THE HIDDEN BIAS OF PROCESS REWARD MODELS: PRISM FOR REWARDING THE RIGHT REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Process Reward Models (PRMs) improve credit assignment for reasoning by providing step-level feedback. However, we identify a hidden bias in PRMs caused by severe imbalance in step-level training data. Standard cross-entropy training amplifies this bias, causing PRMs to overcredit plausible but incorrect steps and produce high false-positive rates. We show that these false positives have an asymmetric downstream effect: false negatives mainly slow exploration, whereas false positives actively steer Best-of-N selection, guided decoding, and policy optimization toward flawed reasoning. This suggests that PRM training should shift from pointwise label fitting to reliable relative comparisons. To address this, we propose PRISM (Precision Ranking for Improved Step Modeling), a policy-aware PRM training framework that learns from contrastive step-level comparisons and hard negatives generated by a temporal lookahead strategy, requiring no new human labels. We further use a difficulty-aware curriculum to optimize the contrastive step margin. Across PRMBench (Song et al., 2025) and ProcessBench (Zheng et al., 2025), PRISM substantially reduces false positives (↓ 22% on PRMBench) and improves macro F1 over strong discriminative PRMs. When applied to policy optimization and search tasks, including guided decoding and Best-of-N selection, it consistently improves accuracy (up to 22% for guided decoding and 33% for Best-of-N) and robustness. More broadly, trustworthy process supervision is not just about assigning high rewards, but about rewarding the right reasoning for the right reasons.

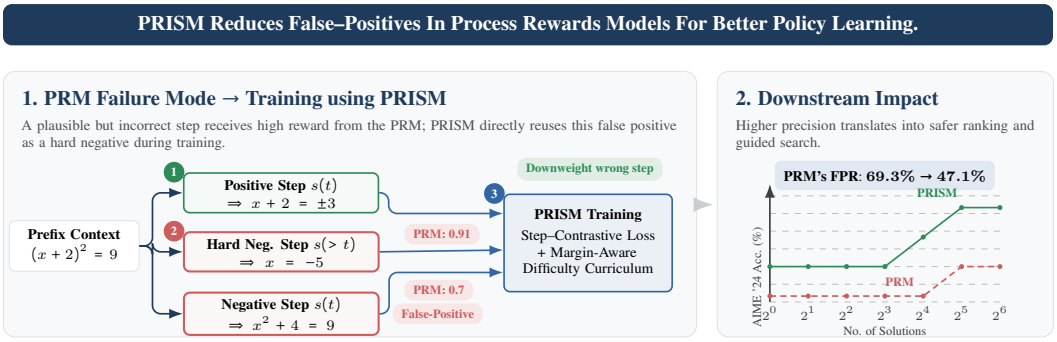


Figure 1: Baseline PRMs can overcredit plausible but incorrect steps, a form of hidden PRM bias that downstream search and policy learning can exploit. PRISM **(1)** converts existing step annotations into matched comparisons, **(2)** augments them with temporal-lookahead hard negatives using future steps $s(> t)$ (hard negatives are more confusing but helps in learning better reasoning), and **(3)** trains with a Step-Contrastive objective plus a simple curriculum. The result is substantially lower FPR on PRMBench($69.3\% \rightarrow 47.1\%$) and more reliable guided search, Best-of-N selection, and policy optimization.

1 INTRODUCTION

Process Reward Models (PRMs) (Wang et al., 2023; Zhang et al., 2025) have become a key ingredient in modern reasoning systems (Guo et al., 2025; Shao et al., 2024; Jaech et al., 2024) because they score intermediate reasoning steps rather than only final answers. This denser supervision improves credit assignment in long-horizon tasks (Lambert et al., 2024) and is especially valuable in mathematics (Lightman et al., 2023; Chen et al., 2024; Zhang et al., 2024), program synthesis, and planning, where final success depends on many locally correct decisions. Yet current PRMs exhibit a hidden and

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

consequential failure mode: across both discriminative and generative formulations (Zhang et al., 2025; Wang et al., 2023; Khalifa et al., 2025; Zhao et al., 2025), they often assign high scores to plausible but incorrect steps, producing elevated false-positive (FP) rates (Table 1). This failure mode is referred to here as *overcredit bias*.

Sources of overcredit bias. Overcredit bias is partly rooted in the structure of process-supervision data. Even incorrect steps frequently contain many locally correct steps before the first decisive error, so correct step labels can outnumber incorrect ones even within ultimately wrong trajectories (e.g., PRM800k; Figure 2) (Lightman et al., 2023). This skews the step-level training distribution toward positives. When PRMs are then trained with standard *pointwise* objectives such as cross-entropy on imbalanced and noisy step annotations, they tend to inherit the same bias, rewarding fluency and local plausibility more than actual correctness. The result is exactly the error pattern that matters most downstream: incorrect steps that receive undeservedly high scores.

Why false positives matter most. This failure mode is especially harmful because PRMs are not merely offline evaluators; they are used inside downstream policy learning and inference-time search. A false negative may discard a correct step and force the system to explore more. A false positive is worse: it can actively steer the policy onto an incorrect branch. In guided decoding, the model may keep expanding a flawed partial solution; in Best-of-N selection, it may choose a wrong but convincing trajectory; and in reinforcement learning, it may reinforce spurious reasoning patterns. False positives therefore do not remain local prediction errors—they become a direct mechanism for bias-induced policy misalignment (Figure 1).

A policy-aware view of PRM quality. These observations suggest that PRMs should be evaluated and trained through the lens of downstream use. The goal is not simply to maximize average step-level accuracy, but to learn scores that reliably rank correct continuations above incorrect yet plausible ones. From this perspective, false positives and false negatives are not symmetric: false negatives primarily reduce recall and slow search, whereas false positives can make incorrect reasoning appear rewarding and thereby mislead the policy itself. This leads to a central question: *can false positives be reduced without sacrificing the recall needed for exploration and search?* Answering it requires solving three practical challenges: **(C1)** learning an objective that optimizes relative ranking rather than pointwise label fit, **(C2)** constructing informative positive–negative comparisons from imbalanced step annotations, and **(C3)** stabilizing training when the most informative negatives are also the hardest.

The proposed method: PRISM. PRISM (Precision Ranking for Improved Step Modeling) is a label-efficient, architecture-agnostic training recipe that addresses these challenges without new human annotations. First, instead of pointwise classification, PRMs are trained with a *Step-Contrastive* objective that optimizes the relative score of correct versus incorrect steps. Second, *matched* positive–negative pairs are constructed from existing step annotations, and hard negatives are generated via *temporal lookahead*, treating future steps as negatives for earlier contexts to expose overcredit errors. Third, optimization is stabilized with a simple difficulty-aware curriculum. Together, these ingredients shift PRM learning from pointwise label fitting to calibrated *preference margins*, directly targeting the false-positive overcredit that undermines downstream alignment.

Empirical validation. On PRMBench (Song et al., 2025) and ProcessBench (Zheng et al., 2025), PRISM consistently lowers FPR and improves macro F1 over strong baselines. More importantly, when deployed in guided decoding, Best-of-N selection, and policy optimization tasks, the resulting PRMs deliver higher downstream accuracy and robustness. These gains narrow the gap between offline reward-model metrics and the behavior of the policies that use them.

Contributions. (1) Precision-first analysis. The analysis formalizes how step-level overcredit compounds across trajectories and proves a sharp asymmetry for Best-of-N: the false-positive rate determines an asymptotic alignment ceiling through precision, whereas false negatives primarily increase sample complexity. **(2) A comparison-based training recipe.** PRISM provides a label-efficient, architecture-agnostic method that converts existing step labels into matched comparisons, trains with a Step-Contrastive objective, and combines temporal-lookahead hard negatives with a simple curriculum. **(3) Downstream-aligned gains.** Results on PRMBench (Song et al., 2025), ProcessBench (Zheng et al., 2025), guided decoding, Best-of-N selection, and policy training show that reducing overcredit yields more reliable PRMs and stronger policy alignment without new human labels.

Model	Pos-Acc/TPR	Neg-Acc/TNR	PRMScore
Qwen-PRM-7B	95.36	30.66	65.5
ReasonEval-7B*	95.5	21.2	60.0
ThinkPRM-7B	83.29	50.89	64.3
GenPRM-7B	52.25	73.92	50.5
GPT-4o*	82.9	58.2	66.8

Table 1: Positive and negative step accuracy for SOTA PRMs, along with PRMScores from Song et al. (2025). Overall, these findings argue for *precision-first* process supervision: trustworthy reasoning systems require not just dense rewards, but rewards that distinguish genuinely correct reasoning from merely plausible traces.

