Reward Under Attack: Evaluating the Sensitivity of Process Reward Models

Udbhav Bamba [1 *] **Heng Yang** [2 *] **Rishabh Tiwari** [2 *] **Michael W. Mahoney** [2 3 4] **Kurt Keutzer** [2] **Amir Gholami** [2 3]

Abstract

Reward models (RMs) supervise large language models (LLMs) by aligning outputs with human preferences. Recently, process reward models (PRMs) have emerged to provide finer-grained evaluations by scoring intermediate reasoning steps. Despite their growing importance, the robustness and biases of PRMs under textual perturbations remain largely unexplored. In this work, we introduce **PRMProbe**, a framework to systematically audit PRMs with respect to their sensitivity to input modifications. We augment ProcessBench, a publicly released benchmark of question–answer trajectories, with eight types of controlled perturbations, and release this extended benchmark as **PRM-BiasBench**. These perturbations include semantics-preserving (e.g., rephrasing) and semantics-altering modifications (e.g., injecting hallucinations). Our analysis reveals that, unlike RMs which have known biases such as length preference, PRMs are generally robust to superficial edits like rephrasing and verbosity changes but exhibit varying levels of vulnerability to semantics-altering attacks. Surprisingly, a substantial fraction of semantically corrupted trajectories still receive unchanged or high rewards, suggesting that PRMs can overlook logical errors when trajectories maintain a fluent structure. These findings expose critical limitations in current PRM designs and underscore the need for more semantically grounded evaluation strategies. The code and dataset is available at https://github.com/SqueezeAILab/reward-under-attack

[*]Equal contribution [1]Transmute AI [2]UC Berkeley [3]ICSI [4]LBNL. Correspondence to: Rishabh Tiwari <rishabhtiwari@berkeley.edu>, Amir Gholami <amirgh@berkeley.edu>.

The second AI for MATH Workshop at the 42nd International Conference on Machine Learning, Vancouver, Canada.

1. Introduction

Large language models (LLMs) have become the backbone of modern AI systems, powering applications ranging from conversational assistants and search engines to code generation and scientific reasoning (Achiam et al., 2023; Team et al., 2023; Bubeck et al., 2023; Yang et al., 2025). Trained on massive corpora of internet-scale text, these models demonstrate strong capabilities in a wide array of tasks, including summarization, translation, question answering, and multi-turn dialogue (Chowdhery et al., 2022; Zhang et al., 2022). More recently, LLMs have been increasingly applied to domains that demand structured step-by-step reasoning, such as solving math word problems, writing correct code, and answering science exam questions, where correctness is not just about producing a fluent output, but about following a valid logical process (Cobbe et al., 2021; Hendrycks et al., 2021; Chen et al., 2021).

To align LLMs with desired behaviors especially in tasks requiring reliable multi-step reasoning *reward models* (RMs) have become a central component in both supervised fine-tuning and RL pipelines. Traditionally, these models assign scalar rewards to full model outputs and are trained to reflect human preferences, playing a key role in reinforcement learning from human feedback (RLHF) (Christiano et al., 2017; Ouyang et al., 2022; Bai et al., 2022). However, outcome-level rewards are coarse: they provide no insight into whether a model reasoned correctly along the way or merely arrived at the correct answer by chance.

This has led to the development of *process reward models* (PRMs), which assign rewards at the level of individual reasoning steps. By offering step-by-step feedback, PRMs promise better control during training, more interpretable scoring, and better incentives for structured reasoning. They have recently been deployed in settings such as scaling test-time compute (Chen et al., 2023), step-level fine-tuning (Lightman et al., 2023), and trajectory filtering in high-quality instruction datasets (Zhou et al., 2023).

Despite their promise, the robustness of PRMs has not been systematically evaluated. Prior studies have identified various biases in traditional outcome-level RMs—including sensitivity to length, sycophancy, prefix biases and reward hacking (Singhal et al., 2024; Shen et al., 2023; Denison et al., 2024; Kumar et al., 2025; Lee et al., 2023; Fu et al.,

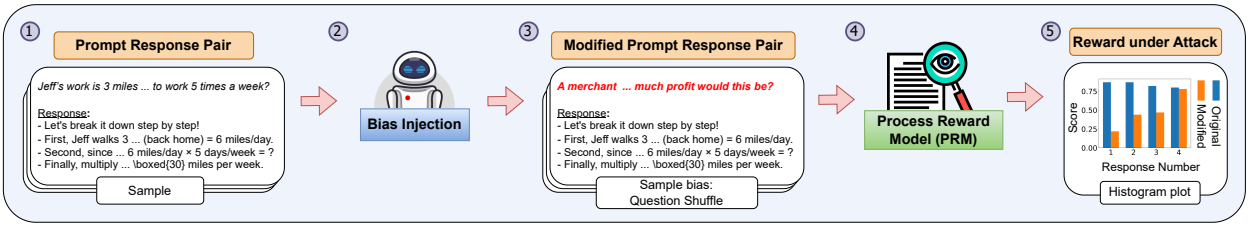


Figure 1. **Overview of Reward Under Attack**: A prompt-response pair (Step 1) undergoes bias injection (Step 2) such as question shuffling. The modified sample (Step 3) is evaluated by a PRM (Step 4), and scores are compared against the original (Step 5).

