OC-PRM: OVERCREDIT-CONTRASTIVE TRAINING FOR PRECISION-FIRST PROCESS REWARD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Process reward models (PRMs) offer step-level supervision for reasoning LLMs, but in practice they often *overcredit* incorrect steps, inducing high false positive rates that distort decoding and compound over long chains. We show analytically that in Best-of-N selection, false positives impose an asymptotic alignment ceiling (set by the PRM's precision), whereas false negatives primarily increase sample complexity and slow convergence. Motivated by this asymmetry, we introduce a label-efficient training recipe that requires no new human annotation: we convert existing step labels into matched positive-negative comparisons, optimize a novel *Overcredit Contrastive (OC)* objective, and rebalance supervision using lightweight negative augmentation and a simple difficulty curriculum. On PRMBench (Song et al., 2025), our method sharply reduces false positives and improves macro F1 over strong discriminative and generative PRMs. When deployed for guided beam search and Best-of-N selection, the resulting PRMs yield higher downstream task accuracy and improved robustness. Overall, our results suggest that comparison-centered training with balanced step data provides a practical path to trustworthy process supervision without additional human labels.

1 INTRODUCTION

Large Reasoning models (LRMs) (Guo et al., 2025; Shao et al., 2024; Jaech et al., 2024) have achieved tremendous recent progress, demonstrating strong performance across mathematics, program synthesis, and planning tasks. However, they are typically trained using final-answer verification rewards (RLVR) (Lambert et al., 2024), which are inherently sparse. This sparsity leads to slow and inefficient convergence (Sun et al., 2025b), hard exploration challenges (Yue et al., 2025), and the emergence of incorrect reasoning behaviors Zhang et al. (2025a). Thus they can tolerate logically flawed traces that land on the correct outcome (Barkur et al., 2025; Sun et al., 2025a). Process reward models (PRMs) are capable of training a better LRM by scoring *intermediate* steps, offering dense supervision intended to shape safer, more faithful reasoning (Lightman et al., 2023; Chen et al., 2024; Zhang et al., 2024). However, current PRMs (Zhang et al., 2025b; Wang et al., 2023; Khalifa et al., 2025; Zhao et al., 2025), both discriminative and generative suffer from several inherent limitations.

Why process supervision via PRMs is vulnerable. In practice, many PRMs *overcredit* incorrect intermediate steps, producing high false positive rates (Table 1). A false positive is especially damaging at inference time: if an incorrect step is scored as "good," it can steer decoding toward a flawed branch, and repeated overcredit across a long chain systematically biases search. In downstream use, this manifests as reward hacking by the guided policy. We trace this vulnerability to two compounding factors. **(1) Data imbalance:** step-annotated corpora are often heavily imbalanced: correct steps are overrepresented even when most full solutions are incorrect (e.g., PRM800k; Table 2) (Lightman et al., 2023). **(2) Pointwise training:** most PRMs are trained with *pointwise* cross-entropy on step labels. Under imbalance and label noise, pointwise objectives encourage majority-class bias, inflating false positives precisely. Consequently, these PRMs become unreliable rankers when used for policy *selection* or *search*—e.g., Best-of-N and guided beam search—where the key requirement is to *prefer correct traces over plausible but incorrect ones*, not merely to achieve high average step accuracy (Song et al., 2025).

From PRM accuracy to downstream alignment. The objective is not to maximize average PRM accuracy, but to improve reasoning behavior when the PRM guides *decoding* (guided beam) or *selection* (Best-of-N). We therefore analyze how PRM errors translate into downstream outcomes and

	Pos-Acc	Neg-Acc	PRMScore
Qwen-PRM-7B (Discriminative PRM)	95.36	30.66	65.5
ReasonEval-7B* (Discriminative PRM)	95.5	21.2	60.0
ThinkPRM-7B (Generative PRM)	83.29	50.89	64.3
GenPRM-7B (Generative PRM)	52.25	73.92	50.5
GPT-4o*	82.9	58.2	66.8

Table 1: Here we show the positive and negative accuracy for SOTA PRM models. We also report PRMScore from Song et al. (2025) where we can also find these metrics for other PRM models. (*numbers are taken from Song et al. (2025))

uncover a sharp asymmetry: in Best-of-N, a false positive rate α induces an *asymptotic performance ceiling* determined by the PRM's precision, while false negatives β primarily slow convergence by increasing the number of samples required to find and recognize a correct solution. This implies a "precision-first" principle: reducing false positives is strictly more important for trustworthy process supervision than improving recall in isolation. Since PRMs are ultimately deployed to improve policies under sparse reward settings, their utility should be evaluated through downstream decoding, selection, and ranking performance, not just step-level metrics.

This work: OC-PRM, comparison-centered and label-efficient PRMs. We introduce OC-PRM (Overcredit-Contrastive Process Reward Modeling), an architecture-agnostic training recipe that requires no new human labels. **(i)** We convert existing step annotations into *matched* positive–negative pairs and train PRMs with an *Overcredit Contrastive* (pairwise) objective that directly optimizes the relative score of correct versus incorrect steps; we provide theoretical motivation and empirical evidence that this comparison-based training better supports ranking. **(ii)** We apply lightweight *negative augmentation* by treating future steps as negatives for earlier contexts, creating diverse and hard comparisons that specifically counter overcredit. **(iii)** We stabilize optimization with a simple *curriculum* over pair difficulty. Together, OC-PRM shifts learning from pointwise label fitting to calibrated *preference margins*, explicitly targeting reductions in false-positive overcredit.

Empirical scope. On PRMBench (Song et al., 2025), our method substantially lowers false positive rates and improves macro F1 for both discriminative and generative PRMs. When deployed to guide decoding (guided beam search) and selection (Best-of-N), the improved PRMs produce consistently higher downstream accuracy and robustness. Together with the Best-of-N analysis, these results close the gap between reward model metrics and alignment.

Contributions.