2 Problem Formulation: PRM Bias and Its Impact on Policy Learning

Let x denote the prompt and $y = (y_1, y_2, \ldots, y_T)$ a reasoning chain with T steps. At step t, the prefix $y_{<t} = (y_1, \ldots, y_{t-1})$ is the partial reasoning trace and y_t is a candidate next step. A Process Reward Model (PRM) is a step-level verifier that gives scalar reward $r_\theta(x, y_{<t}, y_t)$, which measures how appropriate or correct the step y_t is in the context $(x, y_{<t}, y_t)$. This step-level score is then used by downstream policy optimization or search procedures as discussed later.

2.1 Key Observation: Hidden bias in PRMs

Data-Imbalance Problem. We observed a key label imbalance in commonly available open-source PRM training datasets such as PRM800K (Lightman et al., 2023). PRM800K contains a much higher proportion of correct steps (73.1%) than incorrect steps, even though only a small fraction of full trajectories are correct (14.2%) (Figure 2). There are two intuitive reasons behind this mismatch: (1) correct question–response pairs are easier to collect from open-source data, and (2) a trajectory is labeled incorrect as soon as even one step is wrong and a trajectory contains many steps. Thus, it can contain many locally correct steps while still being globally incorrect, making step-level labels far more skewed toward positives than trajectory-level labels.

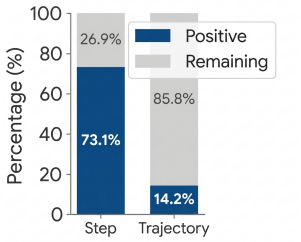


Figure 2: Step-level labels are substantially more skewed toward positives than trajectory-level labels in PRM800K.

Cross-Entropy Loss Exacerbates Bias. In practice, PRMs are typically trained with pointwise, step-level cross-entropy loss Zhang et al. (2025); Wang et al. (2023). Under label imbalance, such pointwise objectives are especially prone to bias. As Table 1 shows, existing models consistently over-predict correct steps. This stark false positive (FP) bias assigns spuriously high rewards to flawed reasoning, directly fueling downstream reward hacking.

PRM bias hurts downstream performance in two ways. In BoN and policy optimization, one overcredited step can raise the score of the entire response. In guided beam search, the effect is even stronger: an early false positive can push decoding onto a flawed path. In both cases, errors can accumulate over long chains, and the misalignment can grow with the number of steps, T.

2.2 How PRM Bias Distorts Downstream Policy Learning

PRMs are ultimately used to guide downstream policy optimization and search, so their quality should be evaluated through their downstream effect. Existing works focuse on standalone PRM accuracy, while overlooking its impact on policy learning and inference-time search. We therefore adopt a policy-aware view of PRM evaluation.

We illustrate how PRM bias affects downstream optimization through a standard KL-regularized RL objective. Without loss of generality, the same dependence also governs other downstream policy optimization (GRPO, PPO) and search tasks (BoN). The policy π is optimized via:

$$\max_{\pi} \mathbb{E}_{y_t \sim \pi(\cdot|x, y_{<t})}[r_\theta(x, y_{<t}, y_t)] - \beta \mathbb{D}_{\mathrm{KL}}(\pi(\cdot|x, y_{<t}) \,\|\, \pi_{\mathrm{ref}}(\cdot|x, y_{<t})) \tag{1}$$

The optimal policy π^* has the well-known closed-form solution:

$$\pi^*(y_t|x, y_{<t}) = \frac{\pi_{\mathrm{ref}}(y_t|x, y_{<t}) \exp(r_\theta(x, y_{<t}, y_t)/\beta)}{Z(x, y_{<t})} \tag{2}$$

where $Z(x, y_{<t})$ is the partition function. Equation 2 highlights a critical vulnerability: the exponentiated reward $\exp(r_\theta/\beta)$ makes the policy highly sensitive to overestimation. In particular, if a

flawed step receives a spuriously high reward r_θ (a false positive, or FP), its probability under π^* can become disproportionately large, making the optimizer learn incorrect information. Thus, reducing FPs is important to train a correct policy.

2.2.1 Analyzing Asymmetric effect of PRM's False Positive Rate on Policy Search

In this section, we study how PRM's false positives affect downstream policy search. In particular, we analyze the asymmetric effects of false positives and false negatives on Best-of-N performance.

Theorem 2.1 (Asymmetric Effect of False Positives (FPs) and False Negatives (FNs))**.** *Consider Best-of-N (BoN) selection using an imperfect binary reward model $\hat{T}(y) \in \{0,1\}$ to select from N i.i.d. samples $y_i \sim \pi(\cdot \mid x)$. Let $T(y) \in \{0,1\}$ be the true label, with base rate $p = \mathbb{P}(T = 1)$. Let $\alpha = \mathbb{P}(\hat{T} = 1 \mid T = 0)$ and $\beta = \mathbb{P}(\hat{T} = 0 \mid T = 1)$ denote the FPR and FNR, respectively. Let $P^{(N)}$ be the probability BoN selects a response where $T = 1$. Assuming N is sufficiently large and the marginal predicted-positive rate is non-vanishing, then:*

Asymmetry I: False positives induce a hard ceiling. If $\alpha > 0$ and $\beta = 0$, then:

$$\lim_{N\to\infty} P^{(N)} = \frac{p}{p + (1-p)\alpha} < 1 \tag{3}$$

This ceiling is strictly decreasing in α.

Asymmetry II: False negatives only slow convergence. If $\alpha = 0$ and $\beta < 1$, then:

$$\lim_{N\to\infty} P^{(N)} = 1 \tag{4}$$

In this case, FN reduce the rate at which $P^{(N)}$ approaches 1, but do not induce asymptotic bias.

Thus, FN merely delays convergence, whereas FP fundamentally caps final performance under BoN section. Therefore, objective is to prioritize minimizing α (reducing FPs) to lift the ceiling on alignment. Detailed proof is provided in appendix section C.1.

Key Insight: Reducing False Positives Improves Policy Learning. For PRMs, small overcredits can compound across long reasoning chains for guided decoding. In BoN and policy optimization, a single overcredited step can inflate the score of an incorrect response. As a result, a high FPR can misalign the policy and cause reward hacking. Objectives and data curation should therefore (i) rebalance step label distributions, and (ii) emphasize hard negatives, all aimed at minimizing α to lift the ceiling on alignment.

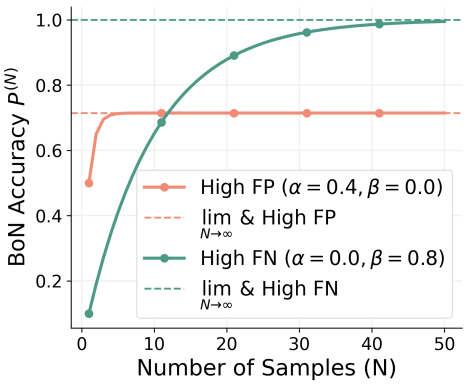


Figure 3: Effect of FP vs. FN on BoN Accuracy. FP induces a strict performance ceiling, while FN only slows the rate of convergence.

3 Policy Aware PRM Training to Mitigate PRM Bias

There have been relatively few works on training and improving PRMs. However, these methods do not explicitly focus on downstream policy learning, which is the main setting in which PRMs are ultimately used. They also rely on mean-square-error loss (Chen et al., 2024) or pointwise losses (Wang et al., 2023; Luo et al., 2024; Zhang et al., 2024) trained on human annotations (Lightman et al., 2023) or MCTS-generated targets (Zhang et al., 2024; Chen et al., 2024). As a result, the precision of step-level annotations becomes critically important, yet remains difficult to obtain. In this work, we argue that PRMs should be evaluated through their effect on downstream policy learning. We therefore propose PRISM (Precision Ranking for Improved Step Modeling), a policy-aware and label-efficient framework for PRM training that combines a Step-Contrastive (SC) loss, hard-negative augmentation using a temporal lookahead strategy that requires no new human labels, and a difficulty-aware curriculum.

We first motivate our policy-aware approach to mitigating PRM bias, then introduce the loss, data augmentation strategy, and curriculum design. Note that, the goal of this work is not to design a

perfect reward model but to highlight, and mitigate, systematic PRM failure modes that undermine alignment.