2023; Webson & Pavlick, 2022)—but analogous investigations for PRMs are largely absent. In particular, it remains unclear whether PRMs truly assign rewards based on semantic correctness or whether they are vulnerable to superficial textual variations. This is especially concerning given their growing role in downstream applications such as reward-guided decoding, chain-of-thought distillation, and filtering training data for alignment (Chen et al., 2023; Lightman et al., 2023; Zhou et al., 2023). As PRMs increasingly influence both training and inference behaviors of LLMs, a deeper understanding of their biases, failure modes, and generalization properties is critical.

In this work, we conduct the first systematic audit of PRMs with respect to their sensitivity to input variations. Building on ProcessBench (Zheng et al., 2024), a publicly available benchmark of 3.4k verified question-answer reasoning trajectories, we apply eight categories of controlled perturbations. These include both semantic-preserving modifications—such as rephrasing and semantic-altering changes such as addition of hallucinated facts.

Evaluating two representative PRMs, Skywork-o1-Open-PRM-7B and Qwen2.5-Math-PRM-7B, we identify distinct bias profiles and critical robustness failures. While both models show relative stability under certain surface-level edits, they frequently fail to penalize meaning-altering perturbations, particularly when the changes involve numeric or factual content. These findings suggest that **current PRMs may rely more on fluency and structural regularity than on deep semantic understanding**.

To enable systematic investigation of these failure modes and support reproducible research on PRM evaluation, we develop and release a comprehensive diagnostic suite. Our contributions are summarized as follows:

- We introduce **PRMProbe**, a diagnostic workflow for evaluating PRM robustness under both semantic-preserving and semantic-altering perturbations (§4).
- We show that different PRMs exhibit inconsistent behaviors when subjected to input alterations, often relying more on semantic flow rather than logical correctness. (§5, §6).
- We release an annotated dataset **PRM-BiasBench** based on ProcessBench, with 8 perturbation types, along with evaluation code and semantic verification prompts to facilitate future research (§4.2).

2. Background

2.1. Reward Models and RLHF

Reward models (RMs) are trained to approximate human preferences over language model outputs and play a central role in reinforcement learning from human feedback (RLHF) (Christiano et al., 2017; Ouyang et al., 2022; Bai et al., 2022). Given a prompt x and two completions (y_1, y_2), human annotators (or heuristics) indicate which completion is preferred. The reward model $r_\theta(x, y)$ is trained to assign higher scalar scores to the preferred outputs, allowing these comparisons to be modeled probabilistically.

A common approach is to use the Bradley–Terry model, where the probability that y_1 is preferred over y_2 given x is:

$$p(y_1 \succ y_2 \mid x) = \frac{\exp(r_\theta(x, y_1))}{\exp(r_\theta(x, y_1)) + \exp(r_\theta(x, y_2))}.$$

This induces a preference likelihood objective over a dataset of chosen vs. rejected examples $(x, y_{\text{chosen}}, y_{\text{rejected}})$:

$$\mathcal{L}(\theta; \mathcal{D}) = \mathbb{E}_{(x, y_c, y_r) \sim \mathcal{D}} \Big[\log \sigma \big(r_\theta(x, y_c) - r_\theta(x, y_r) \big) \Big].$$

This loss encourages the model to assign higher scores to the preferred completions. Once trained, RMs are used to optimize LLMs via reinforcement learning (e.g., PPO), or to rank and filter generated outputs in supervised pipelines. However, these traditional reward models operate on full outputs and provide no signal on intermediate reasoning quality, motivating the development of step-wise reward modeling, which we discuss next.

