ARMOR-PRM: Adversarially Robust Process Reward Models via Counterfactual Margins and Invariance Regularization

Anonymous ACL submission

Abstract

Process Reward Models (PRMs) are vulnerable to reward hacking: policies exploit surface cues such as confident phrasing or structured formatting in place of logical correctness, which yields an inverted-U collapse under optimization pressure. We propose ARMOR-PRM, which augments standard PRM training with three regularizers: a counterfactual margin loss, an adversarial invariance loss with low-score anchoring, and a gradient penalty on the reward head. On ProcessBench (three splits; `math` excluded due to training data overlap), ARMOR-PRM improves Detection Rate by 14.9 percentage points and AUROC by 4.1 percentage points over Vanilla PRM, and is the only variant whose Best-of-N reranking accuracy improves monotonically through $N = 128$. A domain-shift diagnostic further shows that objective-level robustness and data coverage are distinct but complementary axes: ARMOR-PRM responds best to iterative hard-negative retraining, suggesting a practical two-stage deployment path.

1 Introduction

Process Reward Models (PRMs) score each intermediate step in a reasoning chain, providing denser supervision than Outcome Reward Models that only score the final answer (Lightman et al., 2024; Wang et al., 2024). This step-level signal is used both for inference-time Best-of-N reranking and as a reward function for online reinforcement learning (RL) in agentic tasks (Setlur et al., 2025; Luo et al., 2024). The same granularity that makes PRMs attractive also enlarges the attack surface: a policy can manipulate individual steps, not just the terminal output.

Under optimization pressure, the proxy reward and the true task performance drift apart, producing an inverted-U curve (Gao et al., 2023; Cobbe et al., 2021). We attribute this to *shortcut learning* (Geirhos et al., 2020; Du et al., 2023): PRMs come to treat confident phrasing and structured formatting as proxies for logical validity. For example, a step with an algebraic sign error scores 0.35 under a standard PRM; prepending "By the triangle inequality, it follows directly that…" raises the score to 0.72 with the error intact. In agent environments, the same mechanism produces *phantom execution*: a policy claims to execute actions it never issues, inflating per-step scores without producing task progress.

We formalize this as *exploitability*: the expected maximum reward by which an adversarially disguised flawed step can exceed a correct step,

$$\mathcal{E}(\mathcal{R}_\theta) = \mathbb{E}_{(\tau^+,\tau^-)\sim\pi}\Big[\max_{\hat{\tau}^- \in \mathcal{N}_\epsilon(\tau^-)} \max\big(0,\, \mathcal{R}_\theta(\hat{\tau}^-) - \mathcal{R}_\theta(\tau^+)\big)\Big] \tag{1}$$

where $\mathcal{N}_\epsilon(\tau^-)$ is the adversarial neighborhood of a flawed step τ^- (§3.1), and (τ^+, τ^-) are paired correct/flawed steps from the same reasoning position.

Two Robustness Dimensions. We identify two distinct dimensions of PRM robustness. The first is *objective design*: what the training loss penalizes, which governs robustness under a fixed perturbation space. The second is *data coverage*: which failure modes the training set contains, which governs robustness to on-policy exploitation paths absent from static data.

ARMOR-PRM. ARMOR-PRM targets the first dimension with three regularizers: (i) a **counterfactual margin loss** $\mathcal{L}_{\text{cf}}$ enforcing a minimum reward gap between correct steps and their adversarially disguised flawed counterparts; (ii) an **adversarial invariance loss** $\mathcal{L}_{\text{adv}}$ penalizing reward changes under semantics-preserving perturbations and anchoring perturbed steps to the low-reward region; and (iii) a **gradient penalty** $\mathcal{L}_{\text{lip}}$ bounding the local Lipschitz constant of the reward head, extending robustness beyond predefined perturbation types. The

three losses control the quantities in our observed-neighborhood diagnostic $\mu_i > L\delta_i + \eta_i$ (Proposition 1), which holds on 90.2% of held-out pairs (§5.2). For the second dimension, we show that iterative retraining with on-policy hard negatives closes the data-coverage gap, and that ARMOR-PRM's regularization structure makes it the most responsive variant to such feedback.

Our contributions are:

- We identify two distinct dimensions of PRM robustness, objective design and data coverage, and introduce ARMOR-PRM together with an observed-neighborhood diagnostic for the first (§3).
- ARMOR-PRM is intrinsically robust on ProcessBench (+14.9 percentage points Detection Rate, +4.1 percentage points AUROC, averaged over three evaluation splits; see §4 for split selection) and is the only model among the six evaluated configurations (Table 3) whose Best-of-N reranking accuracy improves monotonically through $N = 128$ (§5).
- A two-round ALFWorld domain-shift diagnostic shows that objective-level robustness and data coverage are complementary, and that ARMOR-PRM's regularization makes it the most responsive variant to iterative hard-negative retraining (§6).

2 Related Work

Process Reward Models, Reward Hacking, and Outcome Supervision. PRMs are effective for mathematical reasoning and agentic tasks (Lightman et al., 2024; Wang et al., 2024; Setlur et al., 2025), but Gao et al. (2023) characterize the inverted-U collapse under optimization pressure, motivating robustness as a first-class concern. Skalse et al. (2022) formalize reward gaming as a structural property of proxy objectives, and Pan et al. (2022) show that misspecification compounds under optimization even when the proxy is well-calibrated, so overoptimization is an inherent risk rather than an implementation artifact. Automated process labels from Monte Carlo rollouts or self-consistency (Wang et al., 2024; Luo et al., 2024) improve scalability but do not resolve shortcut learning when annotation conflates surface fluency with logical validity. Recent diagnostic work (Zheng et al., 2025; Khalifa et al., 2025) exposes large variance in step-error detection and shows that current PRMs are sensitive to surface presentation rather than to logical content. ARMOR-PRM intervenes at training time to make the reward robust to such surface manipulation.

Test-Time Compute Scaling and Best-of-N Search. Inference-time compute scaling complements model scale. Best-of-N sampling with a reward model (Cobbe et al., 2021; Lightman et al., 2024) is the simplest instantiation, but its effectiveness degrades when the reward model is exploitable (Gao et al., 2023). A policy trained against a fixed verifier can exploit the verifier's blind spots instead of improving task performance (Gao et al., 2023); Best-of-N reranking inherits this failure mode, making reward robustness a prerequisite for reliable scaling (Snell et al., 2025).

Adversarial Robustness and Shortcut Learning. Adversarial training (Madry et al., 2017) addresses robustness via worst-case perturbations, but standard formulations target input-label classification accuracy rather than calibrated reward ordering under semantics-preserving perturbations. Lipschitz regularization and gradient penalties (Gulrajani et al., 2017) improve robustness in classification settings but target global Lipschitz constants, with overhead unsuitable for full LLM fine-tuning. For reward modeling, Bukharin et al. (2025) mine high-reward, low-quality responses at the outcome level; in process supervision, heuristic data augmentation (Wang et al., 2024) improves scalability without providing a checkable diagnostic on reward ordering. ARMOR-PRM instead targets step-level reward ordering with training-time, reward-head-local regularization and a checkable observed-neighborhood diagnostic, remaining practical at the 1.5B–7B scale.

3 ARMOR-PRM: Method and Theory

ARMOR-PRM minimizes systematic exploitability by training the reward model to preserve reward ordering under label-preserving surface perturbations within observed neighborhoods. Figure 1 gives an overview; full proofs are in Appendix F. We write f_ϕ for the PRM backbone and g_ψ for the scalar reward head, so the reward logit is $R_\theta(\tau) = g_\psi(h_\tau)$ where h_τ is the final-token hidden state; τ^+ and τ^- denote correct and flawed steps, and $\tilde{\tau}^-$ (resp. $\hat{\tau}^-$) denotes a training-observed (resp. arbitrary) adversarial variant. Full notation is listed in Appendix A.