3.1 Step-Contrastive Loss

Here we motivate why we chose our proposed objective to train the PRM model followed by the exact details of our loss function. As discussed in section 2.2, FPs can drive the policy toward incorrect reasoning. However, the goal of policy learning is simply to ensure that π^* assigns higher probability to a correct step y_t^{pos} than to an incorrect step y_t^{neg}. From Equation 2, this relative preference can be written as:

$$\frac{\pi^*(y_t^{\text{pos}}|x, y_{<t})}{\pi^*(y_t^{\text{neg}}|x, y_{<t})} = \frac{\pi_{\text{ref}}(y_t^{\text{pos}}|x, y_{<t})}{\pi_{\text{ref}}(y_t^{\text{neg}}|x, y_{<t})} \exp\left(\frac{r_\theta(x, y_{<t}, y_t^{\text{pos}}) - r_\theta(x, y_{<t}, y_t^{\text{neg}})}{\beta}\right) \tag{5}$$

Therefore, as long as $r_\theta(x, y_{<t}, y_t^{\text{pos}}) > r_\theta(x, y_{<t}, y_t^{\text{neg}})$, the exponentiated reward gap pushes the policy toward the correct step. The model does not need to separate the two steps by an arbitrarily large absolute amount; it only needs to rank them correctly. Standard binary cross-entropy (BCE), however, trains each step independently and does not optimize this relative comparison. Under label imbalance, it also becomes harder to keep hard negative steps sufficiently low-scoring. This motivates directly optimizing the margin which can be written as

$$m_\theta(x, y_{<t}) = r_\theta(x, y_{<t}, y_t^{\text{pos}}) - r_\theta(x, y_{<t}, y_t^{\text{neg}}).$$

Thus, improving downstream policy learning reduces to increasing m_θ. This directly motivates our step-contrastive loss which optimizes this margin. We introduce it below,

How we get step-scores. Conventional PRM output reward, $r_\theta(\cdot) \in \mathbb{R}^{2\times 1}$, corresponding to scores for positive and negative labels. We use reward score with respect to positive class and call it $r_\theta(.)$.

Step-Contrastive (SC) loss. We optimize a preference based objective, inspired from pairwise preference loss Ouyang et al. (2022), that pushes the positive above the negative:

$$\mathcal{L}_{\text{SC}}(\theta) = -\mathbb{E}_{(x, y_t^{\text{pos}}, y_t^{\text{neg}})\in\mathcal{D}}\Big[\log\big(\sigma(r_\theta(x, y_{<t}, y_t^{\text{pos}}) - r_\theta(x, y_{<t}, y_t^{\text{neg}}))\big)\Big], \tag{6}$$

where $t \le T$ can be any step uptil total steps T. Here $r_\theta(x, y_{<t}, y_t)$ is the PRM output for step y_t given problem x and prefix $y_{<t}$. Note, each pair $(y_t^{\text{pos}}, y_t^{\text{neg}})$ share the same prefix $(x, y_{<t})$.

Thus writing $\mathcal{L}_{\text{SC}}(\theta) = -\log\sigma(m_\theta)$ and taking its gradient we have,

$$\nabla_\theta\mathcal{L}_{\text{SC}}(\theta) = -\sigma(-m_\theta)\ \nabla_\theta\, m_\theta.$$

Since $\sigma(-m_\theta) > 0$, minimizing $\mathcal{L}_{\text{SC}}$ pushes θ in the direction that increases m_θ. In simple terms, the objective directly encourages the reward of a correct step to exceed that of an incorrect step.

What this buys us. Because $(x, y_{<t})$ is shared, the model learns a *relative* preference within the same context. The objective places the largest penalty when an incorrect step outranks, or nearly ties, a correct step—directly reducing overcredit on negatives. It is architecture agnostic and uses only existing step labels.

Why Step-Contrastive Loss Is Better Aligned Than Step-BCE For Policy Learning To show this, we conducted simulation by training two identical network architectures using CE loss optimized to predict absolute binary labels, and a BT loss optimized directly on pairwise preferences on the same matched dataset $\mathcal{D} = \{(x, y_{<t}, y_t^{\text{pos}}, y_t^{\text{neg}})\}$. We show pairwise classification accuracy for different absolute margin $|s(a) - s(b)| \in [0, 0.5]$. As shown in Figure 4, the BT model maintains significantly higher accuracy as the margin approaches zero. This is because BCE optimizes the two steps independently rather than directly maximizing their contextual margin. In contrast, pairwise objective learns a relative preference between two steps, which is better aligned with downstream task. For more details check Appendix C.3.

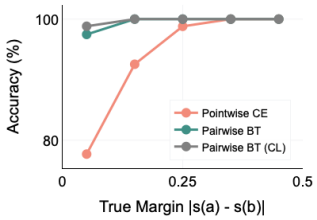


Figure 4: Pairwise classification accuracy of Pointwise versus Pairwise models.

Training with the step-contrastive loss requires matched positive–negative pairs with hard negatives that need no new human labels. We discuss this in section 3.2. We also need a curriculum strategy for learning a stronger margin as discussed in section 3.3.

3.2 Hard-Negative Augmentation Using Temporal Lookahead

To train with the step-contrastive loss, we need matched positive–negative step pairs whose negatives are genuinely hard: plausible continuations that can confuse the reward model while remaining invalid for the current prefix. Obtaining reliable annotations at this granularity is difficult, so we first use PRM800K's positive, negative, and neutral labels to form explicit positive–negative pairs from annotated intermediate steps, yielding about 26k training pairs. We then apply temporal lookahead to create additional hard negatives.

Simpler augmentation alternatives do not satisfy these hard-negative constraints as cleanly. Random negatives from other problems are often too easy because they are contextually unrelated; synthetic perturbations can introduce superficial artifacts that the model may exploit; and model-generated adversarial negatives require extra decoding, filtering, or human validation.

Our temporal lookahead augmentation strategy instead treats a future step from the same trajectory as a negative candidate for the current prefix. This teaches the model to judge whether a step is correct *at the current point* in the reasoning chain, not merely whether it is valid in isolation. The resulting pairs are challenging: the future step is usually relevant to the problem and may even be correct on its own, but it is still an invalid next step because it appears too early. For example, in a quadratic-equation solution, writing the final root before the intermediate simplifications may be mathematically valid, but it is not a valid continuation of the current partial solution. After augmentation, we obtain roughly 220k training pairs without any additional human labeling.

3.3 Margin Difficulty-Aware Curriculum Design

Learning a strong margin from hard positive–negative pairs with a contrastive loss is often noisy for the model. This is because pairwise losses are known to struggle on overly difficult comparisons (Gao et al., 2025; Wu et al., 2024a).

A natural strategy is therefore to gradually improve the margin by progressively introducing harder positive–negative pairs during training. We design a novel curriculum learning (Bengio et al., 2009) paradigm in which the margin itself determines the difficulty level. Specifically, we define the continuous difficulty score as the difference between the reward assigned to the positive and negative step. Let us call this score:

$$D_{\text{Hard}} = r_\theta(x, y_{<t}, y_t^{pos}) - r_\theta(x, y_{<t}, y_t^{neg})$$

where $0 \le r_\theta(x, y_{<t}, y_t) \le 1$. Thus, $D_{\text{Hard}} \in [-1, 1]$. We retain pairs with $D_{\text{Hard}} \ge 0$ and discard the rest, since negative values typically correspond to comparisons that are too difficult for the model to learn from reliably. The retained pairs are then divided into four bins; the number of samples in each bin is reported in Appendix D.1. We also compare training with and without curriculum learning on the same set of training examples in Tables 8, and provide ablations over different curriculum bins to show robustness to this hyperparameter.

Since existing PRMs, both discriminative and generative, exhibit high FPR and are trained on imbalanced class-label data, our proposed loss is broadly applicable to both.

4 Experiments and Results

Models and Baselines. The SOTA PRM model Qwen2.5-Math-PRM-7B (Zhang et al., 2025) serves as the primary baseline for all experiments. To test generalizability, ReasonEval-7B PRM (Xia et al., 2025) is also included in select experiments. Models trained with the proposed recipe are denoted as **PRISM** hereafter. Because Qwen-PRM is a discriminative model, comparisons additionally include ThinkPRM (Khalifa et al., 2025), a SOTA generative PRM. A standard baseline trained on the same dataset with the default pointwise loss is included as well.

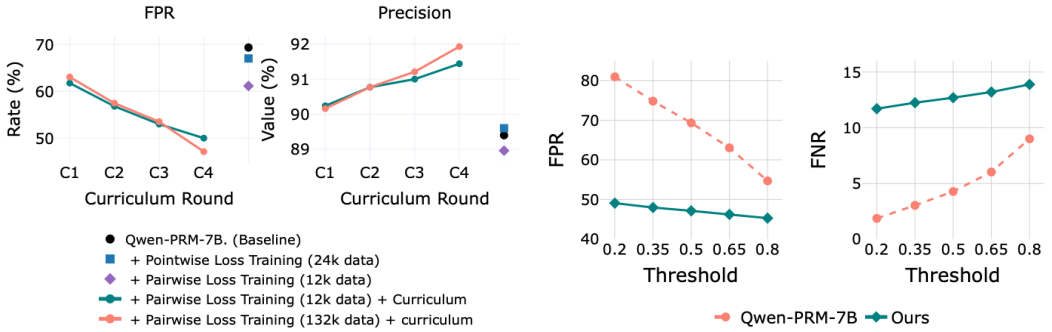
PRM Evaluation Benchmarks. Step-level reasoning quality is evaluated using **PRMBench** (Song et al., 2025) and **ProcessBench** (Zheng et al., 2025). PRMBench assesses the correctness, faithfulness, and robustness of intermediate chain-of-thought steps, providing detailed step-level accuracies and confusion matrices.