2.2. Step-by-Step Reasoning Models

To address the limitations of single-shot generation, recent work has emphasized reasoning models that produce struc-

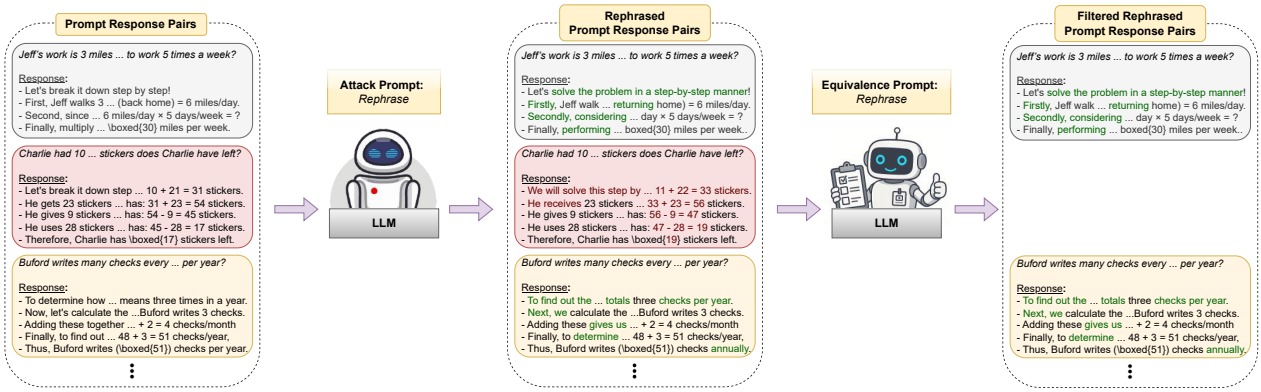


Figure 2. **Step-by-step framework for creating the *PRM-BiasBench* dataset.** Original prompt-response pairs are perturbed using an attack prompt via an LLM. An equivalence checker then filters out semantically altered outputs, retaining only meaning-preserving transformations. The figure illustrates this process using a rephrasing attack as an example, incorrectly altered responses are highlighted in red, while semantically equivalent responses passing the filter are shown in green.

tured, multi-step outputs. In settings such as math problem solving (Cobbe et al., 2021), competition-level mathematics (Hendrycks et al., 2021), and code generation (Chen et al., 2021), models are prompted or trained to generate solutions one logical step at a time. This approach helps improve both interpretability and accuracy by exposing intermediate computations.

Each step in the reasoning trajectory contributes to the final outcome. As such, evaluating only the end result may miss crucial errors in the reasoning. This motivates the need for more fine-grained supervision and evaluation strategies that can assess the quality of individual steps.

2.3. Process Reward Models

Process reward models (PRMs) extend traditional reward modeling by assigning scores to individual steps within a reasoning trajectory. Formally, given a question q and a partial trajectory $(s_1, \ldots, s_i)$, a PRM computes a reward $r_i = \text{PRM}(q, s_{\leq i})$ for each step s_i. This allows for localized feedback throughout the reasoning process.

The nature of the reward assigned by a PRM depends heavily on its data annotation strategy and training objective (Zhang et al., 2025). For instance, a PRM may be trained to regress toward the probability of reaching a correct final answer using Monte Carlo estimates. Alternatively, a PRM can be trained to predict whether each individual reasoning step is factually correct.

PRMs have been applied in multiple settings, including reward-guided speculative decoding (Chen et al., 2023), step-level finetuning (Lightman et al., 2023), and trajectory filtering for alignment (Zhou et al., 2023). They are also used to select the most promising reasoning path during inference or training.

The step-level nature of PRMs enables more targeted supervision, but also raises new challenges: because they operate at a finer granularity, their sensitivity to surface-level cues (e.g., rephrasing) and their semantic fidelity are important open questions that motivate our empirical analysis.

3. Related Work

Robustness of Reward Models. Although reward models (RM) have significantly advanced the alignment of language models with human preferences, they remain susceptible to various forms of misalignment. One prominent issue is *reward hacking*, where the policy generates outputs that receive high scores from the reward model without actually reflecting the intended behavior (Ibarz et al., 2018; Denison et al., 2024). This misalignment can degrade downstream performance (Bai et al., 2022) and widen the gap between what is rewarded and what is semantically helpful (Stiennon et al., 2020).

Reward hacking often arises through superficial heuristics. For example, numerous studies have documented *length bias*, where longer outputs receive inflated rewards irrespective of content quality (Singhal et al., 2024; Dubois et al., 2023; Liu et al., 2024). This is particularly problematic in RLHF pipelines, where optimization techniques such as PPO can amplify spurious correlations between structure and reward.

Improving Process Reward Models. Recent efforts have explored ways to train more robust and informative process reward models (PRMs) that better reflect human evaluations at the step level. For instance, Zhang et al. (2025) analyzed and discussed tips to create datasets for training strong PRMs. Similarly, Zheng et al. (2024) construct a dataset with human-annotated reasoning trajectories, where

annotators identify incorrect steps within multi-step solutions, and train a PRM to detect such first erraneous step in the trajectory.

