achieves significant improvement in step-wise values compared to GRPO. In Figure 3, 360 the average MC estimations across all five benchmarks are consistently higher for our model. The 361 specific improvement gaps are 9.2% on Math500, 37.4% on Minerv Math, 15.9% on Olympiad 362 Bench, 9.2% on AMC2023, and 11.1% on AIME2024, which are much larger than the outcome 363 accuracy gap in Table 1. Hence, in addition to improving the outcome accuracy, our PROF method 364 substantially improves the quality and consistency of intermediate steps.



365 Figure 3: Reasoning intermediate-steps performance of PROF-GRPO in comparison with GRPO. 366 The most left plot is the Monte Carlo (MC) estimation scores across five benchmarks. The other three 367 are on Math500 under metrics of number of steps (2nd left), the averaged process rewards generated 368 by Qwen2.5-Math-PRM-7B (3rd left), and LLM’s preference between two modes’ responses (most 369 right). 370

371

372 **Deeper Analysis on Math500.** We further compare responses where both models were correct on 373 Math500 in Figure 3. In the second left figure, PROF-GRPO exhibits more reasoning steps. In the 374 third left figure, the PRM used for training assigns higher rewards for PROF-GRPO’s responses. In 375 the rightmost figure, we use Claude to judge which one’s reasoning process has more complete and 376 detailed steps, and PROF-GRPO’s responses are significantly preferred. The prompt for LLM-as-a- 377 judge is presented in Table B.2. The key takeaway is that our PROF method reshapes the model’s CoT 378 process from unfaithful reasoning into **detailed and easy-to-verify** steps. This is further validated by 379 two examples in Figure 7, 8. 380

373 **D Ablations**

374 **D.1 Separation of Correct and Incorrect Group**

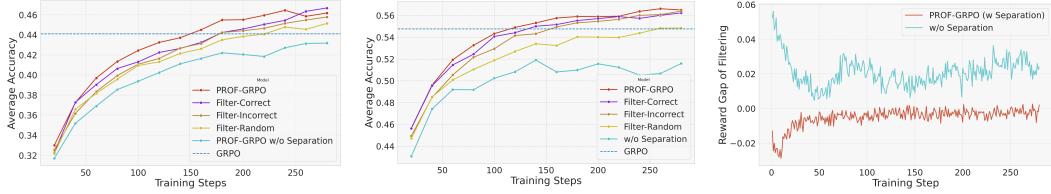


Figure 4: Left two: averaged accuracy over Math500, Minerva Math and Olympiad Bench for PROF-GRPO and its variants initialized from Qwen2.5-Math-1.5B-base and Qwen2.5-Math-7B-base. Most right: the gap between the training rewards after and before the filtering for PROF-GRPO in comparison with not separating correct and incorrect groups (w/o separation).

375 We conduct an ablation experiment on the necessity of separating correct and incorrect samples,
 376 named as PROF-GRPO w/o separation, where the rollouts are ranked and filtered together. To
 377 mitigate bias in PRM, each step’s PRM is subtracted by the averaged PRM of the batch. Even after
 378 centering, the rightmost plot in Figure 4 shows that PROF-GRPO w/o Separation has over 2% gap
 379 between the training reward after and before the filtration. This indicates that a disproportionate
 380 number of negative samples are removed. One explanation is that incorrect responses often contain
 381 several correct intermediate steps, thus increasing the averaged PRM over steps and leading to lower
 382 consistency. Consequently, incorrect responses exhibit lower consistency than correct ones, especially
 383 as the policy model improves over training. In contrast, PROF-GRPO successfully balances the bias
 384 by separating the correct and incorrect groups.

385 To further disentangle the contributions of filtering correct versus incorrect samples, we design the
 386 following variants of PROF: (1) Filter-Correct: use PRM consistency to filter the correct group
 387 and randomly filter the incorrect group; (2) Filter-Incorrect: only use PRM consistency to filter
 388 the incorrect group; (3) Filter-Random: randomly filter both correct and incorrect samples Xu
 389 et al. (2025). In Figure 4, Filter-Correct and PROF-GRPO (Filter-both) achieve comparably best
 390 performances among the variants across the 1.5B and 7B models. While Filter-both converges
 391 more efficiently because it leverages the consistency filtration for both correct and incorrect groups.
 392 Filter-incorrect is less efficient and has slightly poorer performance. In contrast, Filter-Random only
 393 performs slightly better than GRPO, and w/o Separation performs the worst.