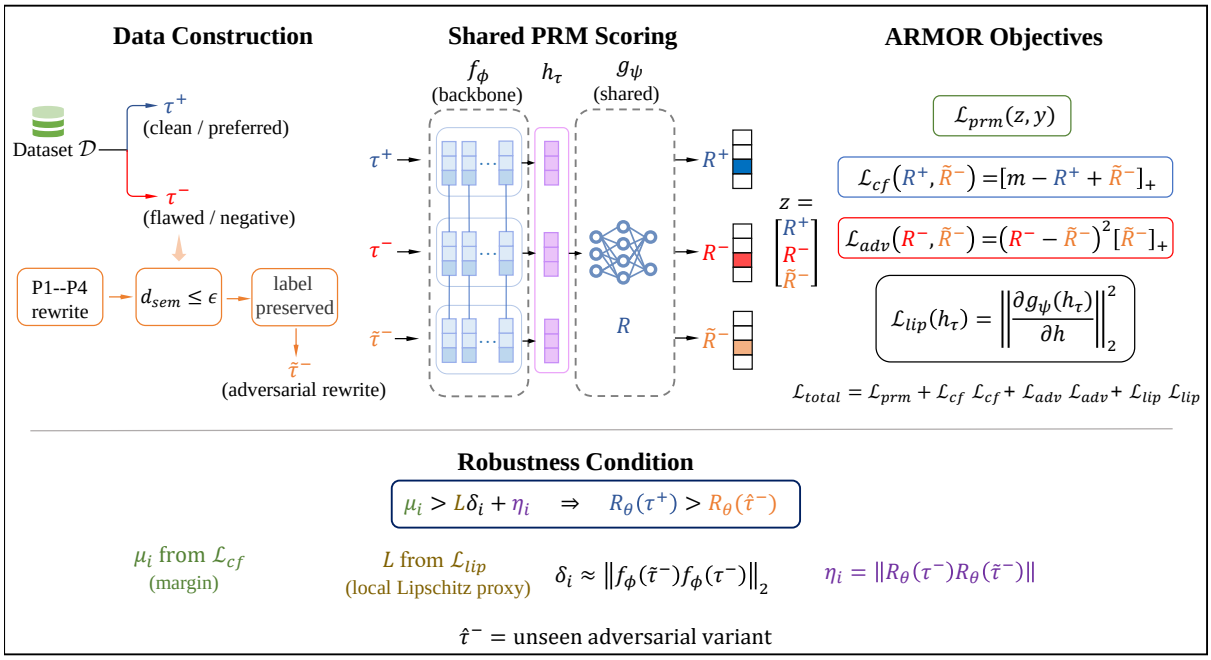

Figure 1: **Overview of ARMOR-PRM.** Correct steps τ^+, flawed steps τ^-, and adversarially rewritten flawed steps $\tilde{\tau}^-$ are scored by a shared PRM backbone f_ϕ and reward head g_ψ. The training objective combines step-label supervision $\mathcal{L}_{\text{prm}}$ with a counterfactual margin loss $\mathcal{L}_{\text{cf}}$, an adversarial invariance loss $\mathcal{L}_{\text{adv}}$, and a reward-head gradient penalty $\mathcal{L}_{\text{lip}}$. Each term controls one quantity in the observed-neighborhood diagnostic $\mu_i > L\delta_i + \eta_i$: $\mathcal{L}_{\text{cf}}$ controls the training margin γ (and hence the per-pair empirical margin μ_i), $\mathcal{L}_{\text{adv}}$ controls the invariance residual η, and $\mathcal{L}_{\text{lip}}$ controls the local Lipschitz proxy L. When the condition holds (90.2% of held-out pairs), a disguised flawed step cannot outscore a correct step within the training-observed neighborhood.

3.1 Formalizing Systematic Exploitability

PRMs are most vulnerable to *validity-preserving rhetorical manipulation*: confident phrasing, structured but logically irrelevant elaboration, or plausible-looking but fabricated intermediate formulas.

Let $\tau = (s_1, \ldots, s_T)$ denote a reasoning trajectory. We define the adversarial neighborhood through a *two-criterion gate*. A step s' must (i) be semantically close to s (cosine distance $\leq \epsilon$ under a frozen sentence encoder), and (ii) preserve the correctness label, as confirmed by an automated verifier:

$$\mathcal{N}_\epsilon(\tau) = \{\, \tau' \in \mathcal{T} \mid \max_t d_{\text{sem}}(s_t, s'_t) \leq \epsilon \;\wedge\; \mathcal{V}(s_t, s'_t) = 1 \} \tag{2}$$

The embedding filter cheaply prunes semantically distant candidates, and the verifier enforces label preservation on the remainder. Each perturbation modifies exactly one step per trajectory to isolate per-step attribution.

We instantiate $\mathcal{N}_\epsilon(\tau)$ via four perturbation families (referred to as P1–P4 in the rest of the paper): **P1** authoritative assertion, **P2** spurious verification, **P3** phantom execution, and **P4** verbose decomposition. Each family substitutes confident-sounding language for logical validity while preserving the correctness label. Domain-specific instantiations (math: A1–A6; agent variants) are listed in Appendix B. We verify the automated pipeline empirically in §4.

3.2 ARMOR-PRM: Three-Pronged Optimization

To enforce $R_\theta(\tilde{\tau}^-) < R_\theta(\tau^+)$ across observed P1–P4 manipulations, we adopt a standard autoregressive PRM. A pretrained backbone f_ϕ (e.g., Qwen2.5-Math-1.5B (Yang et al., 2024)) encodes the partial trajectory $(s_1, \ldots, s_t)$ with causal attention, and a scalar head g_ψ maps the final-token hidden state h_τ to a real-valued reward logit $R_\theta(\tau) = g_\psi(h_\tau) \in \mathbb{R}$ (Lightman et al., 2024), with positive values for correct steps and negative values for incorrect ones. The full parameter set is $\theta = \{\phi, \psi\}$. We optimize R_θ with the composite loss

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{prm}} + \lambda_{\text{cf}}\mathcal{L}_{\text{cf}} + \lambda_{\text{adv}}\mathcal{L}_{\text{adv}} + \lambda_{\text{lip}}\mathcal{L}_{\text{lip}} \tag{3}$$

where $\mathcal{L}_{\text{prm}}$ is Focal Loss (Lin et al., 2017) on step-wise binary classification applied to the logit

$R_\theta(\tau)$. Each regularizer controls one measurable quantity in the observed-neighborhood diagnostic $\mu_i > L\delta_i + \eta_i$ (§3.3): $\mathcal{L}_{\text{cf}}$ enforces the training margin γ between correct and disguised-wrong steps; $\mathcal{L}_{\text{adv}}$ drives the invariance residual η toward zero while preventing reward inflation on wrong steps; and $\mathcal{L}_{\text{lip}}$ penalizes the local Lipschitz constant L of g_ψ, reducing sensitivity to unseen representation shifts. We detail each below.

Counterfactual Margin Loss ($\mathcal{L}_{\text{cf}}$). $\mathcal{L}_{\text{cf}}$ enforces a minimum reward margin γ between correct steps and *adversarially disguised* incorrect steps, making it harder for rhetorically embellished errors to outscore logically valid steps:

$$\mathcal{L}_{\text{cf}} = \mathbb{E}_{(\tau^+,\tilde{\tau}^-)}\big[\max\big(0,\ \gamma - R_\theta(\tau^+) + R_\theta(\tilde{\tau}^-)\big)\big] \tag{4}$$

Here $\tilde{\tau}^-$ is an adversarial variant of a flawed step drawn from $\mathcal{N}_\epsilon(\tau^-)$, and τ^+ is a correct step from the *same problem*. The margin is therefore enforced within-problem, preventing the loss from using difficulty or length as a proxy for correctness.

Adversarial Invariance Loss ($\mathcal{L}_{\text{adv}}$). $\mathcal{L}_{\text{adv}}$ encourages invariance between a flawed step and its adversarial rewrite while anchoring the rewrite below the decision boundary ($R_\theta \le 0$):

$$\mathcal{L}_{\text{adv}} = \mathbb{E}_{\tau^-,\tilde{\tau}^-}\big[\big(R_\theta(\tau^-) - R_\theta(\tilde{\tau}^-)\big)^2\big] + \mathbb{E}_{\tilde{\tau}^-}\big[\max\big(0,\ R_\theta(\tilde{\tau}^-)\big)\big] \tag{5}$$

The first term enforces reward consistency between a flawed step τ^- and its surface-manipulated variant $\tilde{\tau}^-$. The second term, a ReLU hinge, anchors adversarial variants to the incorrect region ($R_\theta(\tilde{\tau}^-) \le 0$), ruling out the degenerate solution in which both scores drift upward together while their difference stays small.

Gradient Penalty on the Reward Head ($\mathcal{L}_{\text{lip}}$). $\mathcal{L}_{\text{adv}}$ only covers perturbation instances explicitly constructed during training (the P1–P4 instantiations). For continuous-level smoothness beyond these predefined attack patterns, we add a gradient penalty on the reward head:

$$\mathcal{L}_{\text{lip}} = \mathbb{E}_\tau\left[\left\|\frac{\partial g_\psi(h)}{\partial h}\bigg|_{h=h_\tau}\right\|_2^2\right] \tag{6}$$

This loss penalizes an empirical estimate of the local Lipschitz constant of g_ψ with respect to h_τ, so that small representation-space movements within observed neighborhoods produce only small reward changes. We use the 95th-percentile gradient norm as an empirical proxy for L. We validate the independent contributions of $\mathcal{L}_{\text{adv}}$ and $\mathcal{L}_{\text{lip}}$ in §5.2.