More recently, Xu et al. (2025) investigate the limitations of PRMs and find that they often latch onto shallow consistency cues rather than learning true causal reasoning structures. While their work highlights an important misalignment between human causal understanding and PRM scoring, it does not comprehensively analyze PRM sensitivity to a wide range of semantic-preserving and semantic-altering edits. In contrast, our work systematically audits how state-of-the-art PRMs respond to diverse controlled perturbations, and releases a benchmark for future evaluation.

4. PRMProbe

In this section, We introduce **PRMProbe**, our systematic framework for auditing the robustness and bias of process reward models. Figure 1 shows the schematic representation of the workflow. This section describes the models and datasets used, the construction of our new PRM BiasBench benchmark, our equivalence checking and manual review procedures, and the evaluation metrics used in our analyses.

4.1. Setup

Models. We evaluate two process reward models: *Qwen2.5-Math-PRM-7B* and *Skywork-o1-Open-PRM-Qwen-2.5-7B*. The Skywork PRM is trained to estimate the probability of success at each intermediate step. The Qwen PRM, in contrast, is trained to locate the first incorrect step in a trajectory and assign it a low score, reflecting a break in reasoning correctness.

Reasoning Trajectories. We use verified step-by-step reasoning trajectories from ProcessBench as the foundation for our robustness evaluation. ProcessBench combines high-quality trajectories generated by a diverse set of LLMs, including Qwen2, Qwen2.5-Math, Llama-3 and Llama-3.1 variants, spanning multiple scales from 1.5B to 72B parameters, and covers diverse math benchmarks such as GSM8K, MATH500, Omni-MATH, and OlympiadBench. This diverse pool provides a representative sample of reasoning chains across different problem types and model families.

4.2. PRM BiasBench

To systematically evaluate the robustness and failure modes of process reward models (PRMs), we introduce **PRM BiasBench**, a benchmark suite comprising semantically verified perturbations applied to high-quality reasoning trajectories. PRM BiasBench extends ProcessBench by injecting controlled biases and adversarial edits specifically designed to probe PRM sensitivity to both superficial and meaning-altering changes. Figure 2 and Algorithm 1 shows the overall steps to generate the modified trajectories.

Bias Injection. Starting from verified trajectories in ProcessBench, we apply eight controlled perturbation templates that target distinct reasoning properties. These include semantics-preserving modifications—such as rephrasing, verbosity adjustments, and within-step reordering—as well as semantics-altering attacks such as question shuffling, numerical value changes, hallucinated facts, and question removal. Each perturbation is generated automatically via structured prompting and subsequently filtered through an equivalence validation pipeline.

Bias Verification and Manual Review. To ensure that each modified trajectory faithfully reflects its intended modification, we use a strong LLM (GPT-4o) to verify that the injected bias is present and works as intended. For high-impact cases with large reward deviations, we conduct manual inspection to resolve any ambiguous outcomes. This hybrid validation ensures that reward differences are attributable to the target bias rather than spurious generation artifacts.

Benchmark Composition. PRM BiasBench consists of thousands of verified perturbation pairs, each annotated with the perturbation type, and original versus modified PRM scores. This enables fine-grained auditing of PRM robustness and bias profiles under diverse conditions. We release the complete suite, along with prompt templates and validation code, to facilitate reproducibility and future extensions.

4.3. Robustness Evaluation

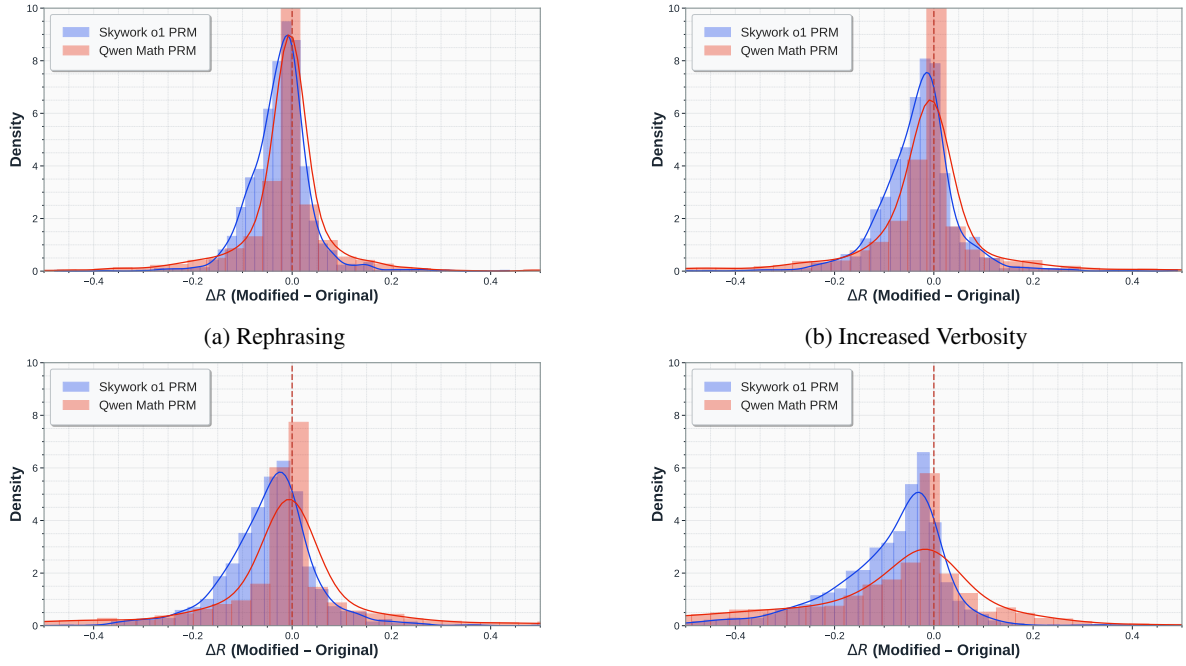
For each perturbation pair $(T, \tilde{T})$, we compute the reward difference $\Delta R = R(\tilde{T}) - R(T)$ where $R(T)$ denotes the scalar reward assigned to the entire trajectory, extracted from step-wise scores according to the specific PRM, depending on their training strategy. In particular, we define:

$$R_{\text{Skywork}}(T) = r_n,$$
$$R_{\text{Qwen}}(T) = \min_i r_i.$$

Here, r_i denotes the reward at step i and r_n denotes the reward at the final step. To evaluate PRM robustness we used the mean and standard deviation of ΔR for each perturbation type, along with distribution plots, to highlight model-specific failure modes and outliers.

5. Semantics-Preserving Modifications

Process reward models (PRMs) should evaluate the logical integrity and correctness of each reasoning step independent



(a) Rephrasing

(b) Increased Verbosity

(c) Decreased Verbosity

(d) Reordering Within Steps

Figure 3. Distribution of reward changes (ΔR) for four semantics-preserving perturbations: (a) Rephrasing, (b) Increased Verbosity, (c) Decreased Verbosity, and (d) Reordering within steps. For robust PRMs, ΔR should be centered near zero with minimal deviation, indicating invariance to superficial linguistic edits. Skywork shows broader and slightly left-skewed distributions, revealing mild sensitivity to style variation compared to Qwen.

Algorithm 1 Reward Under Attack

Input:
  Trajectory T
  PRM function `ComputePRMReward`
  Transformation function `ApplyAttack`
  Equivalence checker `CheckEquivalence`
Output:
  Reward difference ΔR or `not_equivalent`
$R_{\text{orig}} \leftarrow$ `ComputePRMReward`(T) // Score original T
$\widetilde{T} \leftarrow$ `ApplyAttack`(T) // Apply transformation
$R_{\text{mod}} \leftarrow$ `ComputePRMReward`$(\widetilde{T})$ // Score modified
// Skip semantically different pairs
if `CheckEquivalence`$(T, \widetilde{T}) =$ **False then**
  return `not_equivalent`
end if
$\Delta R \leftarrow R_{\text{mod}} - R_{\text{orig}}$
return ΔR

of superficial surface-level variations in phrasing. Although prior works (Singhal et al., 2024; Dubois et al., 2023; Liu et al., 2024) have shown that reward models are biased towards trivial linguistic changes such as length bias, rephrasing, and verbosity this raises a critical question: Do PRMs inherit similar sensitivities?

In this section, we apply a suite of *semantics-preserving* modifications that are designed to change the linguistic form of a reasoning trajectory while leaving its underlying logical structure and final answer intact. By subjecting PRMs to these controlled perturbations, we aim to probe whether they genuinely capture semantic accuracy or simply react to surface-level text artifacts. Below, we detail the four distinct categories of semantics-preserving perturbations applied in our evaluations, accompanied by illustrative examples:

5.1. Rephrasing

Example 1: Rephrasing

Original:
Step R: "Compute the sum of the first three terms: $2 + 4 + 6 = 12$."
Rephrased:
Step R: "Add the initial three numbers together to get $2 + 4 + 6 = 12$."

This attack rephrases each step in the reasoning trajectory using a different but equivalent language (Example 1: Rephrasing). Figure 3(a) shows that both distributions have low deviations and are centered near zero, indicating that many trajectories are scored similarly before and after the rephras-

ing. Thus, it is safe to say that current PRMs are mostly robust to small paraphrasing changes. However, Skywork PRM exhibits a broader distribution with a heavier left tail. This suggests that Skywork is more sensitive to paraphrasing than Qwen.