394 We find that separating the correct and incorrect groups is essential to prevent the over-removal
 395 of valuable incorrect samples during training. While both Filter-both and Filter-Correct are top-
 396 performing strategies, with the former being more efficient, the trade-offs between them will be
 397 discussed in the following section. Furthermore, the comparable performance of Filter-both and Filter-
 398 Correct indicates that the process quality for correct samples is more crucial than the consistency for
 399 incorrect samples during the training process.

400 **D.2 Ablation Study on Base Model**

Algorithm	Math500	Minerva Math	Olympiad Bench	AIME24	AMC23	Average
Base	30.0	8.8	6.1	2.3	10.6	11.6
GRPO	50.5	18.8	17.9	5.0	25.6	23.6
Blend-PRM-GRPO	37.2	13.1	9.9	1.0	17.2	15.7
PROF-GRPO (Both)	50.4	19.1	18.7	3.5	27.8	23.9
PROF-GRPO (Correct)	52.4	19.5	19.8	6.7	28.6	25.4
PROF-GRPO (Incorrect)	49.0	18.0	17.3	5.4	23.9	22.7

Table 2: The test accuracy of different methods initialized from LLaMA-3.2-3B-instruct that is average@16 under temperature 1.0 and further averaged across all the five benchmarks.

401 To showcase the generalization of our algorithm, we conduct experiments on LLaMA-3.2-3B-instruct
 402 (Dubey et al., 2024) that has weaker math-reasoning abilities and more distribution shift since
 403 Qwen2.5-Math-PRM-7B is trained on the distribution of Qwen’s family. As provided in Table 2,

404 PROF-GRPO with PRM consistency filtering both correct and incorrect groups (Both) achieves
 405 23.9%, marginally outperforming the GRPO baseline (23.6%), while only applying PRM consistency
 406 to filter the correct group (Correct) exhibits the strongest (25.4%) performance. Conversely, applying
 407 the filter solely to the incorrect group (PROF-GRPO (Incorrect)) is counterproductive, causing
 408 accuracy to drop to 22.7%. Blend-PRM-GRPO still scores the worst (15.7%) among all the methods.
 409 These results suggest that our PROF methods can consistently outperform baselines across various
 410 base models.

411 For the trade-off between the Both and Correct, we conclude that when the PRM is less reliable or
 412 prone to reward hacking (as in this cross-model scenario), the “Correct” method offers more robust
 413 improvements by safely constraining the PRM’s influence. However, when the PRM is highly reliable
 414 and training efficiency is a priority, the “Both” method is recommended. Due to the space limit, more
 415 ablations such as rollout numbers and various filtration methods are provided in Appendix D.

416 D.3 Effect of Rollout Numbers

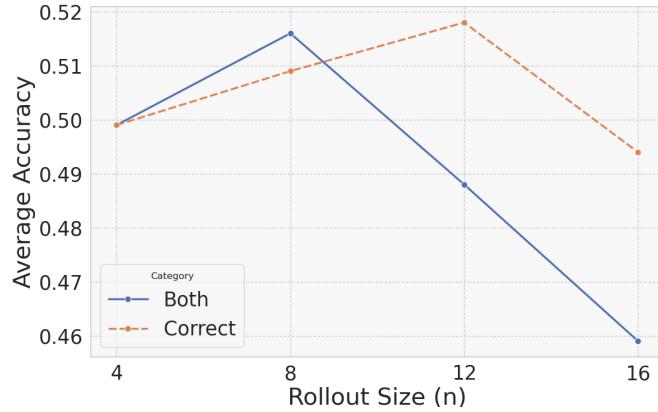


Figure 5: The averaged accuracy across all five benchmarks over rollout sizes $n = 4, 8, 12, 16$ for filtering both correct and incorrect groups with PRM consistency (Both) and only the correct group with PRM consistency (Correct).

417 We study the scale of rollout numbers n with fixed policy-update number $m = 4$ by varying
 418 $n = 4, 8, 12, 16$. The lower-right plot in Figure 5 presents the test accuracy averaged over all five
 419 benchmarks for PROF-GRPO (Both) and Filter-Correct (Correct) started from Qwen2.5-Math-7B-
 420 base. We observe the performance first increases then decreases as n increases, revealing a trade-off
 421 between enhancing process reasoning quality and avoiding reward hacking. Notably, Filter-Correct
 422 decreases later (after $n = 12$) because it only leverages the influence of PRM only in the correct
 423 group, indicating that Filter-Correct is more robust when the PRM’s influence is higher, like when
 424 increasing the scale of ranking and filtering.