3.3 Observed-Neighborhood Diagnostic

Each of the three losses controls one measurable quantity: $\mathcal{L}_{\text{cf}}$ enforces the training margin γ, $\mathcal{L}_{\text{lip}}$ penalizes the local Lipschitz constant L of g_ψ, and $\mathcal{L}_{\text{adv}}$ drives the invariance residual η toward zero. The per-pair empirical margin $\mu_i = R_\theta(\tau_i^+) - R_\theta(\tilde{\tau}_i^-)$ and the backbone embedding shift δ_i are measured directly from the evaluation checkpoint. Together they yield a checkable *empirical sufficient condition* (not a certified robustness guarantee) on whether a flawed step can outscore its correct counterpart within the observed adversarial neighborhood.

Proposition 1 (Observed-Neighborhood Diagnostic)**.** *Let τ^+ be a correct step, τ^- a flawed step, and $\tilde{\tau}^- \in \mathcal{N}_\epsilon(\tau^-)$ a training-observed adversarial variant with $\|f_\phi(\tilde{\tau}^-) - f_\phi(\tau^-)\|_2 \le \delta$. Let $\eta = |R_\theta(\tau^-) - R_\theta(\tilde{\tau}^-)|$ be the adversarial invariance residual, and let $\mu = R_\theta(\tau^+) - R_\theta(\tilde{\tau}^-)$ be the measured margin for this pair. If (i) $\mu \ge 0$ and (ii) g_ψ is locally L-Lipschitz on the hidden-state space, then for any $\hat{\tau}^- \in \mathcal{N}_\epsilon(\tau^-)$ with $\|f_\phi(\hat{\tau}^-) - f_\phi(\tau^-)\|_2 \le \delta$:*

$$R_\theta(\hat{\tau}^-) - R_\theta(\tau^+) \le L\delta + \eta - \mu \tag{7}$$

Corollary 1 (Sufficient Non-Inversion Condition within Empirical Neighborhood)**.** *If $\mu_i > L\delta_i + \eta_i$, then for every correct step τ^+ and every adversarially disguised flawed step $\hat{\tau}^-$ with $\|f_\phi(\hat{\tau}^-) - f_\phi(\tau^-)\|_2 \le \delta_i$ (i.e., within the 95th-percentile embedding-shift range observed over P1–P4 training variants):*

$$R_\theta(\hat{\tau}^-) < R_\theta(\tau^+) \tag{8}$$

All four quantities are measured from the evaluation checkpoint: L is the 95th-percentile gradient norm of g_ψ, δ_i is the per-pair backbone embedding shift under P1–P4, η_i is the measured adversarial invariance residual, and $\mu_i = R_\theta(\tau_i^+) - R_\theta(\tilde{\tau}_i^-)$ is the per-pair empirical margin. The condition holds on 90.2% of held-out pairs (§5.2); we discuss its scope and limitations in the Limitations section. Full proofs are in Appendix F.

Corollary 2 (Exploitability Bound within $\mathcal{N}_\epsilon$)**.** *Under the conditions of Proposition 1:*

$$\mathcal{E}(R_\theta) \;\le\; \mathbb{E}_i\Big[\big[L\delta_i + \eta_i - \mu_i\big]_+\Big] \tag{9}$$

and $\mathcal{E}(R_\theta) = 0$ *for pairs satisfying* $\mu_i > L\delta_i + \eta_i$ *within the empirical neighborhood.*

$\mathcal{L}_{\text{cf}}$ increases γ (and hence μ_i), $\mathcal{L}_{\text{adv}}$ decreases η, and $\mathcal{L}_{\text{lip}}$ decreases L; each loss term thus directly shrinks the right-hand side. The full proof is in Appendix F.

4 Experimental Setup

Training, Baselines, and Variants. We use Qwen2.5-Math-1.5B (Yang et al., 2024) as the PRM backbone and train on PRM800K (Lightman et al., 2024) with $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{prm}} + 0.5\,\mathcal{L}_{\text{cf}} + 0.5\,\mathcal{L}_{\text{adv}} + 0.1\,\mathcal{L}_{\text{lip}}$ and margin $\gamma = 0.3$. Two external baselines, Math-Shepherd (Wang et al., 2024) (7B) and ORM (1.5B), differ from ARMOR-PRM in backbone scale, training data, or objective; we include them for context only. All internal variants share the same backbone and data, differing only in loss configuration (Table 1). Each internal variant is trained with 3 independent random seeds (seeds 1–3); reported values are mean ± standard deviation across seeds. Tables 2 and 3 report results from separate training runs under the same protocol; minor numerical differences between shared rows reflect seed-level variance.

Variant	Loss	Pairs	Role
Vanilla PRM	$\mathcal{L}_{\text{prm}}$	—	Unregularized reference
$\mathcal{L}_{\text{prm}}$+$\mathcal{L}_{\text{cf}}$ (nat.)	$\mathcal{L}_{\text{prm}}$+$\mathcal{L}_{\text{cf}}$	nat.	Pairing strategy ablation
Pairwise PRM	$\mathcal{L}_{\text{prm}}$+$\mathcal{L}_{\text{cf}}$	adv.	Scoring formulation ablation
CF-Adv PRM	$\mathcal{L}_{\text{prm}}$+$\mathcal{L}_{\text{cf}}$	adv.+cf	Counterfactual pair ablation
$\mathcal{L}_{\text{prm}}$+$\mathcal{L}_{\text{adv}}$	$\mathcal{L}_{\text{prm}}$+$\mathcal{L}_{\text{adv}}$	—	Component accumulation
$\mathcal{L}_{\text{prm}}$+$\mathcal{L}_{\text{cf}}$+$\mathcal{L}_{\text{adv}}$	$\mathcal{L}_{\text{prm}}$+$\mathcal{L}_{\text{cf}}$+$\mathcal{L}_{\text{adv}}$	adv.	Component accumulation
ARMOR-PRM	$\mathcal{L}_{\text{prm}}$+$\mathcal{L}_{\text{cf}}$+$\mathcal{L}_{\text{adv}}$+$\mathcal{L}_{\text{lip}}$	adv.	Full objective (ours)

Table 1: Internal variant family. All variants share the Qwen2.5-Math-1.5B backbone and PRM800K training data; they differ only in loss terms and pairing strategy. The **Pairs** column denotes the type of flawed step used in $\mathcal{L}_{\text{cf}}$: *nat.* = natural flawed steps without adversarial perturbation; *adv.* = adversarially perturbed flawed steps (with the within-problem pairing of §3.2); *adv.+cf* = adversarially perturbed flawed steps augmented with explicit counterfactual pair sampling. CF-Adv PRM is used only as a baseline in the ALFWorld experiment (§6); it is not reported in Tables 2 or 3.

RL Setup. The policy backbone is Qwen2.5-Math-1.5B-Instruct, trained with GRPO (Shao et al., 2024) for 200 steps (lr=2×10^{-5}, KL-β=0.005, centered advantage normalization, K=4 rollouts per problem) using LoRA (rank 32, α=32, applied to $q/k/v/o$-proj) on ALFWorld household navigation tasks (unseen-134 split). For the iterative retraining experiment (§6.1), Round-2 launches a new 200-step GRPO run with PRMs retrained on Round-1 on-policy hard negatives. Full LoRA and GRPO hyperparameters appear in Appendix A.

Perturbations, Validity, and Isolation. DeepSeek-V4-Flash generates adversarial variants (up to six per step, one per A1–A6 type), which are then filtered by a two-criterion gate: semantic distance $\le \epsilon$ under a frozen `all-mpnet-base-v2` sentence encoder and label verification by Qwen-Plus (prompt templates in Appendix G). This pipeline restricts perturbations to surface-form manipulation while preserving the original correctness label. A 300-pair manual audit, stratified by perturbation type, reports 96.7% label preservation among accepted variants; the gate rejects 8.3% of generated variants, with per-type preservation rates between 89.2% and 98.1%.

For evaluation, test attacks B1–B3 are drawn from a *mixed-source* pool. Each incorrect step is attacked by (i) a GPT-4o-generated variant prompted independently with the same template intent, and (ii) a manually constructed variant written from only the category label and the label-preservation constraint. The two sources are sampled in equal proportion and are disjoint from the DeepSeek-V4-Flash generator used in training.

For benchmark isolation, the `math` split overlaps PRM800K at 78.8% problem level, so we exclude it from reported averages. The remaining three splits (`gsm8k`, `olympiadbench`, `omnimath`) are used for evaluation.