5.2. Increased Verbosity

Example 2: Increased Verbosity

Original:
Step V: "Divide both sides by 4 to isolate x: $8x/4 = 12/4$, so $x = 3$."
Verbose:
Step V: "Now, in order to solve for the variable x, we take the equation $8x = 12$ and divide both sides of this equality by 4. This yields $8x/4 = 12/4$, which simplifies directly to $x = 3$."

This attack adds redundant but semantically equivalent language to each step in the reasoning trajectory (Example 2: Increased Verbosity). Figure 3(b) shows that both PRMs' reward-change distributions remain tightly centered near zero, indicating minimal impact from mere lengthening of the text. Nevertheless, Skywork PRM has a slightly wider spread and a more pronounced left tail, suggesting it is somewhat more sensitive to verbosity than Qwen.

5.3. Decreased Verbosity

Example 3: Decreased Verbosity

Original:
Step C: "The height of the beanstalk after n days can be expressed as: 4×2^n."
Concise:
Step C: "After n days, the beanstalk's height is 4×2^n."

This modification reduces verbosity by making each reasoning step more concise while preserving all logical content (Example 3: Decreased Verbosity). Figure 3(c) presents the distribution of reward changes (ΔR= Modified – Original) for both PRMs. Qwen2.5-Math-PRM-7B remains tightly centered near zero, indicating minimal sensitivity to brevity, whereas Skywork-o1-Open-PRM-7B exhibits a broader spread and a more negative mean shift, suggesting it is somewhat more sensitive to concise phrasing as well.

5.4. Reordering Within Steps (Conclude Before Reasoning)

Example 4: Reordering

Original:
Step O: "Josh has 2 apples. He got two more, so Josh now has $2 + 2 = 4$ apples."
Reordered:
Step O: "Josh now has $2 + 2 = 4$ apples, since he had 2 apples and got two more."

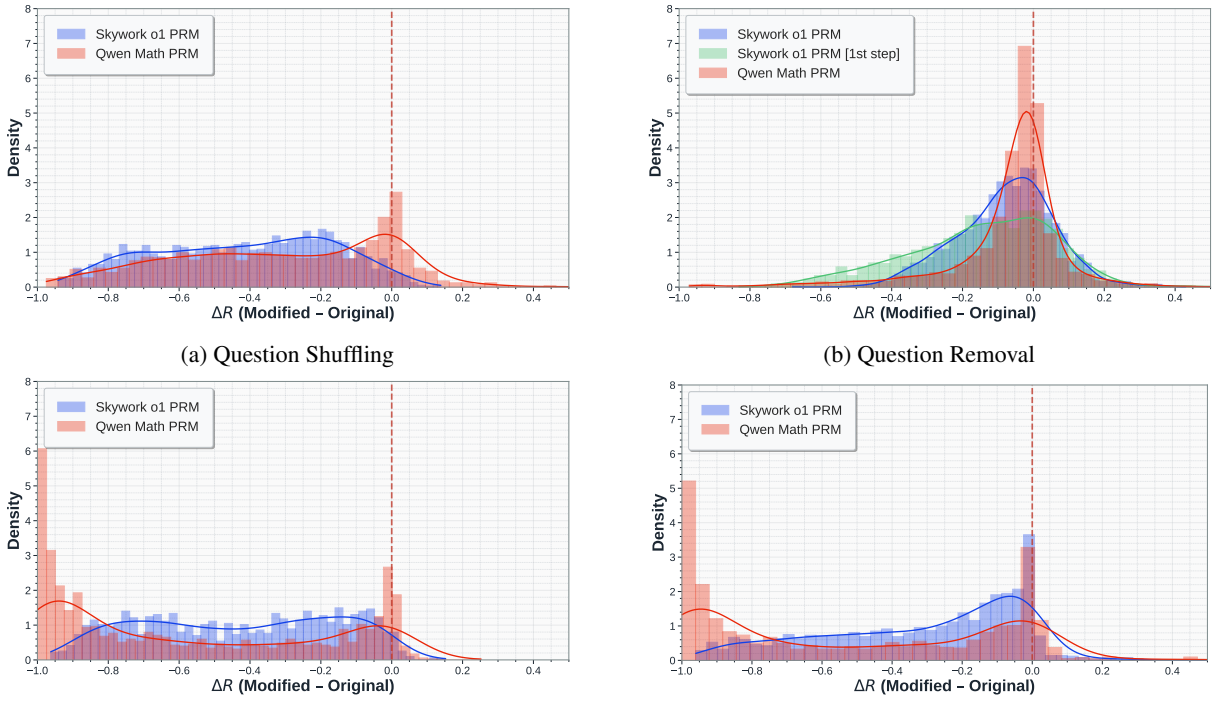
This modification places the conclusion of a step before its supporting reasoning while preserving all logical content (Example 4: Reordering). Figure 3(d) shows the distribution of reward changes for both PRMs. Qwen2.5-Math-PRM-7B displays a heavier left tail and greater variance, indicating more frequent reward drops under atypical ordering. Skywork-o1-Open-PRM-7B remains closer to zero with only a slight left skew and a moderately wider spread. These results indicate that both PRMs rely on structural conventions rather than purely semantic understanding.