425 D.4 Variants of Filtration Methods

Algorithm	Math500	Minerva Math	Olympiad Bench	AIME24	AMC23	Average
Mean	83.1	39.0	47.8	17.5	70.9	51.7
Minimum	82.9	38.3	46.7	20.8	65.9	50.9
Sum	82.4	38.1	47.4	17.7	67.5	50.6
Ratio	81.4	36.6	45.0	24.8	65.2	50.6

Table 3: Performance of different filtration ways in PROF starting from Qwen2.5-Math-7B-base.

426 In this subsection, we investigate the influence of different computation methods of consistency
 427 score r^{PRO} in addition to the mean of PRMs over steps, where Mean denotes averaging over steps
 428 in Algorithm 1, Minimum and Sum denotes taking the minimum and sum summation over steps,
 429 Ratio denotes filtering while preserving the original positive–negative sample distribution, instead

430 of balancing. As shown in Table 3, the performances of Minimum (50.9%), Sum (50.6%), and
 431 Ratio (50.6%) are inferior to the mean. This suggests that the mean provides a more stable estimate
 432 of reasoning consistency: unlike the minimum, it is less sensitive to a single poorly scored step,
 433 and unlike the summation, it avoids bias towards longer trajectories. Additionally, balancing the
 434 correct-incorrect ratio can use data consistency to select the group with more sufficient samples
 435 without breaking their balance.

436 **D.5 Effect of Step Number**

437 To prove that PROF effect not by simply increasing the step number, We conduct the Filter-Nstep:
 438 Ranking and filtering out the samples with smaller number of steps instead of lower PRM-ORM
 439 consistency.

440 From Table 4, we find that Ratio scores 51.7% on average and cannot compete with balancing
 441 the proportion (PROF-GRPO), which also corroborates the conclusion that maintaining a balanced
 442 correct-incorrect proportion is essential. Additionally, since we observe that PROF boosts the number
 443 of intermediate reasoning steps, to verify that PROF does not simply increase the step length, but
 444 more importantly, enhances the quality of reasoning steps, we simply use the step length as the
 445 filtering criterion for comparison. As shown in Figure 6 and Table 4, Filter-Nstep manipulates the
 446 step length, which exhibits an unreasonable increase followed by a sudden drop, and its average
 447 accuracy is inferior.

Algorithm	Math500	Minerva Math	Olympiad Bench	AIME24	AMC23	Average
PROF-GRPO	83.1	39.0	47.8	17.5	70.9	51.7
Filter-Nstep	81.5	35.5	45.9	16.3	58.6	47.6

Table 4: Performance of variants of filtering methods besides PROF-GRPO starting from Qwen2.5-
 Math-7B-base averaged on all five benchmarks, where Ratio denotes filtering while preserving the
 original correct/incorrect proportion, Filter-Nstep denotes ranking and filtering based on the number
 of step segments.

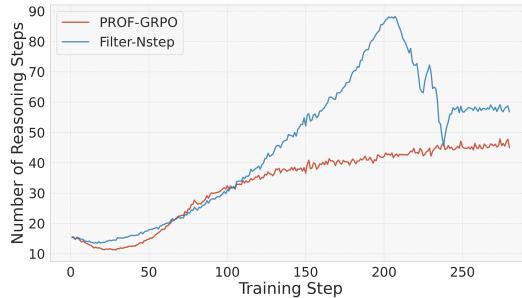


Figure 6: The number of reasoning steps during training time for PROF-GRPO and Filter-Nstep initialized from Qwen2.5-Math-7B-base.

448 **E Additional Examples**

Problem: There are 4 coins of 1, 2, 3, 5 cruzeiros, which weigh 1, 2, 3, 5 grams respectively. One of them is counterfeit, differing in weight (but not known whether it is heavier or lighter). How can the counterfeit coin be identified with the minimum number of weighings using a balance scale without weights?

Step 1: Introduction. We need to identify a counterfeit coin with an unknown weight difference from ...

Step 2 - 6: The Flawed Weighing Logic and Code.

1. Weigh coins of **1 gram and 2 grams** against coins of **3 grams and 5 grams**. 2. Based on the result of the first weighing: - If the left side is heavier, then the counterfeit coin is either 1 gram or 3 grams. - If the right side is heavier, then the counterfeit coin is either 2 gram or 5 grams. - If both sides are equal, then the counterfeit coin is either 2 gram or 5 grams. 3. For the second weighing, ...