5 PRM Robustness Evaluation

5.1 Intrinsic Evaluation: Clean and Attacked Accuracy

Setup. We evaluate on ProcessBench (Zheng et al., 2025) (4 splits; 3,064 problems, 23,203 steps) under clean and attacked conditions. Attacked evaluation applies three disjoint test attack types (B1–B3) to incorrect steps; each type draws from a mixed-source pool of GPT-4o-generated and manually constructed variants in equal proportion, both disjoint from the DeepSeek-V4-Flash training generator (Appendix G). ASR_B is averaged over the three types (per-type worst-case is omitted for space). We report Clean F1, AUROC, ASR_B

Method	Clean F1↑	AUROC↑	Atk F1↑	ASR_B↓	Det. Rate↑	PR-AUC_+↑
Evaluation splits: GSM8K + OlympiadBench + OmniMath (math split excluded; see §4):						
Math-Shepherd (7B) (Wang et al., 2024)	76.20	79.70	78.80	21.10	32.40	78.50
ORM (1.5B)	67.80	73.00	69.40	8.30	46.30	74.20
Vanilla PRM (1.5B)	78.69$\pm$0.37	80.95$\pm$0.43	82.35$\pm$0.34	2.28$\pm$0.12	57.03$\pm$0.81	82.82$\pm$0.39
$\mathcal{L}_{\text{prm}}$+$\mathcal{L}_{\text{cf}}$ (nat. pairs, 1.5B)	75.07$\pm$0.38	74.10$\pm$0.41	75.81$\pm$0.38	3.30$\pm$0.17	39.71$\pm$0.73	75.56$\pm$0.34
Pairwise PRM (1.5B)	78.89$\pm$0.55	81.88$\pm$0.38	81.17$\pm$0.32	4.10$\pm$0.13	54.60$\pm$0.79	84.06$\pm$0.31
ARMOR-PRM (1.5B, Ours)	**79.91**$\pm$0.29	**85.08**$\pm$0.35	82.33$\pm$0.25	**1.33**$\pm$0.10	**71.88**$\pm$0.73	**88.18**$\pm$0.33

Table 2: PRM intrinsic evaluation on ProcessBench (three splits; the math split is excluded, see §4). External baselines (Math-Shepherd, ORM) are shown for context only: backbone scale, training data, and objective are uncontrolled, and we draw no performance claims from them. ASR_B is attack success rate (lower is better).

(the fraction of attacked incorrect steps flipped to "correct"), and Detection Rate. Thresholds are calibrated on a held-out 10% dev split. **Detection Rate** is the recall of incorrect steps under attack: $|\mathcal{D}|/|\mathcal{W}|$, where $\mathcal{D} = \{s : \hat{y}_{\text{wrong}}(s) = 1 \wedge y(s) = \text{wrong}\}$, $\mathcal{W} = \{s : y(s) = \text{wrong}\}$, and $\hat{y}_{\text{wrong}}(s) = \mathbf{1}[R_\theta(s) < \tau]$.

Results. Table 2 summarizes the intrinsic results on ProcessBench. ARMOR-PRM improves AUROC by 4.13 percentage points and Detection Rate by 14.85 percentage points over Vanilla PRM, and reduces ASR_B from 2.28% to 1.33%. Test attacks combine GPT-4o-generated and manually constructed variants from sources disjoint from training, so the gains reflect resistance to the attack pattern rather than familiarity with a particular generation style. The elevated ASR_B of $\mathcal{L}_{\text{prm}}$+$\mathcal{L}_{\text{cf}}$ (nat. pairs) at 3.30% confirms that ranking on natural pairs reinforces shortcuts rather than correcting them. External baselines are reported for context only, as backbone scale, data, and objectives are uncontrolled.

5.2 Ablation Study

Table 3 reports the six evaluated configurations trained on PRM800K (1.5B backbone) and evaluated on ProcessBench. **Layer 1** (rows 2–3) isolates the pairing strategy for $\mathcal{L}_{\text{cf}}$; **Layer 2** (rows 4–6) cumulatively ablates loss components with adversarial pairs held fixed.

Results. *Layer 1: pairing strategy.* $\mathcal{L}_{\text{cf}}$ with natural pairs causes substantial degradation (AUROC: 80.94→74.08), and switching to adversarial pairs recovers near-Vanilla performance. This isolates pairing strategy as the decisive factor. Adversarial pairs alone do not reduce ASR_B below Vanilla (4.10 vs. 2.24), so $\mathcal{L}_{\text{adv}}$'s anchoring is necessary to prevent upward score drift.

$\mathcal{L}_{\text{prm}}$	$\mathcal{L}_{\text{cf}}$	$\mathcal{L}_{\text{adv}}$	$\mathcal{L}_{\text{lip}}$	Pairs	Clean F1	AUROC	ASR_B↓	Det. Rate
Layer 1: pairing strategy for $\mathcal{L}_{\text{cf}}$ (all other components fixed)								
✓				—	78.72$\pm$0.39	80.94$\pm$0.48	2.24$\pm$0.11	57.01$\pm$0.82
✓	✓			nat.	75.10$\pm$0.42	74.08$\pm$0.55	3.26$\pm$0.17	39.78$\pm$0.91
✓	✓			adv.	78.89$\pm$0.40	81.88$\pm$0.46	4.10$\pm$0.13	54.60$\pm$0.87
Layer 2: component accumulation (adv. pairs fixed for $\mathcal{L}_{\text{cf}}$)								
✓		✓		—	79.15$\pm$0.44	82.88$\pm$0.51	10.30$\pm$0.63	62.36$\pm$1.02
✓	✓	✓		adv.	79.12$\pm$0.41	83.02$\pm$0.49	5.80$\pm$0.38	60.88$\pm$0.95
✓	✓	✓	✓	adv.	**79.93**$\pm$0.35	**85.10**$\pm$0.43	**1.31**$\pm$0.08	**71.93**$\pm$0.88

Table 3: Ablation of ARMOR-PRM loss components on ProcessBench. **Layer 1** (rows 2–3): natural pairs cause substantial degradation, and adversarial pairs recover Vanilla-level performance; pairing strategy is therefore the decisive factor. **Layer 2** (rows 4–6): cumulative ablation with adversarial pairs held fixed, reaching the full ARMOR-PRM configuration.

Layer 2: component accumulation. $\mathcal{L}_{\text{adv}}$ alone is itself a failure mode (ASR_B rises to 10.30%), and combining $\mathcal{L}_{\text{cf}}$+$\mathcal{L}_{\text{adv}}$ resolves it. Adding $\mathcal{L}_{\text{lip}}$ yields the largest single Detection Rate gain (+11.05 percentage points). The full objective achieves the lowest ASR_B (1.31%) and the highest Detection Rate (71.93). Per-domain breakdowns and training loss dynamics are in Appendices C and D.

Observed-neighborhood diagnostic. We check the diagnostic condition $\mu_i > L\delta_i + \eta_i$ on each held-out pair. ARMOR-PRM satisfies it on 90.2% of pairs, compared with 54.5% for Vanilla PRM. The μ_{p5} column in Appendix E summarizes the empirical margin distribution (5th percentile) and is reported for interpretability, not as input to the bound. The full per-pair L, δ, η breakdown is in Appendix E.

5.3 Best-of-N Scaling

We next test whether ARMOR-PRM avoids the inverted-U collapse under increasing optimization pressure, in line with the observed-neighborhood diagnostic verified in §5.2.

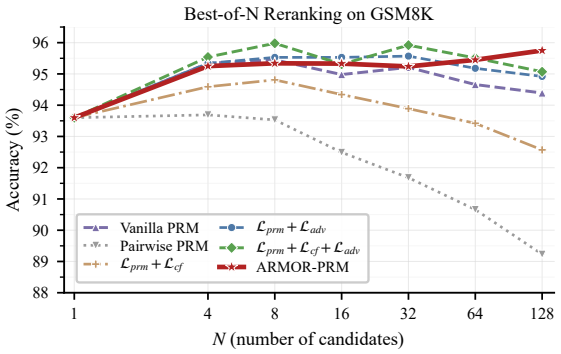

Figure 2: Best-of-N reranking accuracy on GSM8K as a function of N. Vanilla PRM and several ablated variants show the inverted-U collapse under larger candidate budgets, whereas ARMOR-PRM improves monotonically through $N = 128$.

Setup. We use Qwen2.5-Math-1.5B-Instruct (temperature 0.7) to generate N candidate solutions for each problem in a 300-problem evaluation subset of the GSM8K (Cobbe et al., 2021) test set, with $N \in \{1, 4, 8, 16, 32, 64, 128\}$. Each PRM reranks the candidates, and we measure the true pass@1 accuracy of the top-ranked solution (verified against ground-truth answers). Accuracy at each N is averaged over $n_{\text{trials}} = 5$ independent subsampling runs, and reported values are point estimates.

Results. Figure 2 shows that Vanilla PRM follows the expected inverted-U pattern: it peaks at $N=8$ (95.45%) and drops to 94.39% at $N=128$. ARMOR-PRM, by contrast, improves monotonically from 93.60% at $N=1$ to 95.75% at $N=128$. It is the only model among the six evaluated configurations (Table 3) to do so, and it ends with a 1.36-point advantage over Vanilla at the largest budget (bootstrap 95% CI: [0.7, 2.0]; McNemar $p<0.05$). Ablated variants with natural-pair losses collapse the most severely.