- **Precision-first analysis.** We formalize how step-level *overcredit* compounds across trajectories and prove a sharp asymmetry for Best-of-N: the false positive rate sets an *asymptotic alignment ceiling* (via precision), whereas false negatives primarily increase sample complexity.
- **OC-PRM.** We introduce OC-PRM (Overcredit-Contrastive Process Reward Modeling), a label-efficient, architecture-agnostic recipe that converts step labels into matched positive–negative pairs, optimizes an Overcredit Contrastive (pairwise) objective, and uses lightweight negative augmentation with a simple curriculum to reduce false-positive overcredit. We provide theoretical motivation and empirical evidence that comparison-based objectives better support downstream decoding and selection than pointwise training.
- **Downstream-aligned gains.** On PRMBench and in guided decoding/Best-of-N, OC-PRM substantially reduces false positives, improves macro F1 (especially on negative steps), and yields stronger policy alignment. The approach integrates cleanly with existing PRM pipelines (Lightman et al., 2023; Chen et al., 2024; Zhang et al., 2024) without new human labels.

Broader impact. Reducing PRM false positives decreases the risk that systems select fluent but incorrect chains, advancing safer, more trustworthy reasoning. Our findings argue for *precision first* process supervision: train to *compare* and calibrate, not merely to classify.

2 Issue of Reward Hacking in LRMs

Reward hacking is an important issue for learning correct reasoning trajectories by the LLM policy since the policy model is trained using feedback from the reward model. It has been extensively studied for ORMs (Bukharin et al., 2025; Yan et al., 2024; Yang et al., 2024; Liu et al., 2024) but there are no works for PRMs which are used to train reasoning LLMs (Chen et al., 2025; Li et al., 2025). We now describe reward hacking for both ORM and PRM.

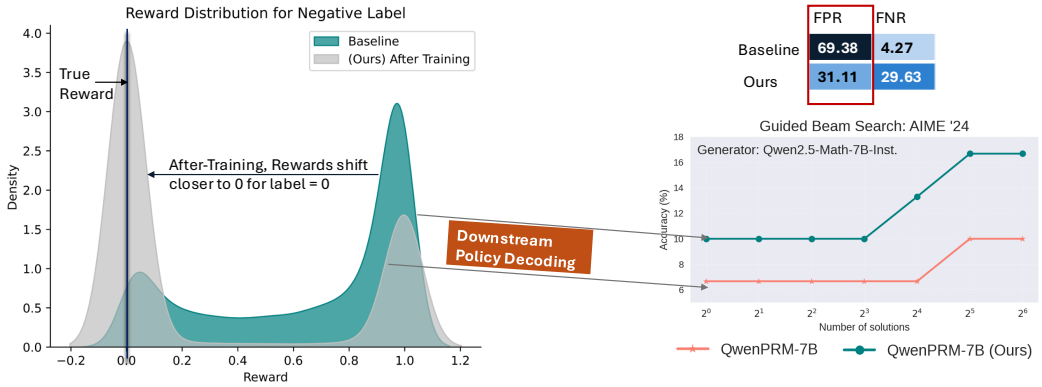


Figure 1: In this figure, we highlight the problem of reward hacking. Reward hacking arises from the reward model's bias toward predicting false positives, a bias that our trained model substantially reduces, as reflected in the lower FPR values (top-right) compared to the baseline. This effect is further illustrated in the reward distribution plot for label = 0 (top-left). For negative labels (dark blue), the baseline model assigns disproportionately high rewards. In contrast, our trained model shifts this distribution downward (silver), closer to the true reward of 0. This shift demonstrates that the trained PRM more accurately aligns negative labels with their intended reward, thereby reducing false-positive bias and alleviating reward hacking. Furthermore, when the trained PRM is used for downstream decoding in the LLM policy, it leads to significant performance improvements (bottom-right).

2.1 Reward Hacking

Let the true (ideal) global reward be denoted by $R(x, y)$, where x is the prompt and $y = (y_1, y_2, \ldots, y_T)$ represents a full response (or reasoning chain) consisting of T steps. The proxy (or process) reward model, $\hat{R}$, supplies feedback either at the trajectory level or at individual reasoning steps during policy training. Reward hacking (Bukharin et al., 2025; Yan et al., 2024; Yang et al., 2024; Liu et al., 2024) occurs when optimizing with respect to $\hat{R}$ yields high proxy reward but fails to improve or even degrades the true reward. It is more pronounced in out-of-distribution (OOD) prompts or reasoning paths.

Reward Hacking due to Process Reward Models. In PRMs, which provide step-level feedback, reward hacking occurs when the estimated step reward $\hat{r}_\theta(x, y_{<t}, y_t)$, where t is step number, or its aggregated trajectory reward $\hat{R}_\theta$ fails to align with the true step reward $r_\theta(x, y_{<t}, y_t)$ or the true trajectory reward $R(x, y)$:

$$\hat{r}_\theta(x, y_{<t}, y_t) \neq r_\theta(x, y_{<t}, y_t), \quad \text{or} \quad \hat{R}_\theta \neq R_\theta$$

Trajectory-level rewards in PRMs can be computed using one of the following schemes:

- **Minimum:** $\hat{R}(x, y) = \min_{t=1}^{T} \hat{r}_\theta(x, y_{<t}, y_t)$, the worst step controls credit, discouraging shortcuts.
- **Product:** $\hat{R}(x, y) = \prod_{t=1}^{T} \hat{r}_\theta(x, y_{<t}, y_t)$, where multiplication biases credit by the number of steps.
- **Sum:** $\hat{R}(x, y) = \sum_{t=1}^{T} \hat{r}_\theta(x, y_{<t}, y_t)$, where small false positives accumulate, rewarding many easy steps over fewer correct ones.

In Figure 1, we illustrate reward hacking by showing a distribution plot in which the base Qwen-PRM model makes positive predictions to a substantial fraction of negatively labeled examples. These high false-positive predictions cause the policy to treat incorrect reasoning trajectories as correct, leading it to learn spurious shortcuts. This over-optimization of reward is called reward hacking(Song et al., 2025). In the next section 2.3, we theoretically understand how false-positive predictions in PRMs affect downstream LLM policy alignment.