ProcessBench targets complex mathematical reasoning, evaluating a model's ability to pinpoint the first erroneous step in Olympiad-level solutions.

Policy Evaluation. Alignment is evaluated at both inference and training time. For **inference-time evaluation**, **best-of-N** and **guided beam-search** experiments are conducted using both in-distribution

Method	FPR ↓	FNR ↓	Precision ↑	TPR ↑	TNR ↑	PRMScore ↑	FPR Reduction ↑
Baseline (Qwen-PRM-7B)	69.34	4.27	89.40	95.36	30.66	65.50	0.00%
+ Pairwise Loss (12k data)	61.13	7.10	88.96	92.41	38.86	67.00	11.84%
+ Pairwise Loss (12k data) + CL	60.58	7.20	90.4	92.79	39.42	67.30	12.63%
+ Pointwise Loss + Data Aug. (132k)	67.21	6.00	89.60	94.00	32.79	65.60	3.07%
PRISM	**47.13**	**12.69**	**91.93**	**87.30**	**52.86**	**68.00**	**32.22%**

Table 2: Reported metrics include the False Positive Rate (FPR), False Negative Rate (FNR), Precision, TPR, TNR, and overall PRMScore. Because global accuracy is distorted by heavy class imbalance, these localized metrics better reflect model alignment. **PRISM, trained via pairwise loss, data augmentation, and curriculum learning,** significantly reduces FPR while improving overall PRMScore. Note: A 1–2% increase in PRMScore translates to substantial downstream capability gains (Song et al., 2025).



(a) FPR and precision across curriculum rounds.

(b) FPR–FNR trade-off across thresholds.

Figure 5: Curriculum and threshold analyses for PRISM. **Left:** Later curriculum rounds reduce false positives while maintaining strong precision. **Right:** Across classification thresholds, PRISM achieves a more favorable FPR–FNR trade-off than the baseline, indicating greater robustness to threshold selection.

(ID) and out-of-distribution (OOD) policies. Evaluation covers MATH-500 (Hendrycks et al., 2021), AIME 2024 problems, and LiveCodeBench (Jain et al., 2024). Following Khalifa et al. (2025), 100 MATH-500 problems spanning all difficulty levels are sub-sampled. For ID policies, the Qwen family is used: Qwen2.5-Math-1.5B-Instruct for MATH-500, Qwen2.5-Math-7B-Instruct for AIME 2024, and Qwen2.5-Coder-7B-Instruct (Hui et al., 2024) for coding tasks. For OOD evaluation on MATH-500, Llama-3.2-3B-Instruct is used. Finally, for **training-time evaluation**, policies are optimized using the GRPO algorithm on the Math-Verify dataset.

4.1 PRM Evaluation

PRMBench: Lower FPR and Higher Precision with PRISM. Table 8 presents detailed PRM-Bench results for PRISM. Additionally, Table 2 reports step-level metrics—including positive and negative accuracy, false-positive rate (FPR), false-negative rate (FNR), and precision—to evaluate the reduction in false positives.

A significant reduction in FPR is observed alongside a slight increase in precision, indicating that the model is more conservative in its positive predictions. While this is accompanied by a marginal rise in FNR, this degradation is substantially smaller than the FPR improvement. Moreover, unlike FPR, FNR does not directly harm policy alignment. Overall, the model demonstrates consistent performance gains. Although the numerical increase in PRMScore appears modest, such improvements translate to substantial gains in downstream model capability (Song et al., 2025).

FPR-FNR Trade-off Across Classifier Thresholds: Figure 5b illustrates the FPR-FNR curves for various classifier thresholds. PRISM demonstrates significantly greater robustness to threshold variations compared to the baseline.
Sensitivity to Curriculum Learning Bins: Ablation results across different curriculum learning bins (Appendix D.5) consistently reflect this same performance trend as discussed before.
Generalization to ReasonEval-7B PRM: Results in Appendix D.4 demonstrate that PRISM generalizes effectively beyond Qwen-PRM.
ProcessBench Evaluation: Evaluations on ProcessBench (Zheng et al., 2025) (Appendix D.2) reveal trends consistent with the PRMBench findings.

4.2 Policy Evaluation: Downstream Alignment

4.2.1 Inference-Time Search Evaluation

To evaluate whether PRM improvements translate into better policy alignment, both in-distribution (ID) and out-of-distribution (OOD) policies (generators) are tested using two alignment algorithms: **guided beam-search** and **best-of-N**. For both methods, a verifier-weighted majority rule is used to select the final answer, where answers are scored based on the sum of verifier scores across their supporting solutions (Wu et al., 2024b; Uesato et al., 2022). A detailed description of this selection strategy is provided in Appendix D.6. Results are reported using two generator model families: Qwen (ID) and LLaMA (OOD). Using multiple policy models is important because PRM bias is especially likely to surface when either the prompt or the response distribution is OOD relative to the PRM. OOD policies are especially likely to produce OOD responses, making them a strong testbed for alignment performance. Evaluating across different generators also ensures that the findings are not tied to a particular model family or size.

Guided Beam-Search. This is an extension of standard beam search. It incorporates verifier (PRM) scores when ranking partial reasoning chains. Instead of relying solely on model likelihoods, the search is guided toward trajectories that both the model and the verifier find promising, improving alignment with correct reasoning.

Best-of-N. samples N candidate solutions from the policy and then uses the PRM (or verifier) to score each solution.

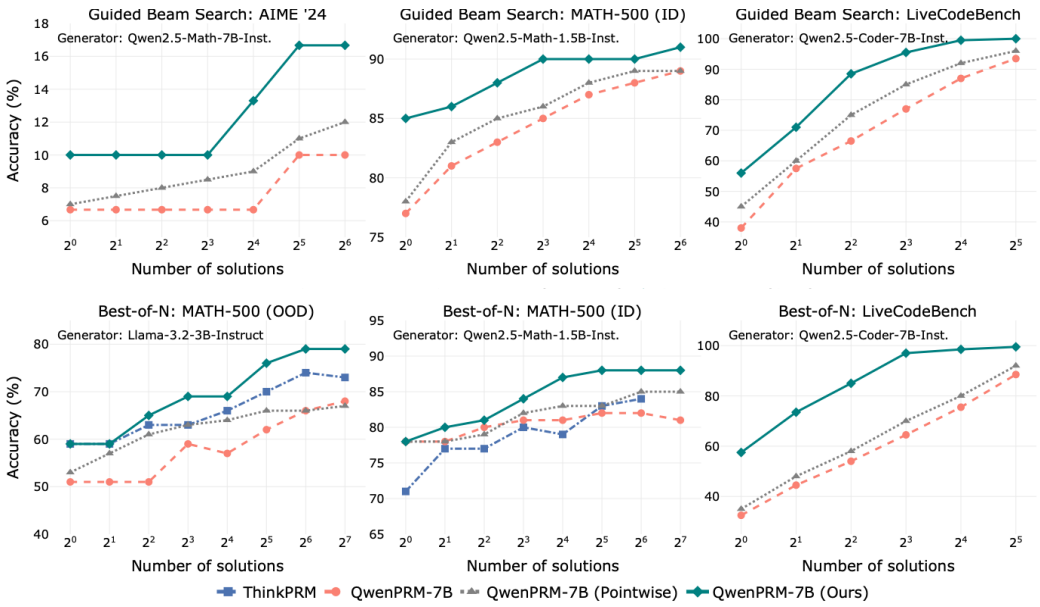


Figure 6: Comparison of Best-of-N and guided beam-search alignment on MATH-500, AIME'24, and LiveCodeBench. **Top:** Guided beam-search results on AIME'24 (left), MATH-500 (middle), and LiveCodeBench (right). Across all settings, PRISM (blue) consistently outperforms the baseline (orange), highlighting its effectiveness in guided beam-search decoding, where the PRM plays a critical role. **Bottom:** Best-of-N results on MATH-500 using two generator policies, LLaMA for OOD (left) and Qwen for ID setting (middle), and on LiveCodeBench using the Qwen2.5-Coder generator (right). ThinkPRM is shown in gray. PRISM achieves clear performance gains across all settings, with especially large improvements in the OOD regime.