6 Reward Inflation Diagnostic in Agentic Settings

In agentic closed-loop settings, a deployed policy keeps producing *new* exploitation patterns that no static dataset contains. This exposes a second robustness dimension beyond objective design: *data coverage*. We study two questions. **RQ1**: do all PRMs inflate under domain shift before corrective data is available? **RQ2**: does a PRM's regularization structure affect how efficiently it absorbs on-policy hard negatives?

PRM	Self $\Delta_{\text{gap}}\downarrow$	Off-diag mean$\downarrow$	Cross-policy infl.$\downarrow$
Vanilla PRM	0.638	0.521	(anchor)
ARMOR-PRM	0.822	0.585	+0.390
Pairwise PRM	0.825	**0.502**	**+0.196**
CF-Adv PRM	0.791	0.521	+0.610

Table 4: ALFWorld Round-1 cold-start hack metrics (step 200, SR=0 universally). All PRMs exhibit substantial reward inflation. Cross-policy metrics identify Pairwise PRM as the most conservative one. The full cross-PRM scoring matrix is in Appendix H.

6.1 ALFWorld: Two-Round Iterative Retraining

We study these questions in ALFWorld (Shridhar et al., 2021) via a two-round closed-loop experiment that uses the same five-lane GRPO protocol on household navigation tasks (unseen-134 split, 50 held-out tasks per checkpoint).

Setting. We deliberately apply a math-specialized policy to a navigation domain to set up a *domain-shift stress test*. Since the policy has no navigation capability, SR equals zero universally, so every non-zero reward score is pure inflation with no task-completion signal to confound it.

Round-1: Cold-Start Hack Diagnosis (RQ1). Table 4 reports self-scoring Δ_{gap} and cross-PRM metrics at step 200. All PRMs exhibit substantial inflation (0.638–0.825), so objective-level robustness alone cannot prevent reward inflation when policies hit exploitation patterns that the static training data does not cover. Among the four PRMs, Pairwise PRM has the lowest off-diagonal mean (0.502) and is the most conservative under cold-start domain shift; ARMOR-PRM has the highest off-diagonal mean (0.585). This confirms that the advantage of ARMOR-PRM's regularization lies not in cold-start conservativeness but in how efficiently it absorbs corrective data, which we examine in Round-2. The Round-1 results validate data coverage as an independent second dimension of robustness.

Round-2: Iterative Retraining with On-Policy Hard Negatives (RQ2). Round-1 yields 430 failed rollouts. We retrain each PRM via LoRA continuation on the original 14,134 step labels augmented with 4,300 hard-negative steps (18,434 total), and then evaluate in two configurations.

B1 (Held-Out Generalization). We withhold 20% of the rollouts (86) as an OOD test set. Ta-

PRM	R1 Δ_{gap}	R2 Δ_{gap} (OOD)	Reduction
Vanilla PRM	0.622	0.087	-86%
ARMOR-PRM	0.748	**0.053**	$\mathbf{-93\%}$
Pairwise PRM	0.554	0.078	-86%
CF-Adv PRM	0.640	0.117	-82%

Table 5: Round-2 held-out generalization (B1): Δ_{gap} on 86 OOD failed rollouts that no retrained PRM has seen. All PRMs improve substantially, and ARMOR-PRM benefits the most (-93%). R1 values are recomputed on this 86-rollout held-out subset and therefore differ from the full Round-1 self-scoring values in Table 4.

PRM	s040	s080	s120	s160	s200	5-step mean↓	R1→R2
Vanilla PRM	0.036	0.104	0.104	0.077	0.056	0.075	-88%
ARMOR-PRM	**0.036**	**0.025**	**0.066**	**0.083**	**0.068**	**0.056**	$\mathbf{-93\%}$
Pairwise PRM	0.110	0.142	0.059	0.249	0.189	0.150	-82%
CF-Adv PRM	0.203	0.365	0.365	0.364	0.035	0.266	-66%

Table 6: Round-2 closed-loop GRPO (B2): reward-inflation gap across five checkpoints after hard-negative retraining. Lower is better.

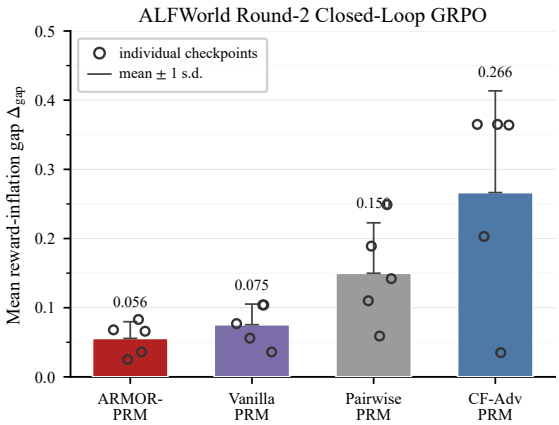

Figure 3: Round-2 closed-loop GRPO (B2): 5-checkpoint mean Δ_{gap} with ± 1 s.d. error bars.

ble 5 shows that all four PRMs improve substantially. **ARMOR-PRM reaches the lowest post-retraining Δ_{gap} (0.053)**, versus 0.087/0.078/0.117 for Vanilla/Pairwise/CF-Adv PRM.

B2 (Closed-Loop GRPO with Retrained PRMs). Whereas B1 tests generalization to held-out instances of the *same* Round-1 hack patterns, B2 tests resistance to *new* patterns generated under fresh RL pressure, which is a strictly harder evaluation. We launch a fresh 5-lane GRPO run (200 steps) and report the 5-checkpoint mean Δ_{gap} (Table 6, Figure 3). **ARMOR-PRM achieves the lowest 5-step mean** (0.056). It shows neither the late-stage spike seen in Pairwise PRM (peak 0.249, mean 0.150) nor the sustained high plateau of CF-Adv PRM (mean 0.266). Vanilla PRM's higher mean (0.075) is driven by mid-run inflation at steps 80–120, even though its final value is comparable. The Round-2 ranking is also *reversed* from Round-1: Pairwise PRM, which inflated the least in Round-1, performs the worst in B2. Cold-start conservativeness therefore does not predict closed-loop stability after retraining.

Why ARMOR-PRM Responds Best. Both ARMOR-PRM and Vanilla PRM share an absolute scoring target, yet ARMOR-PRM benefits more from hard-negative retraining. This is consistent with $\mathcal{L}_{\text{adv}}$ and $\mathcal{L}_{\text{lip}}$ smoothing the reward landscape so that corrective gradients propagate more uniformly. Pairwise PRM lacks an absolute target, so hard negatives can only shift relative rankings, and the score surface stays susceptible to late-stage drift.

7 Conclusion

We presented ARMOR-PRM, a training objective for PRM robustness against surface-form reward hacking. It combines a counterfactual margin loss, an adversarial invariance loss, and a reward-head gradient penalty. On ProcessBench, ARMOR-PRM improves attacked-step detection and AUROC over a vanilla PRM trained with the same backbone and data, while preserving clean performance. In Best-of-N reranking, it is the only evaluated variant whose accuracy improves monotonically through $N = 128$. An ALFWorld domain-shift diagnostic further shows that every evaluated PRM exhibits reward inflation under cold-start domain shift, but after iterative hard-negative retraining ARMOR-PRM achieves the largest reduction in reward-inflation gap. Robust objectives and on-policy data coverage are therefore complementary components, not substitutes.

The scope of these findings is deliberately limited. The observed-neighborhood condition $\mu_i > L\delta_i + \eta_i$ is an empirical diagnostic, not a certified guarantee against arbitrary adversarial rewrites, and the ALFWorld experiment measures reward inflation rather than task success. Future work should evaluate the approach under stronger adaptive attacks, larger backbones, and closed-loop settings where reductions in proxy reward inflation can be tied to task performance.

Limitations

Perturbation Coverage and Theoretical Scope. The observed-neighborhood diagnostic assumes that the six-type taxonomy (A1–A6) covers the relevant attack surface. A policy under prolonged RL may discover exploits outside this set, in which case $\mu_i > L\delta_i + \eta_i$ provides no guarantee and the reported Detection Rate and ASR_B gains would not extend. The gradient penalty only bounds the *local* Lipschitz constant within the empirically observed embedding-shift range δ; hidden-state shifts beyond the 95th-percentile training value fall outside this bound, so the diagnostic may overstate robustness in that regime. Our experiments are limited to 1.5B–7B backbones, and the regularization benefits may not hold at larger scales.

Scope of the Agent Domain Findings. The ALFWorld experiment uses a math-specialized policy on navigation tasks so that SR equals zero universally, which means every non-zero proxy score is unambiguously reward inflation. The findings are therefore scoped to the inflation-and-correction regime under distribution shift, not to task-level performance. Whether correction efficiency in the zero-SR regime transfers to settings where the policy actually succeeds is not established: a non-zero SR mixes genuine task progress with inflation, and the gradient dynamics of hard-negative retraining may differ accordingly.