Insight 1 💡:
Across all four semantics-preserving edits: rephrasing, verbosity changes and reordering within steps both PRMs remain largely invariant (most $|\Delta R| \leq 0.02$), with Qwen showing tighter, more symmetric distributions and Skywork exhibiting slightly heavier tails and mild left skews. This demonstrates strong but not perfect semantic robustness and highlights the benefit of augmenting PRM training with diverse, perturbed trajectories to eliminate remaining stylistic biases.

6. Semantics-Altering Modifications

Process reward models (PRMs) should not only be invariant to trivial rewordings but also sensitive to genuine breaks in logical correctness. While the previous section (Section 5) probed surface-level robustness, here we introduce *semantics-altering* modifications that deliberately disrupt key aspects of a reasoning trajectory and should cause a marked decrease in per-step rewards.

In this section, we apply four classes of semantics-altering edits: question shuffling, numerical perturbation, hallucination injection, and information removal. Below, we describe each attack category with illustrative examples:



(a) Question Shuffling

(b) Question Removal

(c) Numerical Value Change

(d) Reasoning Hallucination

Figure 4. Distribution of reward changes (ΔR) for four semantics-altering perturbations: (a) Question Shuffling, (b) Question Removal, (c) Numerical Value Change, and (d) Reasoning Hallucination. Qwen shows weaker penalization for question mismatches and hallucinations, while Skywork more consistently reduces rewards for corrupted reasoning.

6.1. Question Shuffling

Example 5: Question Shuffling

Original:
"Jeff's work is 3 miles away. He walks ... if he has to work 5 times a week?"
Step 1: First, Jeff walks 3 ... = 6 miles/day.
...
Question Shuffling:
"The red rope was four times ... length of the red rope in centimeters?"
Step 1: First, Jeff walks 3 ... = 6 miles/day.
...

This modification assigns each trajectory a different prompt from ProcessBench ensuring none remain paired with their original prompt so that the reasoning steps no longer match the new question. Figure 4(a) shows that Skywork-o1-Open-PRM-7B exhibits a large mean drop with a peak near -0.20, indicating strong penalization for mismatched question–reasoning pairs. Whereas, Qwen2.5-Math-PRM-7B peak near 0 with a heavier left tail, reflecting more variable sensitivity.

6.2. Question Removal

Example 6: Question Removal

Original:
"Jeff's work is 3 miles away. He walks ... if he has to work 5 times a week?"
Step 1: First, Jeff walks 3 ... = 6 miles/day.
...
Question Removal:
(Prompt removed.)
Step 1: First, Jeff walks 3 ... = 6 miles/day.
...

This modification removes the question entirely, providing only the reasoning steps to the PRM. It probes how much each model's reward depends on having the original prompt as context. Figure 4(b) shows Qwen2.5-Math-PRM-7B has a peak around 0 with long tail indicating only minor reliance on the prompt. Whereas, Skywork-o1-Open-PRM-7B's first-step scores drop sharply (heavy negative tail) but its cumulative trajectory score largely recovers, suggesting the prompt is most critical for early step evaluation.

6.3. Question Numerical Value Change

Example 7: Question Numerical Value Change

Original:
"Jeff's work is 3 miles away. He walks ... if he has to work 5 times a week?"
Step 1: First, Jeff walks 3 ... = 6 miles/day.
...
Numerical Value Change:
"Jeff's work is 8 miles away. He walks ... if he has to work 7 times a week?"
Step 1: First, Jeff walks 3 ... = 6 miles/day.
...

This modification alters key numeric values in the prompt while the corresponding reasoning trajectory is untouched. Figure 4(c) shows that both PRMs strongly penalize these semantic breaks: Qwen2.5-Math-PRM-7B's distribution is tightly peaked near –1.0, indicating near-complete reward collapse whenever numbers are wrong, whereas Skywork-o1-Open-PRM-7B exhibits a broader negative distribution centered around roughly –0.5, suggesting it sometimes still grants partial credit. We also observe a secondary bump around 0 in roughly 5–10% of cases, the modified prompts change numeric values that were irrelevant to solution validity (e.g., years or extraneous statistics), so the PRMs correctly maintain their original scores.