Improvement in Alignment Performance. Figures 6 and 8 show alignment performance for both guided beam search and best-of-N strategies on MATH-500, AIME and LiveCodeBench tasks. Consistent and significant improvements appear across all settings. Notably, guided beam search exhibits substantially larger gains with PRISM than with the baselines, indicating a much stronger alignment signal and suggesting that PRISM is more effective. BoN relies on random sampling from the policy, after which the PRM selects the best solution from a fixed set. In contrast, guided beam search actively shapes generation by incorporating PRM feedback at each decoding step. This forces the PRM to reliably distinguish promising partial trajectories, providing a stronger and more realistic evaluation of whether the PRM captures true step-level reasoning quality rather than simply identifying winners post hoc.

Improvement in Alignment for both ID and OOD policy. Evaluating OOD policies is particularly important for studying bias-induced policy misalignment, since reward models are most vulnerable when they encounter data outside their training distribution. In such cases, the model may assign high rewards to spurious or incorrect reasoning. OOD policies naturally produce more OOD responses, making them an effective stress test for alignment robustness. Results in Figures 6 and 8 show that PRISM reduces bias-induced policy misalignment for both ID and OOD policies, confirming stronger generalization.

Improvement in Alignment across diverse datasets and tasks: Both mathematics and coding tasks are included. For mathematics, evaluation covers MATH-500 and AIME'24, where MATH-500 is closer in difficulty to the paired training data while AIME'24 is more challenging. Improvements on MATH-500 confirm that PRISM strengthens alignment within distribution, while gains on AIME'24 demonstrate generalization to more difficult, out-of-distribution problems. For coding, evaluation uses LiveCodeBench (Jain et al., 2024). These results, shown in Figure 6, indicate that the method improves the model's underlying reasoning ability rather than simply overfitting to training-like distributions.

4.2.2 Training-Time Policy Optimization via GRPO

To assess practical impact, a Qwen2.5-1.5B policy is optimized using GRPO. As shown in Table 3, substituting the Qwen-PRM-7B baseline with PRISM yields significant downstream gains. Most notably, GSM8K improves by an absolute $+11.37\%$, alongside strong gains on AMC and MATH-500. These results demonstrate that optimizing the FPR-FNR trade-off directly translates to a more capable generative policy.

PRM Model	AIME	AMC	GSM8K	MATH-500	Minerva	Olympiad
Baseline (Qwen-PRM-7B)	0.00	3.61	31.16	14.20	**6.25**	**5.04**
PRISM (Ours)	0.00	**9.64**	**42.53**	**15.80**	5.88	4.00

Table 3: Downstream GRPO performance on a Qwen2.5-1.5B policy. Using PRISM as the reward signal drives substantial gains on core mathematical reasoning benchmarks compared to the baseline.

5 Related Work

Process Reward Models. There are two kinds of PRM: Discriminative PRMs and Generative PRMs. Discriminative PRMs are typically framed as classification tasks, where the model assigns a correctness score to each reasoning step. These models require step-level supervision (Uesato et al., 2022; Lightman et al., 2023; Zhang et al., 2025). For a given solution prefix, the text is encoded and passed through a classification head that outputs step-level correctness probabilities, commonly trained using binary cross-entropy loss. To evaluate a full solution, the step-level scores are aggregated into an overall correctness measure (Beeching et al., 2024; Snell et al., 2024; Wu et al., 2024b). Generative PRMs (Khalifa et al., 2025; Zhao et al., 2025; Zheng et al., 2023; Zhu et al., 2023) treat verification as a sequence generation problem, where the model outputs natural language tokens such as "correct" or "incorrect" at each reasoning step. Instead of relying solely on binary labels, they are trained with the standard language modeling objective using explanatory rationales.

PRM Bias and Reward-Model Robustness Reward hacking is the broader term typically used in reinforcement learning (Skalse et al., 2022), and closely related reward-model bias has also been explored extensively in the context of ORMs for LLMs. However, little to no work has addressed PRM bias directly. Prior works have used reward model uncertainty as a training signal to guide policy learning, which in turn generates OOD samples for improving the reward model (Bukharin et al., 2025). Refer to Appendix A for more detailed related work section.

6 Conclusion and Limitations

In sum, we show that the alignment ceiling in process supervision is set not by recall but by precision, making false positives the central obstacle to safe reasoning. Our pairwise, augmentation-driven training recipe directly tackles this issue, reducing overcredit without additional human labels. By bridging PRM metrics with downstream alignment outcomes, our results suggest a clear design principle: precision-first process supervision for more trustworthy reasoning systems.

In practice, reducing false negatives without inflating false positives remains challenging. Achieving this balance—maintaining a low false-positive rate while also lowering false negatives—would be a significant step toward making PRMs far more capable and reliable.

REFERENCES

Bowen Baker, Joost Huizinga, Leo Gao, Zehao Dou, Melody Y Guan, Aleksander Madry, Wojciech Zaremba, Jakub Pachocki, and David Farhi. Monitoring reasoning models for misbehavior and the risks of promoting obfuscation. *arXiv preprint arXiv:2503.11926*, 2025.

Edward Beeching, Lewis Tunstall, and Sasha Rush. Scaling test-time compute with open models, 2024. URL https://huggingface.co/spaces/HuggingFaceH4/blogpost-scaling-test-time-compute.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pp. 41–48, 2009.

Alexander Bukharin, Haifeng Qian, Shengyang Sun, Adithya Renduchintala, Soumye Singhal, Zhilin Wang, Oleksii Kuchaiev, Olivier Delalleau, and Tuo Zhao. Adversarial training of reward models. *arXiv preprint arXiv:2504.06141*, 2025.

Guoxin Chen, Minpeng Liao, Chengxi Li, and Kai Fan. Alphamath almost zero: process supervision without process. *Advances in Neural Information Processing Systems*, 37:27689–27724, 2024.

Carson Denison, Monte MacDiarmid, Fazl Barez, David Duvenaud, Shauna Kravec, Samuel Marks, Nicholas Schiefer, Ryan Soklaski, Alex Tamkin, Jared Kaplan, et al. Sycophancy to subterfuge: Investigating reward-tampering in large language models. *arXiv preprint arXiv:2406.10162*, 2024.

Chengqian Gao, Haonan Li, Liu Liu, Zeke Xie, Peilin Zhao, and Zhiqiang Xu. Principled data selection for alignment: The hidden risks of difficult examples. *arXiv preprint arXiv:2502.09650*, 2025.

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

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

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

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

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

Muhammad Khalifa, Rishabh Agarwal, Lajanugen Logeswaran, Jaekyeom Kim, Hao Peng, Moontae Lee, Honglak Lee, and Lu Wang. Process reward models that think. *arXiv preprint arXiv:2504.16828*, 2025.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

Tianqi Liu, Wei Xiong, Jie Ren, Lichang Chen, Junru Wu, Rishabh Joshi, Yang Gao, Jiaming Shen, Zhen Qin, Tianhe Yu, et al. Rrm: Robust reward model training mitigates reward hacking. *arXiv preprint arXiv:2409.13156*, 2024.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Meiqi Guo, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, et al. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*, 2024.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Joar Skalse, Nikolaus Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward gaming. *Advances in Neural Information Processing Systems*, 35:9460–9471, 2022.

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

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

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

Peiyi Wang, Lei Li, Zhihong Shao, RX Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *arXiv preprint arXiv:2312.08935*, 2023.

Junkang Wu, Yuexiang Xie, Zhengyi Yang, Jiancan Wu, Jiawei Chen, Jinyang Gao, Bolin Ding, Xiang Wang, and Xiangnan He. Towards robust alignment of language models: Distributionally robustifying direct preference optimization. *arXiv preprint arXiv:2407.07880*, 2024a.

Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. An empirical analysis of compute-optimal inference for problem-solving with language models. *arXiv preprint arXiv:2408.00724*, 2024b.

Shijie Xia, Xuefeng Li, Yixin Liu, Tongshuang Wu, and Pengfei Liu. Evaluating mathematical reasoning beyond accuracy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 27723–27730, 2025.

Yuzi Yan, Xingzhou Lou, Jialian Li, Yiping Zhang, Jian Xie, Chao Yu, Yu Wang, Dong Yan, and Yuan Shen. Reward-robust rlhf in llms. *arXiv preprint arXiv:2409.15360*, 2024.

Adam X Yang, Maxime Robeyns, Thomas Coste, Zhengyan Shi, Jun Wang, Haitham Bou-Ammar, and Laurence Aitchison. Bayesian reward models for llm alignment. *arXiv preprint arXiv:2402.13210*, 2024.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. Rest-mcts*: Llm self-training via process reward guided tree search. *Advances in Neural Information Processing Systems*, 37:64735–64772, 2024.