Data Pipeline and Domain Scope. The two-criterion gate reports 96.7% label preservation; the remaining ≈3.3% mislabeled variants could bias the counterfactual margin loss if they are concentrated in specific perturbation types. All results are established on PRM800K and ProcessBench within the math reasoning domain. Transfer to other step-level reasoning domains (e.g., code generation) is not tested, and the perturbation taxonomy may not capture the dominant shortcuts in those settings. Total compute was approximately 250–300 GPU-hours on A100-class GPUs.

Ethical Considerations

ARMOR-PRM is intended to make AI reasoning systems more reliable by reducing surface-form reward hacking. The main dual-use risk is that the perturbation taxonomy (A1–A6) could be repurposed to craft more effective reward-hacking strategies against deployed models; we therefore withhold full prompt templates and release only representative transformations. A second risk is overconfidence: the observed-neighborhood diagnostic holds only within the empirically observed perturbation range, and treating the 90.2% pass rate as a broad safety guarantee may lead deployers to underestimate residual attack surface under adaptive adversaries or out-of-distribution domains. The data used in this work do not contain personally identifying information, unique personal identifiers, or offensive content, so no anonymization procedure was required.

References

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

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, and 1 others. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Mengnan Du, Fengxiang He, Na Zou, Dacheng Tao, and Xia Hu. 2023. Shortcut learning of large language models in natural language understanding. *Communications of the ACM*, 67(1):110–120.

Leo Gao, John Schulman, and Jacob Hilton. 2023. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, volume 202, pages 10835–10866. PMLR.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. 2017. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

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

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike,

John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *International Conference on Learning Representations*, pages 39578–39601.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007.

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

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, abs/1706.06083.

Alexander Pan, Kush Bhatia, and Jacob Steinhardt. 2022. The effects of reward misspecification: Mapping and mitigating misaligned models.

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. 2025. Rewarding progress: Scaling automated process verifiers for llm reasoning. In *International Conference on Learning Representations*, volume 2025, pages 60808–60838.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Cote, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2021. Alfworld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*.

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

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2025. Scaling llm test-time compute optimally can be more effective than scaling parameters for reasoning. In *International Conference on Learning Representations*, pages 10131–10165.

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

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, and 1 others. 2024. Qwen2. 5-math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*.

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

A Hyperparameter Details

PRM Training. All PRMs are trained for 3 epochs with batch size 32, learning rate 2×10^{-5} (linear warmup over 5% of steps), and weight decay 0.01. The perturbation radius ϵ is set to the 95th percentile of $d_{\text{sem}}(\tau, \tau')$ over training pairs ($\epsilon \approx 0.15$ in cosine distance). A stratified 10% dev split of PRM800K is held out for threshold calibration.

RL Training. Policy backbone: Qwen2.5-Math-1.5B-Instruct with LoRA (Hu et al., 2022) (rank 32 for ALFWorld; α=32, dropout 0.05, applied to $q/k/v/o$-proj). GRPO: PPO-clip ϵ=0.2, K=4 rollouts per problem (group size 4), K_{epochs}=4, temperature 0.7, centered advantage normalization without std division, `save_every`=40. *ALFWorld Round-1*: household navigation tasks (unseen-134, 200 steps, 5 lanes), lr=2×10^{-5}, KL-β=0.005; PRMs are domain-fine-tuned ALFWorld checkpoints. *ALFWorld Round-2*: LoRA continuation of the Round-1 PRM checkpoints on the augmented training set (14,134 original + 4,300 hard-negative steps = 18,434 total), followed by a 200-step GRPO re-run with the retrained PRMs evaluated at five checkpoints (steps 40–200).

Notation. Table 7 lists the symbols used throughout §3. The key diagnostic condition is $\mu_i > L\delta_i + \eta_i$.

B Perturbation Taxonomy (A1–A6)

We define six perturbation types that each target a specific shortcut in PRM training data. Every type preserves the underlying logical (in)correctness of the step. Table 8 maps the four universal categories P1–P4 to their domain-specific instantiations.

- **A1: Authoritative Preamble Injection.** Prepend a logically vacuous but authoritative

Symbol	Meaning
τ, τ^+, τ^-	Trajectory; correct/flawed step
$\tilde{\tau}^-,\ \hat{\tau}^-$	Training adversarial variant; arbitrary variant
$f_\phi,\ g_\psi$	PRM backbone; reward head
h_τ	Final-token hidden state of τ
$R_\theta(\tau){=}g_\psi(h_\tau)$	Scalar reward logit
$\mathcal{N}_\epsilon(\tau)$	Adversarial neighborhood (radius ϵ)
$\mathcal{L}_{\text{prm}}$	Step-label classification loss
$\mathcal{L}_{\text{cf}}, \mathcal{L}_{\text{adv}}, \mathcal{L}_{\text{lip}}$	Counterfactual / invariance / Lipschitz loss
γ	Training margin in $\mathcal{L}_{\text{cf}}$
μ_i	Per-pair empirical margin $R_\theta(\tau_i^+){-}R_\theta(\tilde{\tau}_i^-)$
δ_i	Per-pair backbone embedding shift
η_i	Per-pair adversarial invariance residual
L	Local Lipschitz constant of g_ψ

Table 7: Notation used in §3.

connector (e.g., "By the triangle inequality, it follows directly that…") that mimics theorem-invoking language without taking part in the derivation.

- **A2: Pseudo-Systematic Decomposition.** Rewrite a single flawed step as a numbered multi-part derivation that mimics a systematic proof. The original error is distributed across sub-steps, but no new reasoning is introduced.
- **A3: Spurious Symbolic Augmentation.** Insert syntactically valid but semantically inert mathematical elements (unused variables, unchecked theorem citations, redundant equations) so that symbol density goes up without changing the logical content.
- **A4: Fabricated Intermediate Verification.** Insert a logically irrelevant sanity-check sentence in the middle of the derivation to create the illusion of a correctness check while preserving the original error.
- **A5: Pseudo-Self-Correction Injection.** Insert a fake self-correction passage that appears to catch and fix a mistake on the surface, but ultimately reconfirms the original erroneous conclusion.
- **A6: Confident Rephrasing.** Rewrite the step in a declarative, assertive style ("clearly", "obviously", "it follows immediately"), so that hedging language is swapped for confidence markers while the core error is preserved verbatim.

C Per-Domain Results

Table 9 breaks down Clean F1 and ASR_B across all ProcessBench splits. The `math` split is excluded from the reported averages (see §4) but is shown here for completeness. ARMOR-PRM achieves the best or tied-best Clean F1 across all three evaluation splits and the lowest ASR_B on all three. ASR_B is lowest on `olympiadbench` for every method (0.48–7.50%), which suggests that competition-level errors are harder to disguise with rhetorical decoration.

D Training Loss Dynamics

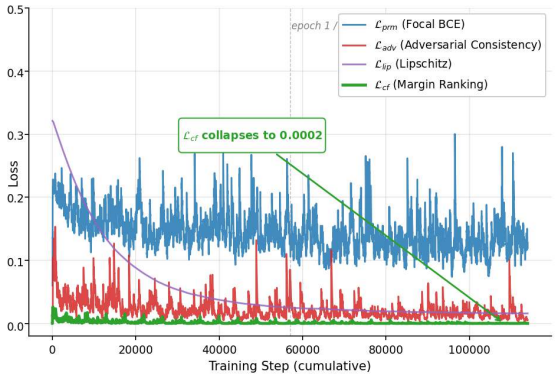

Figure 4: Training dynamics for ARMOR-PRM loss components. $\mathcal{L}_{\text{cf}}$ saturates quickly with adversarial negatives (the robustness constraint is met early), whereas $\mathcal{L}_{\text{adv}}$ and $\mathcal{L}_{\text{lip}}$ stay active throughout training.

E Robustness Condition Verification Details

Table 10 reports the full component breakdown for the observed-neighborhood diagnostic summarized in §5.2. L is the 95th-percentile gradient norm of g_ψ, δ is the 95th-percentile backbone embedding shift under P1–P4 perturbations measured over training-observed variants, η is the adversarial invariance residual at the training optimum, and μ_{p5} is the 5th-percentile empirical margin μ_i across held-out pairs.

F Proofs for Observed-Neighborhood Diagnostic

Proof of Proposition 1. Let τ^+ be a correct step, let τ^- be a flawed step, and let $\hat{\tau}^- \in \mathcal{N}_\epsilon(\tau^-)$ be any adversarial variant with $\|f_\phi(\hat{\tau}^-) - f_\phi(\tau^-)\|_2 \le \delta$.