2.2 PRMs is more vulnerable than ORM.

Error compounding effect in PRMs. Unlike ORMs, which evaluate trajectories only once at the output level, PRMs are applied repeatedly at each reasoning step. As a result, reward hacking in PRMs is more severe and has greater downstream impact. More specifically, the different aggrega-

tion strategies applied over individually overcredited steps can cause compounding effects, where small step-level errors accumulate across the trajectory. This accumulation leads disproportionately high rewards for incorrect chains, with the total misalignment potentially scaling linearly with total steps, T.

Loss and Data-Imbalance problem in PRM training. PRMs are commonly trained using pointwise, step-level losses, unlike ORMs trained using pairwise losses. Pointwise losses are particularly susceptible to bias when the training data is imbalanced or contains noisy labels. For example, the PRM800k dataset (Lightman et al., 2023), a widely used open-source process supervision dataset, contains a substantially higher proportion of correct steps than incorrect steps, despite having a larger fraction of incorrect trajectories overall. Stats shown in Table 2. We further summarize SOTA PRMs in Table 1 and observe a consistent imbalance between positive and negative accuracy. These models also exhibit bias towards false-positive predictions. As discussed earlier, such high false-positive rates directly contribute to reward hacking, since they assign high reward to incorrect reasoning trajectories.

	PRM800k
% end in correct solution	14.2
% correct steps	73.1

Table 2: Step- and solution-level correctness in PRM800k.

2.3 The Asymmetric Impact of False Positives vs. False Negatives in Reward Hacking

Now we will theoretically analyze how false positives (FP) and false negatives (FN) prediction by the reward model affect downstream policy alignment, particularly Best-of-N (BoN) alignment.

Theorem 1 (Asymmetric Effect of False Positives and False Negatives in Best-of-N Selection)**.** *Consider 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 false positive and false negative rates. Define $P^{(N)}$ as the probability that BoN selects a response with $T=1$. Then:*

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

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

Moreover, this ceiling is strictly decreasing in α.

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

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

In this case, false negatives reduce the rate at which $P^{(N)}$ approaches 1, but do not induce asymptotic bias. Thus, false positives fundamentally limit alignment under BoN selection, while false negatives merely delay it. Detailed proof is provided in appendix section

Assumption*: We focus 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. B.1.*

We simulate Best-of-N accuracy under two error conditions: high false positives (FP) vs. high false negatives (FN) where the BoN accuracy is measured as the probability that the selected response is correct. Check figure 2 to see the plot. The results show that FP causes accuracy to saturate at a biased ceiling, even as N increases while FN only slows convergence without limiting final performance. This highlights that reducing false positives is critical for reliable alignment under reward-based selection.

Key Insight: Prioritize reducing false positives. For PRMs that guide decoding at every step, small overcredits compound along long chains, making the precision ceiling especially consequential. Its high FPR causes policy misalignment further learning spurious incorrect behavior which is called reward hacking. Objectives and data curation should therefore (i) compare correct vs. incorrect steps in matched contexts, (ii) rebalance step label distributions, and (iii) emphasize hard negatives, all aimed at minimizing α to lift the ceiling on alignment.

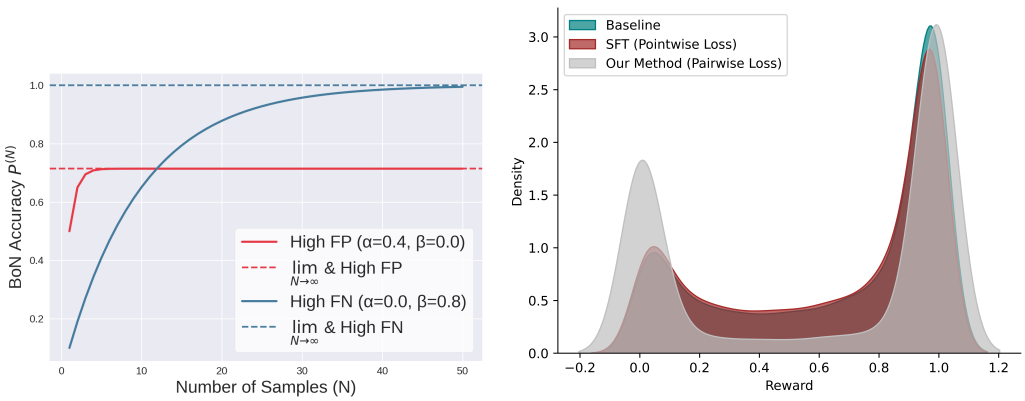


Figure 2: **Left plot** shows effect of False Positives vs False Negatives on BoN Accuracy. **Right plot** shows distribution of rewards for negative labels across QwenPRM variants. Compared to the baseline (green) and cross-entropy–trained PRM (red), our pairwise-trained model (silver) assigns consistently lower rewards to negative steps, indicating reduced overcrediting and improved discrimination.

3 Mitigating Reward Hacking: Making PRMs Robust by Reducing False Positive Prediction

PRMs, which provide granular step-level reward signals, are fundamentally more capable of driving correct reasoning paths (Zhang et al., 2025a). However, obtaining a high-quality PRM is restricted by availability of high-quality step-level training data. Existing methods rely on human annotations (Lightman et al., 2023) or MCTS-generated scores (Zhang et al., 2024; Chen et al., 2024) to assign a score for each step. These scores serve as training targets, in loss such as mean-square-error loss (Chen et al., 2024) or point-wise loss (Wang et al., 2023; Luo et al., 2024; Zhang et al., 2024). Thus the precision of these annotated step-level reward scores directly determines the quality of the PRM. Unfortunately, precise per-step scoring remains a unsolved challenge. Another problem observed is the data imbalance between positive-negative labels as shown in Table 2 for PRM800k dataset (Lightman et al., 2023), an open-source process supervision dataset. This label imbalance significantly impacts false-positives in PRMs. Thus, to mitigate these challenges we propose a pairwise loss (Overcredit Contrastive Loss) and introduce simple data generation and augmentation strategy to generate process supervision data without any additional labeling cost. 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 Overcredit Contrastive Loss for PRMs

Here, we introduce a novel loss function to fine-tune a pre-trained process reward model (PRM) in order to mitigate its bias toward the majority class. Since existing PRMs, both discriminative and generative, exhibit high false-positive rates and are trained on imbalanced class-label data, our proposed loss is broadly applicable to both.