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

Jian Zhao, Runze Liu, Kaiyan Zhang, Zhimu Zhou, Junqi Gao, Dong Li, Jiafei Lyu, Zhouyi Qian, Biqing Qi, Xiu Li, et al. Genprm: Scaling test-time compute of process reward models via generative reasoning. *arXiv preprint arXiv:2504.00891*, 2025.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Processbench: Identifying process errors in mathematical reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1009–1024, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

Lianghui Zhu, Xinggang Wang, and Xinlong Wang. Judgelm: Fine-tuned large language models are scalable judges. *arXiv preprint arXiv:2310.17631*, 2023.

A Additional Related Work

PRM Bias and Reward-Model Robustness Reward hacking is the broader term typically used in reinforcement learning (Skalse et al., 2022), and closely related reward-model bias has also been explored extensively in the context of ORMs for LLMs. However, little to no work has addressed PRM bias directly. Prior works have used reward model uncertainty as a training signal to guide policy learning, which in turn generates OOD samples for improving the reward model Bukharin et al. (2025). Their approach, however, relies on the strong assumption that all OOD samples correspond to incorrect outputs. Other lines of work have similarly proposed reward ensembles and Bayesian methods to improve robustness and reduce vulnerability to reward-model bias and proxy misalignment (Yan et al., 2024; Yang et al., 2024). Beyond Bayesian approaches, Liu et al. (2024) employ counterfactual augmentations to create paired data, explicitly breaking label-specific artifacts that might otherwise mislead the reward model. Even though the broader reward-hacking literature has not yet focused much on PRMs, shortcut behaviors—closely related to PRM bias—are pervasive in reasoning LLMs (Baker et al., 2025; Denison et al., 2024), which are often trained with process supervision. This highlights the importance of explicitly investigating and addressing PRM bias in process reward models.

B Training Details

B.1 Software and Hardware Details

All experiments were run on four Nvidia RTX A6000 GPUs.

B.2 Hyperparameters

The PRM training and inference settings are summarized in Table 4, and the GRPO training hyperparameters are reported in Table 5.

Table 4: PRM training and inference settings.

PRM Setting	**Value**
PRM step separator	`<extra_0>`
Response step parsing	Model responses split on `\n\n`
Train batch size	8
Train epochs	3
Precision	fp16
Optimizer	adamw_torch
Learning rate	5×10^{-5}
LR scheduler	linear
Warmup	none
Weight decay	0.0
Adam betas / eps	0.9, 0.999, 10^{-8}
Max grad norm	1.0

C Theoretical Results and Insights

C.1 False Positives vs. False Negatives in Policy Misalignment

This section analyzes how reward model errors (false-positive and false-negative prediction) affect Best-of-N (BoN) selection.

Table 5: GRPO training hyperparameters.

GRPO Training Hyperparameter	Value
Base model	Qwen2.5-1.5B-Instruct
Reward	Rule-based math verifier reward
Samples per prompt	8
Training steps	900
Prompt length	256
Generation length	512
Rollout batch size	8
Micro rollout batch size	4
Train batch size	16
Micro train batch size	2
Temperature / top-p	1.0 / 1.0
Discount γ	1.0
Initial KL coefficient	0.0
Learning rate	1×10^{-6}
LR scheduler	`cosine_with_min_lr`
Precision	bf16

C.1.1 Setup: Best-of-N Under an Imperfect Reward Model

Let $y_1, \ldots, y_N \sim \pi(\cdot \mid x)$ be i.i.d. samples. Let $T(y) \in \{0,1\}$ be the true binary label and $\hat{T}(y) \in \{0,1\}$ the reward model prediction. Best-of-N (BoN) selects

$$y^* = \arg\max_{i \in \{1,\ldots,N\}} \hat{T}(y_i). \tag{7}$$

Define

$$p = \mathbb{P}(T = 1), \tag{8}$$

$$\alpha = \mathbb{P}(\hat{T} = 1 \mid T = 0) \quad \text{(false positive rate)}, \tag{9}$$

$$\beta = \mathbb{P}(\hat{T} = 0 \mid T = 1) \quad \text{(false negative rate)}. \tag{10}$$

The marginal probability that a sample is predicted positive is

$$q = (1-p)\alpha + p(1-\beta). \tag{11}$$

Hence, the probability that at least one of the N samples is predicted positive is

$$\hat{p}_N = 1 - (1-q)^N. \tag{12}$$

Conditioned on $\hat{T} = 1$, the precision of the reward model is

$$\mathbb{P}(T = 1 \mid \hat{T} = 1) = \frac{p(1-\beta)}{q}. \tag{13}$$

Therefore, the BoN accuracy is

$$P^{(N)} = \underbrace{\left[1 - (1-q)^N\right]\mathbb{P}(T = 1 \mid \hat{T} = 1)}_{\text{predicted-positive event}} + \underbrace{(1-q)^N \mathbb{P}(T = 1 \mid \hat{T} = 0)}_{\text{all-negative event}} \tag{14}$$

$$= \left[1 - (1-q)^N\right]\frac{p(1-\beta)}{q} + (1-q)^N \frac{p\beta}{(1-p)(1-\alpha) + p\beta}. \tag{15}$$

In the regime of interest where N is moderate to large and the predicted-positive rate q is not vanishing, the probability of the all-negative event, $(1-q)^N$, decays exponentially in N. Consequently,

the second term in equation 15 contributes only a lower-order correction to $P^{(N)}$. For clarity of exposition, and to isolate the dominant mechanism governing BoN behavior, the leading term is used to approximate

$$P^{(N)} = \hat{p}_N \cdot \mathbb{P}\big(T = 1 \mid \hat{T} = 1\big) = \Big[1 - (1-q)^N\Big]\frac{p(1-\beta)}{q}. \tag{16}$$

C.1.2 Asymmetry I: False Positives Impose a Precision Ceiling

If $\beta \approx 0$, then $q = p + (1-p)\alpha$ and

$$P^{(N)} = \Big[1 - (1-q)^N\Big]\frac{p}{p + (1-p)\alpha}. \tag{17}$$

Taking the limit $N \to \infty$ yields,

$$\lim_{N\to\infty} P^{(N)} = \frac{p}{p + (1-p)\alpha} < 1 \qquad (\alpha > 0). \tag{18}$$

Thus, nonzero false positives impose a permanent precision ceiling. Moreover,

$$\frac{\partial}{\partial\alpha}\left(\frac{p}{p + (1-p)\alpha}\right) = -\frac{p(1-p)}{(p + (1-p)\alpha)^2} < 0, \tag{19}$$

showing that the ceiling degrades monotonically with α. This can also be visualized in Figure 3.

C.1.3 Asymmetry II: False Negatives Slow but Do Not Cap Alignment

If $\alpha \approx 0$, then $q = p(1-\beta)$ and

$$P^{(N)} = 1 - \big(1 - p(1-\beta)\big)^N. \tag{20}$$

Taking the limit $N \to \infty$ yields,

$$\lim_{N\to\infty} P^{(N)} = 1. \tag{21}$$

Thus, false negatives reduce the effective success probability per sample but do not induce asymptotic bias; increasing N fully compensates for false negatives.

C.2 Pairwise vs. Pointwise Loss

Setup. Let $s_t^+ = r_\theta(x, y_{<t}, y_t^{\text{pos}})$ and $s_t^- = r_\theta(x, y_{<t}, y_t^{\text{neg}})$. For one pair, pointwise BCE optimizes the two scores independently,

$$\mathcal{L}_{\text{BCE}}(s_t^+, s_t^-) = -\log s_t^+ - \log(1 - s_t^-),$$

whereas pairwise contrastive training optimizes only their relative ordering,

$$\mathcal{L}_{\text{SC}}(s_t^+, s_t^-) = -\log \sigma(s_t^+ - s_t^-).$$

C.3 Why Pairwise Training Outperforms Pointwise Training

The difference is easiest to see through a simple example. Consider a hard negative in mathematical reasoning: a step that sets up most of the solution correctly but makes a subtle arithmetic error at the end. Such a step shares substantial *valid* reasoning with the correct step. Suppose the PRM assigns scores 0.85 to the correct step and 0.80 to the hard negative. For downstream policy learning, this is already mostly a success because the ranking is correct ($0.85 > 0.80$), so the policy prefers the correct step through the exponentiated margin. BCE, however, still treats the 0.80 negative score as a large error and pushes it aggressively toward 0. Because the hard negative shares much of its structure with the positive step, this can create gradient conflict by suppressing useful reasoning patterns and

may increase FNs. Step-Contrastive loss instead focuses only on the margin $m = 0.05$, applying a targeted update that widens the gap while allowing both steps to retain relatively high absolute scores.