Step 1 (Lipschitz bound on reward head). By condition (ii), g_ψ is L-Lipschitz on the hidden-state space:

$$\begin{aligned} R_\theta(\hat{\tau}^-) &= g_\psi(f_\phi(\hat{\tau}^-)) \\ &\le g_\psi(f_\phi(\tau^-)) + L \cdot \left\|f_\phi(\hat{\tau}^-) - f_\phi(\tau^-)\right\|_2 \\ &\le R_\theta(\tau^-) + L\delta. \end{aligned} \tag{10}$$

Universal (P)	Math-Specific (A1–A6)	Agent-Specific	Scope
P1: Authoritative Assertion	A1: Authoritative Preamble Injection	Action-Justification Prefix	General
P1: Authoritative Assertion	A6: Confident Rephrasing	Assertive Action Claim	General
P2: Spurious Verification	A4: Fabricated Intermediate Verification	Self-Confirmation of Action Success	General
P2: Spurious Verification	A5: Pseudo-Self-Correction Injection	Fabricated Output Verification	General
P3: Phantom Execution	A3: Spurious Symbolic Augmentation	Phantom Environment Interaction	Math / Agent
P4: Verbose Decomposition	A2: Pseudo-Systematic Decomposition	Planning Monologue Substitution	Math / Agent

Table 8: Mapping from universal perturbation categories P1–P4 to math-specific instantiations A1–A6 and agent-specific variants. The "Scope" column indicates whether the pattern is domain-general or arises mainly in math or agent settings. All types are filtered by the two-criterion gate ($d_{\text{sem}} \leq \epsilon$ and $\mathcal{V} = 1$) before being used in training.

	gsm8k		math		olympiad		omnimath	
Method	F1	ASR	F1	ASR	F1	ASR	F1	ASR
Vanilla PRM	84.36	3.74	81.52	3.20	79.72	0.75	72.97	1.67
$+\mathcal{L}_{\text{cf}}$ (nat. pairs)	80.36	19.65	78.65	16.42	76.91	7.50	68.32	20.14
$+\mathcal{L}_{\text{adv}}$	84.17	4.35	81.68	3.47	80.02	1.59	73.86	3.25
$+\mathcal{L}_{\text{cf}} + \mathcal{L}_{\text{adv}}$	83.22	4.37	82.69	3.75	79.92	1.30	73.48	3.01
ARMOR-PRM (Ours)	**85.73**	**1.71**	**83.39**	**2.31**	**80.36**	**0.48**	**74.91**	**1.82**

Table 9: Per-domain Clean F1 and ASR_B on ProcessBench. The `math` split is excluded from the reported averages (see §4) and is shown here for reference only. Lower ASR indicates stronger robustness.

Model	L	δ	η	**RHS summary**	μ_{p5}	**Pass %**
Vanilla PRM	0.55	0.28	0.52	0.67	0.11	54.5
Pairwise PRM	0.60	0.30	0.54	0.72	−0.58	37.2
$\mathcal{L}_{\text{prm}}$+$\mathcal{L}_{\text{cf}}$+$\mathcal{L}_{\text{adv}}$	0.35	0.22	0.25	0.38	0.25	85.6
ARMOR-PRM	**0.30**	**0.18**	**0.15**	**0.25**	**0.59**	**90.2**

Table 10: Full verification of the observed-neighborhood diagnostic on held-out counterfactual pairs. Pass% is computed per pair: each pair $(\mu_i, \delta_i, \eta_i)$ is checked individually against $\mu_i > L\delta_i + \eta_i$. The table reports aggregate summaries for interpretability; the RHS summary is computed from the same per-pair measurements as Pass%, rather than by multiplying the rounded table entries. μ_{p5} is the 5th-percentile empirical margin across pairs and is reported as a distributional summary only; it is not used as input to the per-pair check. Adding $\mathcal{L}_{\text{lip}}$ reduces L from 0.35 to 0.30, and the full three-regularizer combination raises Pass% from 85.6 to 90.2. Pairwise PRM's negative μ_{p5} indicates that its pairwise scoring formulation does not enforce a positive absolute margin between correct and adversarially disguised flawed steps.

Note that δ is an empirical constant set to the 95th-percentile backbone embedding shift measured over training-observed P1–P4 variants. The bound therefore holds for any $\hat{\tau}^- \in \mathcal{N}_\epsilon(\tau^-)$ whose embedding shift falls within this empirically observed range, and this range defines the scope of the diagnostic.

Step 2 (Adversarial invariance bridge). Let $\eta \geq 0$ denote the measured residual $|R_\theta(\tau^-) - R_\theta(\tilde{\tau}^-)|$ at the evaluation checkpoint. The proof conditions on this measured value rather than assuming convergence of $\mathcal{L}_{\text{adv}}$. By definition of η:

$$R_\theta(\tau^-) \leq R_\theta(\tilde{\tau}^-) + \eta \tag{11}$$

Step 3 (Counterfactual margin). By condition (i), $\mu = R_\theta(\tau^+) - R_\theta(\tilde{\tau}^-)$, i.e.:

$$R_\theta(\tilde{\tau}^-) \leq R_\theta(\tau^+) - \mu \tag{12}$$

Combining Steps 1–3:

$$\begin{aligned} R_\theta(\hat{\tau}^-) &\leq R_\theta(\tau^-) + L\delta \\ &\leq R_\theta(\tilde{\tau}^-) + \eta + L\delta \\ &\leq R_\theta(\tau^+) - \mu + \eta + L\delta. \end{aligned} \tag{13}$$

Rearranging gives $R_\theta(\hat{\tau}^-) - R_\theta(\tau^+) \leq L\delta + \eta - \mu$. In the idealized case $\eta = 0$ (perfect adversarial invariance), this reduces to $L\delta - \mu$. □

Proof of Corollary 1. Follows directly from Proposition 1. If $\mu_i > L\delta_i + \eta_i$, then $L\delta_i + \eta_i - \mu_i < 0$, which implies $R_\theta(\hat{\tau}^-) - R_\theta(\tau^+) < 0$, i.e., $R_\theta(\hat{\tau}^-) < R_\theta(\tau^+)$. Since $\hat{\tau}^-$ was an arbitrary element of $\mathcal{N}_\epsilon(\tau^-)$ with embedding shift $\leq \delta_i$, the inequality holds for all such elements. □

Proof of Corollary 2. Recall the exploitability definition from Eq. (1):

$$\Delta_\theta(\hat{\tau}^-, \tau^+) = R_\theta(\hat{\tau}^-) - R_\theta(\tau^+). \tag{14}$$

$$\mathcal{E}(R_\theta) = \mathbb{E}_{(\tau^+, \tau^-) \sim \pi} \left[\max_{\hat{\tau}^- \in \mathcal{N}_\epsilon(\tau^-)} \left[\Delta_\theta(\hat{\tau}^-, \tau^+) \right]_+ \right]. \tag{15}$$

By Proposition 1, for any $\hat{\tau}^- \in \mathcal{N}_\epsilon(\tau^-)$ with $\|f_\phi(\hat{\tau}^-) - f_\phi(\tau^-)\|_2 \leq \delta_i$:

$$R_\theta(\hat{\tau}^-) - R_\theta(\tau^+) \leq L\delta_i + \eta_i - \mu_i. \tag{16}$$

Therefore, for every such $\hat{\tau}^-$:

$$\max\big(0,\ R_\theta(\hat{\tau}^-) - R_\theta(\tau^+)\big) \leq \big[L\delta_i + \eta_i - \mu_i\big]_+. \tag{17}$$

Split	Type	Shortcut Targeted	Representative Transformation
Train	A1: Authoritative preamble	Theorem citation / authoritative tone	Prepend "By [theorem], it follows..." while leaving the erroneous derivation unchanged.
Train	A2: Structured elaboration	Numbered proof format	Rewrite one flawed step as "(i)... (ii)... (iii)...", preserving the same wrong conclusion.
Train	A3: Symbolic decoration	High LaTeX or symbol density	Add unused variables, irrelevant identities, or theorem names without changing the faulty computation.
Train	A4: Fabricated intermediate check	Mid-derivation verification	Insert a true but irrelevant sanity check between two parts of the flawed step.
Train	A5: Pseudo self-correction	"Wait / actually" reflection	Add a fake rechecking passage that ultimately confirms the original wrong result.
Train	A6: Confident rephrasing	Certainty markers	Replace neutral phrasing with "clearly", "obviously", or "it follows immediately", without fixing the error.
Test control	C2: Stylistic paraphrase	Benign surface variation	Rephrase wording and sentence structure while preserving all math claims and labels.
Test attack (GPT-4o + human)	B1: False attribution	Source attribution	Claim the wrong conclusion follows from a theorem, prior step, or problem condition.
Test attack (GPT-4o + human)	B2: Spurious verification	Backward checking	Append a circular or erroneous verification that appears to confirm the wrong answer.
Test attack (GPT-4o + human)	B3: Consistency theater	Unit/parity/scale checks	Insert an irrelevant or flawed dimensional, parity, or scale check that declares consistency.

Table 11: Math-domain perturbation templates. Training attacks A1–A6 are generated by DeepSeek-V4-Flash. Test attacks B1–B3 are constructed from a mixed-source pool (GPT-4o-generated and manually constructed variants in equal proportion), and the benign C2 control uses GPT-4o paraphrases only. Both the template families and the generation sources are disjoint from the training pipeline.