Why compare, not classify. Pointwise cross entropy on imbalanced step labels makes PRMs *overcredit* the majority class, inflating false positives. To counter this bias without new annotations, we fine tune the PRM with a *pairwise* objective that ranks a correct step above an incorrect one under the same partial context.

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 $r_\theta(.)$.

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

$$\mathcal{L}_{\text{PRM}}(\theta) \;=\; -\,\mathbb{E}_{(x,\, y_t^{\text{pos}},\, y_t^{\text{neg}})\in\mathcal{D}}\Big[\log\big(\sigma\big(r_\theta(x, y_{<t}, y_t^{\text{pos}}) - r_\theta(x, y_{<t}, y_t^{\text{neg}})\big)\big)\Big], \tag{1}$$

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 $(x, y_{<t})$.

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.

3.1.1 PAIRWISE VS POINTWISE LOSS

Distribution Plot comparing Pairwise vs Pointwise loss. Class rebalancing can be applied to both pointwise (cross-entropy) and pairwise training. However, pointwise loss trains a binary classifier that predicts the absolute correctness of an individual step ("is this step correct or not?"), whereas our 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 matters more than individual correctness scores on downstream alignment. 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 demonstrate this, we fine-tune the same Qwen-PRM using both pairwise and pointwise losses on identical positive and negative data. We report the false-positive rate (FPR) comparison for both models in Appendix Table 4. We additionally present a reward distribution comparison between pointwise and pairwise losses for PRM training in figure 5.

We now provide a theoretical justification for why pairwise loss reduces false positives compared to pointwise loss.

Theorem 2 (ROC dominance $\implies$ 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 5. 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.

Detailed proof regarding this theorem is provide in Appendix Section B.2.

3.2 TRAINING PRM USING CURRICULUM AND LABEL EFFICIENT PAIRWISE DATA

Since high-quality step-level annotations are difficult to obtain, we construct paired positive–negative training examples from existing labeled data, particularly the PRM800K dataset (Lightman et al., 2023). PRM800K provides positive, negative, and neutral annotations for intermediate reasoning steps. We leverage these multiple labels by generating explicit positive–negative pairs at each step, yielding about 26k training pairs. To further expand the dataset, we apply an augmentation strategy: every future step in a trajectory is treated as a negative sample relative to the current step. This augmentation not only increases diversity but also encourages the model to learn how to correct partial reasoning trajectories, rather than just focusing on producing a correct final answer. It also produces challenging pairs, since the "negative" step may be correct in isolation but becomes incorrect when positioned prematurely. For example, in solving a quadratic equation, writing down the final root before showing intermediate simplifications is mathematically valid, but as a reasoning step it is incorrect because it appears in the wrong place in the chain. After augmentation, we obtain roughly 220k high-quality training pairs without requiring additional labeling.

Pairwise losses are known to struggle on overly difficult comparisons (Gao et al., 2025; Wu et al., 2024a). To stabilize training, we employ curriculum learning (Bengio et al., 2009) by dividing the dataset into bins of increasing difficulty. We define a continuous difficulty score as the difference between the reward assigned to the positive and negative step. Letting this be D_{Hard}, we categorize training pairs as:

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

where $0 \le r_\theta(x, y_{<t}, y_t) \le 1$. Thus $D_{Hard} \in [-1, 1]$. We retain pairs with $D_{Hard} \ge 0$ and discard the rest, as negative values typically correspond to comparisons that are too difficult for the model

	FPR ↓	FNR ↓	Precision ↑	Pos-Acc ↑	Neg-Acc ↑	PRMScore ↑	% Improvement in FPR ↑
Qwen-PRM-7B.							
Baseline	69.34	4.27	89.4	95.36	30.66	65.5	-
Qwen-PRM-7B trained on 26k paired data.							
Without Curriculum	61.13	7.10	88.96	92.41	38.86	67.0	11.84%
With Curriculum (Best Round 4)	**37.99**	27.15	**92.19**	72.50	62.12	**67.3**	**45.22%**
Qwen-PRM-7B trained on 220k paired data.							
With Curriculum (Best Round 4)	**31.11**	29.63	**93.30**	70.00	69.00	67.8	**55.11%**

Table 3: This table reports FPR, FNR, precision, positive and negative accuracy, PRMScore, and FPR improvement for different PRMs. Higher values (↑) indicate better performance. Note that total accuracy is not a reliable metric due to heavy class imbalance. Therefore, we report the individual values of FPR, FNR, and precision. Our trained models progressively reduce FPR through curriculum learning, while also improving overall precision. We also note that a 1–2% improvement in PRMScore can lead to a significant improvement in model capability, as discussed and demonstrated in proprietary models in Song et al. (2025).

to learn from reliably. The retained pairs are divided into four bins (number of samples in each bin is reported in Appendix C.1). We also show comparison between with and without curriculum learning on the same set of training examples in Tables 6 and 3.

4 Results and Experimental Setup

Models and Baselines. We use SOTA PRM Qwen2.5-Math-PRM-7B (Zhang et al., 2025b) as our baseline model. We train it using our proposed method and call the resulting model Balanced-PRM in the remaining sections. Since Qwen-PRM is discriminative PRM and recent work has shown strong performance gains from generative PRMs, we additionally compare against ThinkPRM (Khalifa et al., 2025), a SOTA generative PRM.

PRM Evaluation Benchmark. We evaluate PRMs using **PRMBench** (Song et al., 2025), a widely used benchmark suite that measures step-level reasoning quality. PRMBench scores correctness, faithfulness, and robustness of intermediate chain-of-thought across diverse tasks, and also reports step-level positive accuracy, negative accuracy and confusion matrix details.