The difference is even clearer when the ranking is reversed, for example $s_t^+ = 0.40$ and $s_t^- = 0.45$. BCE again spends effort pushing the scores toward 1 and 0, whereas Step-Contrastive training focuses directly on flipping the ordering. This margin-centric view also motivates curriculum learning: large positive margins correspond to easy pairs, small positive margins correspond to hard but learnable pairs, and negative margins are often the most unstable early in training. Training is therefore organized from easier to harder pairs based on the predicted margin $s_t^+ - s_t^-$, which stabilizes optimization, avoids unstable early gradients, and sharpens the decision boundary.

Writing the scores in center–margin coordinates

$$a_t = \tfrac{1}{2}(s_t^+ + s_t^-), \qquad m_t = s_t^+ - s_t^-.$$

Then

$$\mathcal{L}_{\text{BCE}}(a_t, m_t) = -\log\Big(a_t + \tfrac{m_t}{2}\Big) - \log\Big(1 - a_t + \tfrac{m_t}{2}\Big), \qquad \mathcal{L}_{\text{SC}}(a_t, m_t) = -\log\sigma(m_t).$$

Equation 5 depends only on m_t, not on the center a_t. Accordingly,

$$\frac{\partial \mathcal{L}_{\text{SC}}}{\partial a_t} = 0, \qquad \frac{\partial \mathcal{L}_{\text{SC}}}{\partial m_t} = -\sigma(-m_t) < 0,$$

so SC is an exact surrogate for increasing the policy-relevant margin. By contrast, BCE also has

$$\frac{\partial \mathcal{L}_{\text{BCE}}}{\partial a_t} = -\frac{1}{a_t + m_t/2} + \frac{1}{1 - a_t + m_t/2},$$

which is generally nonzero. Thus, even on the same balanced paired data, BCE expends optimization effort on the absolute score level a_t, a nuisance quantity that cancels out in Equation 5.

Now let τ be a decision threshold. Since

$$\text{FNR}(\tau) = \mathbb{P}(s_t^+ < \tau) = \mathbb{P}(m_t < 2(\tau - a_t)), \qquad \text{FPR}(\tau) = \mathbb{P}(s_t^- \geq \tau) = \mathbb{P}(m_t \leq 2(a_t - \tau)),$$

a rightward shift of the margin distribution m_t lowers both the probability that negatives cross the threshold and the probability that positives fall below it, provided the center a_t is not simultaneously shifted upward. Hence a loss that directly enlarges m_t creates a wider threshold region in which FPR decreases without increasing FNR.

C.3.1 Distribution Shift: Pairwise vs. Pointwise Loss

Data Imbalance in Pointwise vs. Pairwise Loss. Class rebalancing can also be applied to pointwise (cross-entropy) training. However, pointwise loss trains a binary classifier that predicts the absolute correctness of an individual step ("is this step correct or not?"), whereas pairwise loss trains a model to learn relative preference between two steps ("is step A better than step B?"). The latter directly optimizes ranking quality, which downstream alignment methods rely on, as reflected in the BON scores. The margin induced by the pairwise loss also leads to a clearer separation between positive and negative steps, which is crucial for reducing false positives.

To illustrate this difference, the same Qwen-PRM is fine-tuned with both pairwise and pointwise losses on identical positive and negative data. Table 6 reports the resulting false-positive rate (FPR) comparison.

Takeaway. These results show that pairwise training reduces false positives more effectively than pointwise loss, which is the default training objective. Figure 7 additionally compares the reward distributions induced by pointwise and pairwise losses for PRM training.

C.3.2 Main Claim: Pairwise Training Reduces FPs at Fixed TPR.

Assumption: The analysis focuses on the practically relevant regime where N is moderate to large and the marginal predicted-positive rate $q = \mathbb{P}(\hat{T} = 1)$ is not vanishing. In this setting, BoN selects from the predicted-positive set with high probability, and the dominant contribution to $P^{(N)}$ arises from selection among predicted positives; the contribution from the event where no predicted positive is present becomes negligible.

Table 6: False-positive rate (FPR) comparison for pointwise- and pairwise-trained models.

Model	False Positives	False-Positive Rate (%)
Baseline	9109	69.34
SFT (Pointwise Loss)	8836	67.26
Pairwise Loss	**8112**	**61.75**

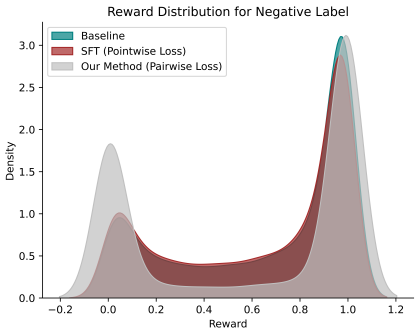


Figure 7: Reward distributions for negative labels across different QwenPRM variants. Green denotes the baseline QwenPRM, red denotes the model trained with default cross-entropy loss, and silver denotes the model trained with pairwise loss. The pairwise-trained model places substantially more mass around lower rewards for negative steps than the default PRM training method, indicating stronger separation of incorrect steps.

Theorem C.1 (ROC dominance $\Longrightarrow$ lower FPR)**.** ***Assumption:*** *Suppose pairwise training produces scores whose positive distribution dominates the negative distribution more than CE does. This assumption has been validated from distribution plots of Figure 7. Concretely, assume there exists* $\delta > 0$ *such that*

$$r_{\theta_{pair}}(y^{pos}) \overset{d}{=} r_{\theta_{ce}}(y^{pos}) + \delta, \qquad r_{\theta_{pair}}(y^{neg}) \overset{d}{=} r_{\theta_{ce}}(y^{neg}) - \delta.$$

where $r_{\theta_{pair}}(y)$ *is reward,* $r_\theta(x, y_{<t}, y_t)$ *from the PRM trained using pairwise loss and* $r_{\theta_{ce}}(y)$ *is the reward from the PRM trained using pointwise loss (cross-entropy loss).*

Then for every recall/TPR level $\rho \in (0, 1)$*, the FPR of the Pairwise-trained model is no larger:*

$$\mathrm{FPR}_{\theta_{pair}}(\tau_{pair}(\rho)) \;\le\; \mathrm{FPR}_{\theta_{ce}}(\tau_{ce}(\rho)),$$

where $\tau_{pair}(\rho)$ *and* $\tau_{ce}(\rho)$ *are thresholds chosen to achieve* $\mathrm{TPR} = \rho$ *under each model.*

Proof Sketch. Under the additive shift assumption, for any τ,

$$\Pr[r_{\theta_{\text{pair}}}(y^{pos}) \ge \tau] = \Pr[r_{\theta_{\text{ce}}}(y^{pos}) \ge \tau - \delta],$$
$$\Pr[r_{\theta_{\text{pair}}}(y^{neg}) \ge \tau] = \Pr[r_{\theta_{\text{ce}}}(y^{neg}) \ge \tau + \delta].$$

Additive shift assumption is valid after looking at the distribution plot of Figure 7.

To attain the same TPR ρ, the BT model uses a higher threshold: $\tau_{\text{pair}}(\rho) = \tau_{\text{ce}}(\rho) + \delta$. Plugging into FPR:

$$\mathrm{FPR}_{\theta_{\text{pair}}}(\tau_{\text{pair}}(\rho)) = \Pr[r_{\theta_{\text{pair}}}(y^{neg}) \ge \tau_{\text{pair}}(\rho)] \tag{22}$$
$$= \Pr[r_{\theta_{\text{ce}}}(y^{neg}) \ge \tau_{\text{pair}}(\rho) + \delta] \tag{23}$$
$$= \Pr[r_{\theta_{\text{ce}}}(y^{neg}) \ge \tau_{\text{ce}}(\rho) + 2 * \delta] \tag{24}$$
$$\le \Pr[r_{\theta_{\text{ce}}}(y^{neg}) \ge \tau_{\text{ce}}(\rho)] = \mathrm{FPR}_{\theta_{\text{ce}}}(\tau_{\text{ce}}(\rho)). \tag{25}$$

Thus, at any fixed recall/TPR, the BT model yields weakly smaller FPR.

Interpretation. The pairwise objective enlarges the score margin $\Delta = s(X^+) - s(X^-)$, which uniformly lifts the ROC curve. ROC dominance implies lower FPR for any desired TPR, i.e., fewer false positives.

D Experimental Details and Additional Results

D.1 Samples per Curriculum Stage

	Number of Samples.
Qwen-PRM-7B trained on 12k paired data.	
CL1 (0.5-1)	7.0k
CL2 (0.3-0.5)	2.0k
CL3 (0.2-0.3)	2.0k
CL4 (0.1-0.2)	1.0k
Qwen-PRM-7B trained on 132k paired data.	
CL1 (0.5-1)	75.0k
CL2 (0.3-0.5)	23.5k
CL3 (0.2-0.3)	23.5k
CL4 (0.1-0.2)	10.0k

Table 7: Number of samples in each curriculum learning round.