Let's illustrate this with code and ensure that the steps are correct ... output: (1, 'heavier').

Step 7, 8, 9: Summary and Final Answer.

Here is a summary of the steps: 1. Weigh coins of 1 gram and 2 grams against coins of 3 grams and 5 grams...

2. For the second weighing, weigh the 1 gram coin against a known genuine coin (3 grams)...

Thus, the minimum number of weighings required to identify the counterfeit coin is 2.

Table 5: An Example of Reaching a Correct Result Through Flawed Reasoning. The proposed first weighing of {1g, 2g} (total 3g) against {3g, 5g} (total 8g) is fundamentally **invalid**. A balance scale requires comparing groups of equal nominal weight. Because this weighing is unbalanced, all conclusions drawn from it are baseless. The final answer is correct but is completely unsupported by the fallacious reasoning.

PROF-GRPO Solution:	GRPO Solution:	Blend-PRM-GRPO Solution:
<p>First, let's recall the formula for the luminosity of a blackbody.</p> <p>The luminosity L of a blackbody is given by the Stefan-Boltzmann law: $L = 4\pi R^2 \sigma T_e^4$</p> <p>We are given: $T_e = 50,000\text{K}$... The Stefan-Boltzmann constant σ is approximately $5.67 \times 10^{-8}\text{Wm}^{-2}\text{K}^{-4}$.</p> <p>Next, let's substitute the values into the Stefan-Boltzmann law.</p> $L = 4\pi(10^7)^2 \times 5.67 \times 10^{-8} \times (50000\text{K})^4$ <p>Let's calculate the value step by step.</p> $(10^7)^2 = 10^{14}, (50000\text{K})^4 = \dots = 6.25 \times 10^{18}$ <p>Now, substitute these values back into the equation..., calculate the numerical part</p> $4\pi \approx 4 \times 3.14159 = 12.56636,$ $12.56636 \times 5.67 \approx 71.148$ $71.148 \times 6.25 \approx 444.675$ $L \approx 4.45 \times 10^{26} \text{W}$ <p>Since 1 W=1 J/s and 1 erg=10⁻⁷ J, ...</p> $L \approx 4.45 \times 10^{33} \text{erg/s}$ <p>Therefore, the luminosity of the white dwarf is:</p> $4.5 \times 10^{33} \text{erg/s}$	<p>To compute the luminosity of the white dwarf, we can use the Stefan-Boltzmann law ...</p> <p>The formula for the luminosity SLs of a blackbody is given by: $L = 4\pi R^2 \sigma T_e^4$, where $\sigma = 5.67 \times 10^{-8}\text{Wm}^{-2}\text{K}^{-4}$.</p> <p>Given: $T_e = 5 \times 10^4\text{K}$, $R_{WD} = 10^9\text{cm}$.</p> <p>First, we need to calculate $T_e^4 = (5 \times 10^4)^4 = 625 \times 10^{16} = 6.25 \times 10^{18}\text{K}^4$</p> <p>Next, we calculate the surface area of the white dwarf:</p> $4\pi R_{WD}^2 = 4\pi(10^9\text{cm})^2 = 4\pi \times 10^{18}\text{cm}^2$ <p>Now we can compute the luminosity:</p> $L = 4\pi \times 10^{18}\text{cm}^2 \times 5.67 \times 10^{-8}\text{ergs}^{-1}\text{cm}^{-2}\text{K}^{-4} \times 6.25 \times 10^{18}\text{K}^4$ $L = 4\pi \times 5.67 \times 6.25 \times 10^{18} \times 10^{-8} \times 10^{18} \times 10^{18} \text{ergs}^{-1}$ $= \dots = 4.42 \times 10^{33} \text{ergs}^{-1}$ <p>Rounded to two significant figures, the luminosity of the white dwarf is</p> $4.4 \times 10^{33} \text{ergs}^{-1}$	<p>The luminosity L of a blackbody can be calculated using the Stefan-Boltzmann law: $L = 4\pi R^2 \sigma T_e^4$, where σ is approximately $5.67 \times 10^{-8}\text{ergs per square centimeter per second per Kelvin to the fourth power}$.</p> <p>$R_{WD} = 10^9\text{cm}$.</p> <p>First, we need to convert the radius from kilometers to centimeters.</p> <p>First, we calculate $R_{WD}^2 = (10^9\text{cm})^2 = 10^{18}\text{cm}^2$</p> <p>Next, we calculate... $T_e^4 = 6.25 \times 10^{18}\text{K}^4$</p> <p>Now we can substitute these values into the Stefan-Boltzmann law:</p> $L = 4\pi \times 10^9\text{cm} \times 5.67 \times 10^{-8}\text{ergs/cm}^2/\text{s}/\text{K}^4 \times 6.25 \times 10^{18}\text{K}^4$ <p>... We can separate the constants and the powers of 10:</p> $4\pi \times 5.67 \times 6.25 \times 10^{-5+18}$ <p>So, 444.51875×10^{13}...</p> <p>Therefore, $L \approx 4.45 \times 10^{15} \text{ergs/s}$</p> <p>So the luminosity of the white dwarf is 4.45 \times 10^{15} \text{ergs/s}</p>