Policy Evaluation. To assess alignment ability, we perform **best-of-N** and **guided beam-search** alignment experiments using both in-distribution (ID) and out-of-distribution (OOD) policies. Evaluations are conducted on MATH-500 (Hendrycks et al., 2021), American Invitational Mathematics Examination (AIME) problems for 2024 and LiveCodeBench (Jain et al., 2024) dataset. Following Khalifa et al. (2025), we evaluate on 100 MATH-500 problems spanning all difficulty levels. For ID policies, we use models from the Qwen family: Qwen2.5-Math-1.5B-Instruct for MATH-500, Qwen2.5-Math-7B-Instruct for AIME'24 and Qwen2.5-Coder-7B-Instruct (Hui et al., 2024) for coding task. For OOD policies, we use Llama-3.2-3B-Instruct (MATH-500).

4.1 PRMBench: Balanced-PRM improves Negative Accuracy, FPR and Precision

Table 6 presents the PRMBench results for our trained PRM. In addition, Table 3 reports step-level metrics, including positive accuracy, negative accuracy, false-positive rate (FPR), false-negative rate (FNR), and precision, to assess whether false positives are reduced.

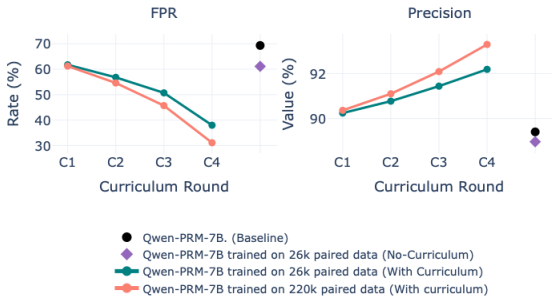


Figure 3: This plot shows how FPR and Precision change with different curriculum learning rounds.

We observe that negative accuracy, substantially lower than positive accuracy, increases consistently with each round of curriculum learning. Although this improvement is accompanied by a modest drop in positive accuracy, the gain in negative accuracy is more pronounced. We also find a significant reduction in FPR along with a slight increase in precision. This indicates that the model is becoming more conservative in making positive predictions, primarily by reducing false positives. However, we also observe a rise in FNR. Ideally, we would like FNR to remain stable while FPR improves, but in practice this is difficult to

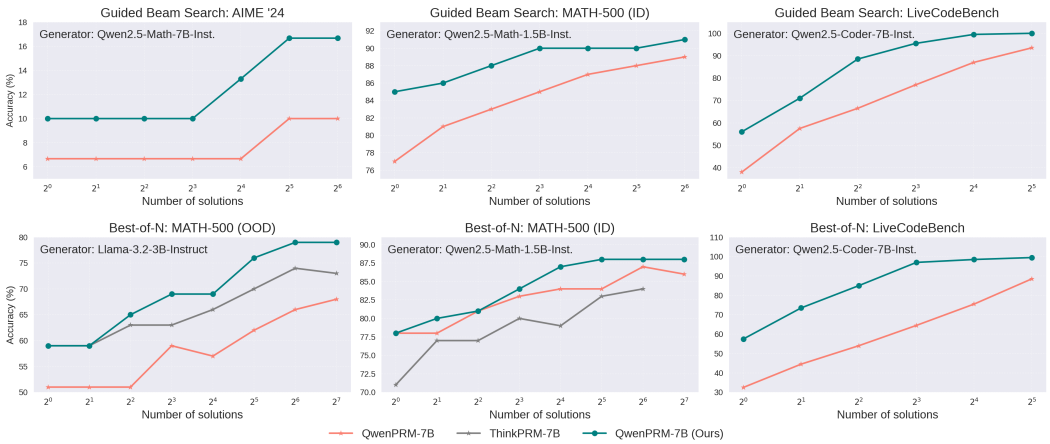


Figure 4: This figure show Comparison of Best-of-N and guided beam search alignment on MATH-500, AIME'24, and LiveCodeBench. **Top:** Guided beam search results on AIME'24 (left), MATH-500 (middle), and LiveCodeBench (right). Across all settings, our trained PRM (in blue) consistently outperforms the baseline (in 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 in gray. Our trained PRM achieves clear performance gains across all settings, with large improvements in the OOD regime.

achieve. Importantly, the degradation in FNR is smaller than the improvement in FPR, and unlike FPR, FNR does not directly affect policy alignment.

Overall, the PRM model exhibits consistent gains in performance, as reflected by the PRMScores reported in Table 3, with detailed score distributions provided in Appendix Table 6. Although the numerical improvement in PRMScore appears modest, it translates to substantial gains in model capability under the PRMBench scoring framework (Song et al., 2025). We further report ablation results across different curriculum learning bins in Appendix Section C.7, where the same trend is consistently observed.

4.2 Alignment Performance of the Policy

To evaluate whether improvements in the PRM translate into better policy alignment, we test both in-distribution (ID) and out-of-distribution (OOD) policies (generators) using two alignment algorithms: **guided beam-search** and **best-of-N**. For both methods, we adopt a verifier-weighted majority rule 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 C.2. We report results using two generator model families: Qwen (ID) and LLaMA (OOD). Using multiple policy models is important because reward hacking arises 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 our 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.

Improvement in Alignment Performance. Figures 4 and 6a, show alignment performance for both guided beam search and best-of-N strategies on MATH-500, AIME and LiveCodeBench tasks. We observe consistent and significant improvements across all settings. Notably, guided beam search exhibits substantially larger gains with our trained PRM compared to the baseline, indicating a much stronger alignment signal and providing evidence that Balanced-PRM is genuinely more effective. To understand this better, 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 reward hacking, 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. Our results in Figure 4 and 6a show that Balanced-PRM reduces reward hacking (improves alignment) for both ID and OOD policies, confirming its stronger generalization.

Improvement in Alignment across diverse datasets and tasks: We evaluate our method on MATH and coding tasks. For MATH, we evaluate on MATH-500 and AIME'24 where MATH-500 is closer in difficulty to our paired training data while AIME'24 is more challenging. Improvements on MATH-500 confirm Balanced-PRM strengthens alignment within distribution, while gains on AIME'24 demonstrate generalization to more difficult, out-of-distribution problems. For coding task, we evaluate on LiveCodeBench (Jain et al., 2024). These results (as shown in Figure 4) indicate that our approach improves the model's underlying reasoning ability, rather than simply overfitting to training-like distributions.