D.2 ProcessBench Results

ProcessBench(Zheng et al., 2025) evaluation shows the same overall pattern observed on PRMBench: lowering FPR is accompanied by a decrease in TPR.

	GSM8K			**MATH500**			**Omnimath**			**OlympiadBench**		
	TPR	**TNR**	**F1**	**TPR**	**TNR**	**F1**	**TPR**	**TNR**	**F1**	**TPR**	**TNR**	**F1**
Baseline	71.0	96.4	81.8	67.0	91.1	77.2	54.8	66.1	83.4	54.6	85.5	66.7
PRISM CL1	72.0	94.8	81.8	69.4	86.9	77.2	55.6	66.2	81.7	58.2	79.6	67.3
PRISM CL2	73.4	94.3	82.6	70.5	85.0	77.1	58.2	65.8	75.5	58.9	75.5	66.1
PRISM CL3	73.4	93.8	82.4	72.6	82.0	77.0	57.8	64.8	72.6	58.5	72.6	64.8

D.3 Detailed PRMBench Results

Model	**Overall**	**Simplicity**			**Soundness**					**Sensitivity**			
		NR.	**NCL.**	**Avg.**	**ES**	**SC.**	**DC.**	**CI**	**Avg.**	**PS**	**DR.**	**MS.**	**Avg.**
Qwen-PRM-7B													
Baseline	65.5	49.1	55.0	52.1	71.7	67.4	66.3	**78.5**	71.0	57.7	69.1	**99.7**	75.5
Qwen-PRM-7B trained on 12k paired data.													
Without Curriculum	67.0	50.6	59.3	54.9	73.2	68.4	66.9	78.0	71.6	59.0	70.7	99.6	76.5
Curriculum-1	66.7	50.5	58.6	54.5	72.8	68.1	67.3	77.8	71.5	58.8	70.4	99.6	76.3
Curriculum-2	67.2	51.4	60.2	55.8	73.3	68.0	67.7	76.4	71.4	60.4	70.4	99.3	76.7
Curriculum-3	67.3	52.5	63.2	57.8	73.2	67.2	67.5	75.0	70.7	61.9	69.8	98.9	76.9
Qwen-PRM-7B trained on 132k paired data.													
Curriculum-1	67.0	50.4	59.3	54.9	72.9	68.9	67.5	78.4	72.0	58.8	70.5	99.7	76.3
Curriculum-2	67.8	51.4	61.9	56.6	73.9	69.1	67.7	78.1	72.2	60.0	70.8	99.6	76.8
Curriculum-3	68.0	52.3	63.7	58.0	<u>73.9</u>	68.5	67.4	77.4	71.8	61.1	70.6	99.3	77.0
Curriculum-4	68.0	53.5	**66.0**	**59.7**	73.8	67.5	67.1	75.6	71.0	62.4	70.1	98.7	<u>77.1</u>

Table 8: PRMBench results for Qwen2.5-Math-PRM-7B (baseline) and its trained variants across different metrics. Columns are grouped into *Simplicity*, *Soundness*, and *Sensitivity*. Bold indicates the best score within each sub-metric, with PRMScore reported as the final comparison metric.

D.4 REASONEVAL-7B RESULTS

Results on another PRM model, ReasonEval-7B, are provided below to assess generalizability. Similar behavior appears here as well: later curriculum rounds substantially reduce FPR with only a slight increase in FNR, while PRMScore improves steadily. This model also appears in Table 1.

Threshold / Curriculum	PRM Score	FPR	FNR
CL1 (0.5 – 1.0)	61.0	78.79	4.21
CL2 (0.3 – 0.5)	61.8	69.76	8.05
CL3 (0.1 – 0.3)	62.3	66.8	9.9
CL4 (0.0 – 0.1)	62.3	61.7	13.6

Table 9: Performance of ReasonEval-7B PRM under the PRISM training recipe, illustrating generalization beyond Qwen-PRM.

D.5 CURRICULUM BIN ABLATIONS

Multiple threshold settings for curriculum learning are reported in Tables 10 and 11. Final performance remains consistent across different bin configurations, which is expected given the gradual progression toward more confusing or difficult regions. Across all curriculum-learning bin configurations, the false-positive rate decreases substantially while the false-negative rate increases only slightly as training advances through the curriculum rounds.

Threshold / Curriculum	PRM Score	FPR	FNR
CL1 (0.7 – 1.0)	66.4	65.58	5.4
CL2 (0.5 – 0.7)	67.2	62.4	6.29
CL3 (0.3 – 0.5)	68.0	56.6	8.24
CL4 (0.1 – 0.3)	68.2	49.0	11.0

Table 10: Results on Qwen-PRM-7B training under the PRISM recipe with one curriculum-bin configuration.

Threshold / Curriculum	PRM Score	FPR	FNR
CL1 (0.5 – 1.0)	67.4	61.25	6.65
CL2 (0.3 – 0.5)	68	54.61	9.16
CL3 (0.1 – 0.3)	67.8	45.70	13.56

Table 11: Results on Qwen-PRM-7B training under the PRISM recipe with an alternative curriculum-bin configuration.

D.6 ANSWER SELECTION METHODS

Setup: Consider N candidate solutions sampled from a model for the same problem. Each solution consists of:

1. A **final answer**, e.g., a number in GSM8K or an option in ARC.
2. A **verifier score**, assigned by a process reward model or external verifier, indicating how plausible or correct the reasoning chain appears.

Simple Majority Voting: In plain majority voting, completions are grouped by their **final answer**.

- Count how many completions lead to each distinct answer.
- Select the answer with the largest count.

Formally, if $c(a)$ is the number of completions yielding answer a, then the majority-vote answer is:

$$a^* = \arg\max_a c(a).$$

Verifier-Weighted Majority Voting: Instead of giving each completion equal weight, votes are weighted by verifier scores. Let answer a appear in solutions $\{s_1, s_2, \ldots, s_k\}$, where each solution s_i has verifier score $v(s_i)$. The total verifier-weighted score for answer a is:

$$V(a) = \sum_{s_i : \text{final}(s_i)=a} v(s_i).$$

The answer with the largest weighted score is then selected:

$$a^* = \arg\max_a V(a).$$

This approach discounts low-scoring (less credible) reasoning chains and prefers answers supported by higher-quality solutions.

D.7 OOD Guided Beam Search Results

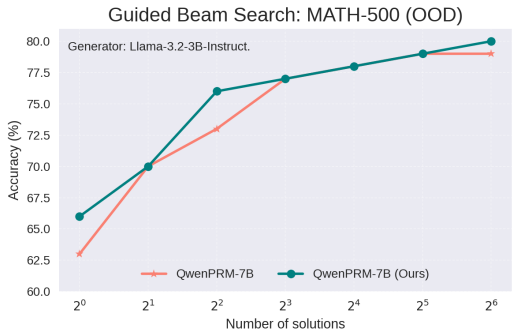


Figure 8: OOD guided beam-search results on MATH-500. PRISM (blue) substantially outperforms the baseline verifier (orange), even when the generator policy is out of distribution.

D.8 Statistical Plots

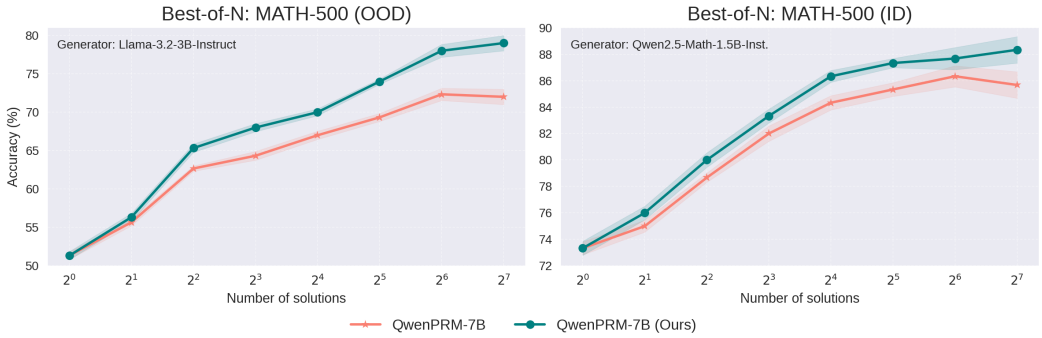


Figure 9: Error plot comparing best-of-N alignment on MATH-500 using two generator policies: LLaMA for the OOD policy (left) and Qwen for the ID policy (right). The baseline PRM is shown in orange and PRISM in blue. Across both settings, PRISM delivers a clear performance boost over the baseline, with the improvement being especially pronounced for the OOD policy.