Type	Shortcut Targeted	Representative Transformation
A1: Fabricated location reasoning	Plausible household causal chains	Explain why the wrong location/object is likely correct using household commonsense.
A2: Fabricated context continuity	References to previous observations	Claim a previous search failed, making the wrong next action appear adaptive.
A3: Thought–action alignment spoof	Semantic match between thought and action	Rewrite the thought to perfectly justify the unchanged wrong action.
A4: Pseudo self-correction	Reconsideration language	Mention a plausible alternative, dismiss it with a wrong reason, and reaffirm the wrong action.
A5: Confident goal binding	Confidence plus task-goal reference	Frame the wrong action as the most direct way to achieve the task goal.

Table 12: ALFWorld-domain perturbation templates used for agent-trajectory data construction. Each template rewrites only the `Thought` field and leaves the `Action` line unchanged, so the original action-level correctness label is preserved.

Since the right-hand side is independent of $\hat{\tau}^-$, taking the supremum over $\hat{\tau}^- \in \mathcal{N}_\epsilon(\tau^-)$ preserves the inequality:

$$\ell_{\text{adv}}(\tau^+, \tau^-) = \max_{\hat{\tau}^- \in \mathcal{N}_\epsilon(\tau^-)} \big[R_\theta(\hat{\tau}^-) - R_\theta(\tau^+)\big]_+. \tag{18}$$

$$\ell_{\text{adv}}(\tau^+, \tau^-) \le [L\delta_i + \eta_i - \mu_i]_+. \tag{19}$$

Taking the expectation over $(\tau^+, \tau^-) \sim \pi$:

$$\mathcal{E}(R_\theta) \le \mathbb{E}_i\Big[\big[L\delta_i + \eta_i - \mu_i\big]_+\Big]. \tag{20}$$

When $\mu_i > L\delta_i + \eta_i$, the per-pair term is zero, so $\mathcal{E}(R_\theta) = 0$, in line with Corollary 1. □

G Prompt Templates

This appendix summarizes the rewriting templates used to construct adversarial variants. We report template intent and representative transformations rather than full API prompts, since the prompts are procedural instructions and not model contributions. All templates enforce the same hard constraint: the rewritten step must preserve the original correctness label. For incorrect steps, the numerical value, formula, action, or inference that makes the step wrong must stay unchanged. For correct steps, we only use benign paraphrases.

Cross-generator test construction. Training variants (A1–A6) are generated by DeepSeek-V4-Flash. Test attacks B1–B3 are drawn from a *mixed-source* pool. For each incorrect step, two variants are produced per attack type: one by GPT-4o (independently prompted with the same template intent), and one manually constructed from only the category label and the label-preservation constraint. The two sources are sampled in equal proportion, and neither overlaps with the training generator. Variants failing the semantic-distance gate ($d_{\text{sem}} > \epsilon$) or the label-verification check were discarded.

The data-generation scripts implement these templates with three additional safeguards. First, adversarial variants are generated only for incorrect steps in the attack settings, whereas benign paraphrases may be generated for all steps. Second, the training and test perturbation families are intentionally disjoint in the math setting (A1–A6 vs. B1–B3/C2), and test attacks B1–B3 are further drawn from a mixed-source pool (GPT-4o and manually constructed) that is disjoint from the DeepSeek-V4-Flash training generator. Third, for ALFWorld the adversarial rewriting keeps the executable action string fixed and modifies only the natural-language thought, so the rewriter cannot accidentally correct the trajectory.

H Additional ALFWorld Diagnostics

Round-1 cross-PRM scoring matrix. Table 13 reports the full 4×5 cross-scoring matrix for the ALFWorld Round-1 experiment (§6.1). Five GRPO lanes (ORM, Vanilla PRM, ARMOR-PRM, Pairwise PRM, and CF-Adv PRM) are each trained for 200 steps. We then collect 50 rollouts per lane at step 200 and score them with all four PRMs.

Scoring PRM	ORM policy	Vanilla policy	ARMOR-PRM policy	Pairwise PRM policy	CF-Adv policy
Vanilla PRM	0.337	*0.638*	0.675	0.329	0.625
ARMOR-PRM	0.354	0.675	*0.822*	0.410	0.744
Pairwise PRM	0.369	0.565	0.544	*0.825*	0.504
CF-Adv PRM	0.261	0.638	0.871	0.321	*0.791*

Table 13: Full ALFWorld Round-1 cross-PRM scoring matrix. Italicized entries are self-scoring values. Since every rollout has $R_{\text{true}} = 0$, any positive proxy score is reward inflation on a failed trajectory. Pairwise PRM has the lowest off-diagonal mean (0.502), and ARMOR-PRM has the highest (0.585).

Scoring PRM	Proxy Mean	Δgap	Phantom ASR	Verbose ASR	ρ
Vanilla PRM	0.330	-0.237	0.075	0.042	0.027
ARMOR-PRM	0.305	-0.261	0.000	0.000	0.122
Pairwise PRM	0.401	**-0.166**	0.375	0.125	0.025
CF-Adv PRM	0.414	**-0.152**	0.025	0.000	-0.125

Table 14: Synthetic ALFWorld diagnostic on 120 trajectories that contain phantom, verbose, normal, and failure steps. Δgap is proxy mean minus true success rate, and ρ is the Spearman correlation between proxy score and true success. Pairwise PRM has the smallest absolute reward gap among the non-CF-Adv PRMs, and CF-Adv has the lowest Verbose ASR.

Rows are the scoring PRMs and columns are the policy lane that generated the failed rollouts. All entries are average proxy scores on failed trajectories (SR=0 universally), so lower scores indicate better rejection of cross-policy reward hacking.

The cross-scoring matrix measures whether a PRM assigns high reward to other policies' failed trajectories. By this criterion, Pairwise PRM has the lowest off-diagonal mean and the smallest cross-policy inflation. The synthetic diagnostic (Table 14) isolates hand-designed phantom and verbose patterns. CF-Adv is strongest on Verbose ASR, while Pairwise PRM remains the strongest on cross-policy inflation in real rollouts.

I Artifact and Reproducibility Statement

Artifacts, Citation, and Licenses. The main external artifacts are cited where they are introduced: PRM800K (Lightman et al., 2024), ProcessBench (Zheng et al., 2025), Qwen2.5-Math (Yang et al., 2024), GSM8K (Cobbe et al., 2021), ALFWorld (Shridhar et al., 2021), LoRA (Hu et al., 2022), and GRPO/DeepSeekMath (Shao et al., 2024). We use PRM800K under its MIT license (OpenAI, 2023), ProcessBench under Apache License 2.0, Qwen2.5-Math checkpoints under Apache License 2.0 (Alibaba Cloud, 2024), and ALFWorld under its MIT license (Shridhar et al., 2021). Our usage is research evaluation and model training/fine-tuning, consistent with the intended research and benchmark use of these artifacts. We do not redistribute the original datasets, model checkpoints, or ALFWorld environment files. Adversarial variants and hard negatives derived from these artifacts are intended for PRM robustness research and remain subject to the licenses and access terms of their source artifacts.

Artifact Coverage and Statistics. PRM800K provides step-level math reasoning supervision and is used for PRM training. ProcessBench is an English mathematical-reasoning benchmark with four splits (`gsm8k`, `math`, `olympiadbench`, and `omnimath`); we exclude `math` from the reported averages due to its 78.8% problem-level overlap with PRM800K. The retained ProcessBench evaluation uses 3 splits, and the full benchmark contains 3,064 problems and 23,203 steps. The adversarial-data pipeline generates up to six variants per step and keeps accepted variants only after the semantic-distance gate and automated label verification; a 300-pair manual audit gives 96.7% label preservation. For ALFWorld, we use the unseen-134 household-navigation split and an augmented Round-2 PRM training set of 18,434 steps (14,134 original + 4,300 hard-negative steps). The robustness-bound diagnostic samples 2,000 pairs per checkpoint for each measured quantity (L, δ, η, and μ).

Implementation and Compute. Training and evaluation use PyTorch and Hugging Face Transformers with the Qwen2.5-Math-1.5B and Qwen2.5-Math-1.5B-Instruct backbones, OpenAI-compatible API clients for perturbation generation, and NumPy/Matplotlib for aggregation and plotting. The semantic gate uses the frozen `all-mpnet-base-v2` encoder. Total compute is approximately 250–300 GPU-hours on A100-class GPUs for the full paper experiments. The additional robustness-bound measurement in Appendix E uses 6 checkpoints in parallel on 8×L20 infrastructure and takes approximately 9 minutes.

Hyperparameter values are reported in Appendix A. Results in Tables 2 and 3 are means $\pm$ standard deviations across three random seeds, while Best-of-N values are point estimates averaged over five subsampling runs.