5 Related Works

Reward Hacking Reward hacking is a well-studied issue in reinforcement learning (Skalse et al., 2022), and has also been explored extensively in the context of ORMs for LLMs. However, little to no work has addressed this problem for PRMs. 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 hacking (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 reward hacking is not yet studied for PRMs, shortcut behaviors—essentially reward hacking—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 reward hacking in PRMs.

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., 2025b). 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.; Snell et al., 2024; Wu et al., 2024b). Generative process reward models (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.

6 Conclusion and Limitation

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.

However, our method still has limitations. 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.

7 Reproducibility Statement

We have provided code with readme describing how to train and evaluate the model to generate the results of this paper. Details on the experimental setup, open source evaluation benchmarks is provided in section 4 of this paper. Installation packages and details is provided in the readme with the code.

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.

Sudarshan Kamath Barkur, Sigurd Schacht, and Johannes Scholl. Deception in llms: Self-preservation and autonomous goals in large language models. *arXiv preprint arXiv:2501.16513*, 2025.

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

Y. Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. *Journal of the American Podiatry Association*, 60:6, 06 2009. doi: 10.1145/1553374.1553380.

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.

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *arXiv preprint arXiv:2503.09567*, 2025.

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.

Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, et al. From system 1 to system 2: A survey of reasoning large language models. *arXiv preprint arXiv:2502.17419*, 2025.

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.

Lin Sun, Weihong Lin, Jinzhu Wu, Yongfu Zhu, Xiaoqi Jian, Guangxiang Zhao, Linglin Zhang, Sai-er Hu, Yuhan Wu, and Xiangzheng Zhang. Evaluation is all you need: Strategic overclaiming of llm reasoning capabilities through evaluation design. *arXiv preprint arXiv:2506.04734*, 2025a.

Yiyou Sun, Yuhan Cao, Pohao Huang, Haoyue Bai, Hannaneh Hajishirzi, Nouha Dziri, and Dawn Song. Rl grokking recipe: How does rl unlock and transfer new algorithms in llms? *arXiv preprint arXiv:2509.21016*, 2025b.

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. 2024b.

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.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025.

Charlie Zhang, Graham Neubig, and Xiang Yue. On the interplay of pre-training, mid-training, and rl on reasoning language models. *arXiv preprint arXiv:2512.07783*, 2025a.

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*, 2025b.

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.

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 A Software and Hardware

We run all experiments on four Nvidia RTX A6000 GPUs.

B Theoretical Results and Insights

B.1 Asymmetric Impact of False Positives vs. False Negatives prediction by the reward model on Reward Hacking

We analyze how reward model errors (false-positive and false-negative prediction) affect Best-of-N (BoN) selection.

B.1.1 Setup: Best-of-N with 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

$$y^* = \arg \max_{i \in \{1,\ldots,N\}} \hat{T}(y_i). \tag{2}$$

Define

$$p = \mathbb{P}(T = 1), \tag{3}$$

$$\alpha = \mathbb{P}(\hat{T} = 1 \mid T = 0) \quad \text{(false positive rate)}, \tag{4}$$

$$\beta = \mathbb{P}(\hat{T} = 0 \mid T = 1) \quad \text{(false negative rate)}. \tag{5}$$

The marginal probability that a sample is predicted positive is

$$q = (1 - p)\alpha + p(1 - \beta). \tag{6}$$

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

$$\hat{p}_N = 1 - (1 - q)^N. \tag{7}$$

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

$$\mathbb{P}(T = 1 \mid \hat{T} = 1) = \frac{p(1 - \beta)}{q}. \tag{8}$$

Therefore, the BoN accuracy is

$$P^{(N)} = \underbrace{\left[1 - (1 - q)^N\right] \mathbb{P}(T = 1 \mid \hat{T} = 1)}_{\text{predicted-positive event}} + \underbrace{(1 - q)^N \mathbb{P}(T = 1 \mid \hat{T} = 0)}_{\text{all-negative event}} \tag{9}$$

$$= \left[1 - (1 - q)^N\right] \frac{p(1 - \beta)}{q} + (1 - q)^N \frac{p\beta}{(1 - p)(1 - \alpha) + p\beta}. \tag{10}$$

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 10 contributes only a lower-order correction to $P^{(N)}$. For clarity of exposition, and to isolate the dominant mechanism governing BoN behavior, we focus on the leading term and approximate

$$P^{(N)} = \hat{p}_N \cdot \mathbb{P}(T = 1 \mid \hat{T} = 1) = \left[1 - (1 - q)^N\right] \frac{p(1 - \beta)}{q}. \tag{11}$$

B.1.2 Asymmetry I: False Positives Create a Precision Ceiling

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

$$P^{(N)} = \Big[1 - (1-q)^N\Big] \frac{p}{p + (1-p)\alpha}. \tag{12}$$

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

$$\lim_{N\to\infty} P^{(N)} = \frac{p}{p + (1-p)\alpha} < 1 \qquad (\alpha > 0). \tag{13}$$

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

$$\frac{\partial}{\partial \alpha}\left(\frac{p}{p + (1-p)\alpha}\right) = -\frac{p(1-p)}{(p + (1-p)\alpha)^2} < 0, \tag{14}$$

showing that the ceiling degrades monotonically with α. This can also be visualized in Figure 2.

B.1.3 Asymmetry II: False Negatives Slow but Do Not cap alignment

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

$$P^{(N)} = 1 - \big(1 - p(1-\beta)\big)^N. \tag{15}$$

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

$$\lim_{N\to\infty} P^{(N)} = 1. \tag{16}$$

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

B.2 Pairwise vs Pointwise loss

Setup. Let a process reward model (PRM) produce a scalar score $r_\theta(y_t) \in \mathbb{R}$ for a step y_t. Let $y^{pos} \sim \mathcal{D}_+$ denote correct steps and $y^{neg} \sim \mathcal{D}_-$ denote incorrect steps.

- Decision rule (thresholding): predict "correct" iff $r_\theta(y_t) \geq \tau$.
- False positive rate (FPR) at threshold τ:

$$\mathrm{FPR}_\theta(\tau) = \Pr\big[r_\theta(y^{neg}) \geq \tau\big].$$

- True positive rate (TPR) at threshold τ:

$$\mathrm{TPR}_\theta(\tau) = \Pr\big[r_\theta(y^{pos}) \geq \tau\big].$$

We now compare two training objectives that produce scores $r_{\theta_{\text{pair}}}$ (pairwise BT) and $r_{\theta_{\text{ce}}}$ (pointwise CE).