Figure 7: A Minerval-Math example to compare distinct intermediate reasoning patterns of PROF-GRPO, vanilla GRPO and Blend-PRM-GRPO. PROF-GRPO presents concrete and correct deduction steps. GRPO's solution skips detailed deduction steps and there are flaws in the calculation precision and the final rounding. Blend-PRM-GRPO has long-winded steps and makes a big mistake in calculating the power.

Problem: The proper divisors of 12 are 1, 2, 3, 4 and 6. A proper divisor of an integer N is a positive divisor of N that is less than N . What is the sum of the proper divisors of the sum of the proper divisors of 284?

PROF-GRPO Solution:

First, we need to find the proper divisors of 284 ...
 Let's start by finding the divisors of 284. We can start by checking the obvious small divisors.
 Let's check if 2 is a divisor of 284.
 $284 \div 2 = 142$
 Since the result is an integer, 2 is a divisor.
 Next, let's check if 4 is a divisor of 284...
 The proper divisors of 284 are 1, 2, 4, 71, and 142.
 Next, we need to find the sum of these proper divisors. Let's add these step by step
 ...
 $1 + 2 + 4 + 71 + 142 = 220$
 Now, we need to find the proper divisors of 220 ...
 The proper divisors of 220 are 1, 2, 4, 5, 10, 11, 20, 22, 44, 55, and 110. The sum is:
 $1 + 2 + \dots + 110 = 284$
 Therefore, the sum of the proper divisors of the sum of the proper divisors of 284 is 284 .

GRPO Solution:

First, we need to find the proper divisors of 284. The proper divisors of 284 are the positive divisors of 284 that are less than 284. The divisors of 284 are 1, 2, 4, 71, and 142. The sum of these proper divisors is:
 $1 + 2 + 4 + 71 + 142 = 220$
 Next, we need to find the proper divisors of 220. The divisors of 220 are 1, 2, 4, 5, 10, 11, 20, 22, 44, 55, and 110. The sum of these proper divisors is:
 $1 + 2 + \dots + 110 = \boxed{284}$
 So the sum of the proper divisors of the sum of the proper divisors of 284 is 284.

Blend-PRM-GRPO Solution:

First, we need to find the proper divisors of 284. Let's start by finding all the divisors of 284. We'll start with 1, which is always a divisor. We check if 2 is a divisor.
 $284 \div 2 = 142$
 So, 2 is a divisor.
 Next, we check if 3 is a divisor.
 $284 \div 3 \approx 94.67$
 So, 3 is not a divisor ...
 We continue this process until we reach 142. The divisors of 284 are 1, 2, 4, 71, 142.
 Now, we need to find the sum of these proper divisors: $1 + 2 + 4 + 71 + 142$
 First, we add 1 and 2: $1 + 2 = 3$
 So, the sum of the proper divisors of 284 is 220.
 Now, we need to find the proper divisors of 220. We'll check each number from 1 to 219 to see if it divides 220 without leaving a remainder ...
 The proper divisors of 220 are 1, 2, 4, 5, 10, 11, 20, 22, 44, 55, and 110....
 So, the sum of the proper divisors of 220 is $\boxed{284}$.

Figure 8: A Math500 example to compare distinct intermediate reasoning patterns of PROF-GRPO, vanilla GRPO and Blend-PRM-GRPO. PROF-GRPO presents concrete and correct deduction steps. PROF-GRPO's solution shows how to find the divisors and summation in detail, and is easy to follow. GRPO skips all core reasoning. Blend-PRM-GRPO has inefficient and excessively tedious steps.