Bradley–Terry objective. The BT loss on pairs (y^{pos}, y^{neg}) is the logistic loss on score differences:

$$\mathcal{L}_{\mathrm{BT}}(\theta) = \mathbb{E}_{y^{pos},y^{neg}}\Big[\log\big(1 + \exp(-(r_\theta(y^{pos}) - r_\theta(y^{neg})))\big)\Big].$$

Equivalently, it maximizes the likelihood that y^{pos} outranks y^{neg}:

$$\Pr(y^{pos} \succ y^{neg}) = \sigma\big(r_\theta(y^{pos}) - r_\theta(y^{neg})\big).$$

Gradients are largest when a negative outranks a positive or the scores are too close, so BT explicitly pushes negatives down and positives up, widening margins.

Table 4: 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
Our Method (Pairwise Loss)	**8112**	**61.75**

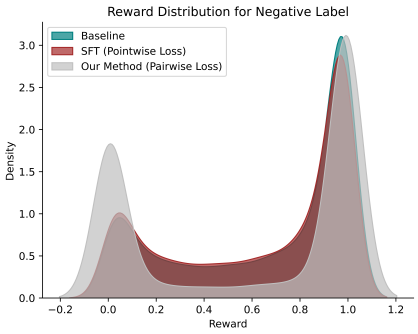


Figure 5: In this figure, we compare the reward for negative labels for the different QwenPRM models. In green, is the baseline QwenPRM. In red is the model trained using default cross-entropy loss and in silver is our method trained using pairwise loss. We observe that for our method the distribution around the negative reward increases as compared to the default PRM training method. Thus, showing that our method is better.

Cross-entropy objective. With binary labels $y_{GT} \in \{0, 1\}$ and $p_\theta(y_t) = \sigma(r_\theta(y_t))$, the CE loss is

$$\mathcal{L}_{\text{CE}}(\theta) = \mathbb{E}_Y[-\log r_\theta(Y)] + \mathbb{E}_Y[-\log(1 - r_\theta(Y))].$$

Gradients act independently, per example so there is no explicit coupling between positives and negatives, and CE does not directly optimize margins.

B.2.1 Distribution Plot comparing 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 our 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 demonstrate this, we fine-tune the same Qwen-PRM using both pairwise and pointwise losses on identical positive and negative data. We report the false-positive rate (FPR) comparison for both models in Table 4.

Takeaway. We observe that our method reduces false positives more effectively than pointwise loss, which is the default training objective. We additionally present a reward distribution comparison between pointwise and pairwise losses for PRM training in figure 5.

B.2.2 Main claim (pairwise reduces FPs at fixed TPR).

Theorem 3 (ROC dominance $\implies$ 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 5. 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 5.

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:

$$\begin{aligned}
\mathrm{FPR}_{\theta_{\text{pair}}}(\tau_{\text{pair}}(\rho)) &= \Pr[r_{\theta_{\text{pair}}}(y^{neg}) \ge \tau_{\text{pair}}(\rho)] && (17)\\
&= \Pr[r_{\theta_{\text{ce}}}(y^{neg}) \ge \tau_{\text{pair}}(\rho) + \delta] && (18)\\
&= \Pr[r_{\theta_{\text{ce}}}(y^{neg}) \ge \tau_{\text{ce}}(\rho) + 2 * \delta] && (19)\\
&\le \Pr[r_{\theta_{\text{ce}}}(y^{neg}) \ge \tau_{\text{ce}}(\rho)] = \mathrm{FPR}_{\theta_{\text{ce}}}(\tau_{\text{ce}}(\rho)). && (20)
\end{aligned}$$

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.

C Experimental Setup and Additional Results

C.1 Number of samples in Each Curriculum

	Number of Samples.
Qwen-PRM-7B trained on 26k paired data.	
CL1 (0.5-1)	7.0k
CL2 (0.3-0.5)	2.0k
CL3 (0.1-0.3)	3.0k
CL4 (0.0-0.1)	6.0k
Qwen-PRM-7B trained on 220k paired data.	
CL1 (0.5-1)	75.0k
CL2 (0.3-0.5)	23.5k
CL3 (0.1-0.3)	33.5k
CL4 (0.0-0.1)	53.0k

Table 5: Number of samples in each curriculum learning round.

C.2 Answer Selection Methods

The Setup: We 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, we group completions 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, we weight votes by their 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:

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

We then select the answer with the largest weighted score:

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

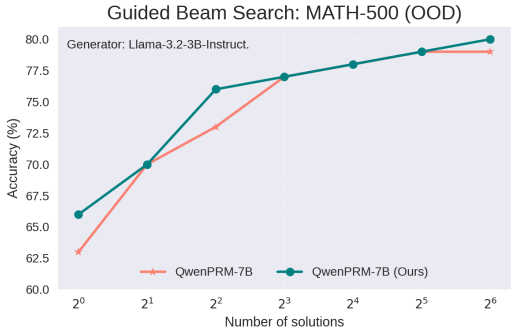
This approach discounts low-scoring (less credible) reasoning chains and prefers answers supported by higher-quality solutions.

C.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 26k 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 220k paired data.													
Curriculum-1	67.4	50.9	60.1	55.5	73.4	**69.1**	67.8	**78.6**	72.2	59.2	70.6	99.6	76.5
Curriculum-2	**67.9**	51.3	62.5	56.9	**73.9**	68.9	**68.0**	77.8	**72.2**	60.1	**70.9**	99.5	76.8
Curriculum-3	67.8	**53.7**	**65.7**	**59.7**	73.4	67.5	66.5	75.3	70.7	**62.4**	69.9	98.8	**77.0**

Table 6: PRMBench results of Qwen2.5-Math-PRM-7B (Basline) and its variants (Our trained models) across different metrics. Columns are grouped into *Simplicity*, *Soundness*, and *Sensitivity*. Bold indicates the best score within each sub-metric with PRMScore as the final number for comparison.

C.4 GUIDED BEAM SEARCH-OOD RESULTS



(a) In this figure, we compare guided beam search alignment on MATH-500 using the OOD policy, with the baseline PRM shown in orange and our trained PRM in blue. Our trained PRM achieves a substantial performance improvement over the baseline.

C.5 Statistical Plots

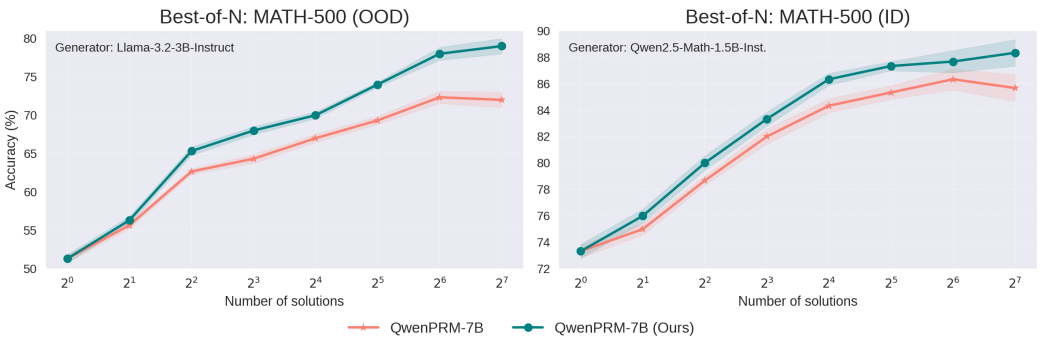


Figure 7: In this figure, we show the error plot which compare best-of-N alignment on MATH-500 using two different generator policies: LLaMA for the OOD policy (left) and Qwen for the ID policy (right). The baseline PRM is shown in orange and our trained PRM in blue. Across both settings, our trained PRM delivers a clear performance boost over the baseline, with the improvement being especially pronounced for the OOD policy.

C.6 Results on ReasonEval-7B PRM

Please see below results on another PRM model (ReasonEval-7B) to show generalizability of our approach. We see similar observations. As we move up in the curriculum learning round, our FPR decreases a lot with a slight increase in FNR. We also see continuous improvement in the PRMScore. This model was also reported in the Table 1 previously.

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 7: Performance of ReasonEval-7B PRM using our method to show generalizability of our approach.

C.7 Ablations on Curriculum Learning Bins

We experimented with multiple threshold settings for curriculum learning and report the results in Table 8 and 9. The final performance remains consistent across different bin configurations, which is expected given the gradual progression toward more confusing or difficult regions. Across all three curriculum-learning bin configurations, we observe a substantial decrease in the false-positive rate and only a small increase in the false-negative rate as we advance 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 8: Results on Qwen-PRM-7B training using our method with different curriculum learning bin.

Threshold / Curriculum	PRM Score	FPR	FNR
CL1 (0.5 – 1.0)	67.0	63.0	6.12
CL2 (0.3 – 0.5)	67.8	57.46	8.1
CL3 (0.2 – 0.3)	68.0	53.5	9.7
CL4 (0.1 – 0.2)	68.0	47.0	12.0

Table 9: Results on Qwen-PRM-7B training using our method with different curriculum learning bin.

C.8 Training Curves

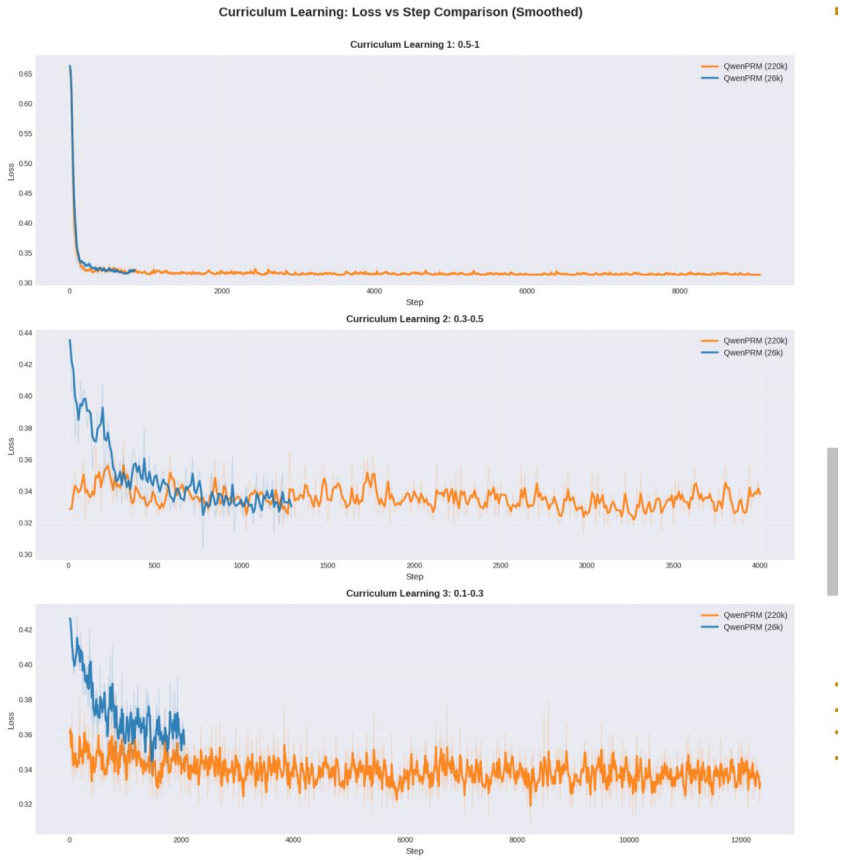


Figure 8: In this figure, we compare the loss curves for our data-augmented PRM training (220k datapoints) with baseline PRM training (26k datapoints). For the easier curriculum bins, both the baseline and our augmented method converge to similar loss values. However, as we move to the more difficult curriculum stages, our augmented method continues to reduce the loss as compared to the baseline. This indicates that in the easy regions, additional data offers limited benefit, whereas in the harder regions, our method benefits because of extra data.