6.4. Reasoning Hallucination

Example 8: Reasoning Hallucination

Original:
If a and b are integers ... is divided by 20?
Step 1: To find the remainder ... divided by 20.
...
Step N: The remainder when ... by 20 is $\boxed{17}$.
Reasoning Hallucination:
If a and b are integers ... is divided by 20?
Step 1: To find the remainder ... divided by 20. Assuming that a and b are both greater than 20, we proceed with the calculation accordingly.
...
Step N: The remainder when ... by 20 is $\boxed{0}$.

This modification injects a spurious assumption or false fact into a reasoning step, breaking the logical flow while preserving surface syntax. Figure 4(d) shows that Qwen2.5-Math-PRM-7B exhibits a sharp peak at -1.0, indicating near-certain collapse of reward whenever a hallucination occurs, whereas Skywork-o1-Open-PRM-7B's distribution is centered around 0 with a long negative tail reflecting that it treats the hallucination as valid and gradually recovers its score over subsequent steps.

Insight 2 💡:
Across these four semantics-altering attacks, we see that both PRMs are capable of detecting genuine breaks in logical correctness, but with markedly different behaviors. Skywork-o1-Open-PRM-7B applies consistent early penalties but often recovering toward the original reward over later steps whereas Qwen2.5-Math-PRM-7B acts as a near-binary filter, collapsing its score almost deterministically for any numeric or factual corruption but failing to flag mismatches in structure or missing context.

7. Discussion

Our systematic audit reveals that while current process reward models (PRMs) are generally robust to superficial linguistic edits such as rephrasing, verbosity changes, and within-step reordering, this invariance is not perfect: we observe mild but measurable reward fluctuations even for semantics-preserving modifications. This suggests that PRMs can still latch onto shallow stylistic cues, highlighting the benefit of augmenting PRM training with diverse, perturbed reasoning trajectories to improve true linguistic invariance.

In contrast, PRMs exhibit clear limitations in detecting more serious semantic corruptions. Semantics-altering perturbations such as question shuffling, numerical inconsistencies, or hallucinated reasoning steps often fail to elicit sufficiently lower rewards, especially in the Qwen PRM. This indicates that fluency and step-like structure can overly influence PRM scoring, allowing reward hacking even in step-wise evaluations.

These findings raise important concerns for downstream applications that rely on PRMs, such as test-time scaling, RL with PRM rewards, and alignment filtering. If PRMs do not reliably capture genuine logical correctness, they risk reinforcing flawed reasoning patterns during training and inference. Our study thus underscores the need for more semantically grounded PRMs, robust to adversarial and out-of-distribution inputs, and trained with diverse perturbations to mitigate reliance on shallow cues.

8. Conclusion

In this work, we presented the first systematic robustness audit of process reward models under a suite of carefully controlled semantic-preserving and semantic-altering perturbations. Our experiments show that existing PRMs demonstrate strong invariance to trivial surface changes but remain vulnerable to deeper semantic failures, highlighting a critical misalignment between fluency and reasoning correctness.

To foster progress, we release the PRMProbe framework and the PRM-BiasBench dataset as open tools to test and improve the robustness of PRM. We hope that this work inspires new training objectives, evaluation metrics, and diagnostic strategies that close the gap between apparent fluency and true stepwise reasoning quality.

9. Acknowledgements

We acknowledge gracious support from the FuriosaAI, Intel, Apple, NVIDIA, Macronix, and Mozilla. We also appreciate the support from Microsoft through their Accelerating Foundation Model Research. Furthermore, we appreciate support from Google Cloud, the Google TRC team, and specifically Jonathan Caton, and Prof. David Patterson. Prof. Keutzer's lab is sponsored by the Intel corporation, UC Berkeley oneAPI Center of Excellence, Intel VLAB team, as well as funding through BDD and BAIR. We appreciate great feedback and support from Ellick Chan, Saurabh Tangri, Andres Rodriguez, and Kittur Ganesh. Michael W. Mahoney would also like to acknowledge a J. P. Morgan Chase Faculty Research Award as well as the DOE, NSF, and ONR. This work was supported by the Director, Office of Science, Office of Advanced Scientific Computing Research, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. Our conclusions do not necessarily reflect the position or the policy of our sponsors, and no official endorsement should be inferred.

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Bai, Y., Kadavath, S., Kundu, S., et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Bubeck, S., Chadrasekaran, V., Eldan, R., Gehrke, J., Horvitz, E., Kamar, E., Lee, P., Lee, Y. T., Li, Y., Lundberg, S., et al. Sparks of artificial general intelligence: Early experiments with gpt-4, 2023.

Chen, M., Tworek, J., Jun, H., et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Chen, S., Singh, A., Lee, K., Berrio, A., et al. Scaling test-time compute for longer reasoning in language models. *Hugging Face Cookbook*, 2023. https://huggingface.co/learn/cookbook/en/search_and_learn.

Chowdhery, A., Narang, S., Devlin, J., et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., and Amodei, D. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, 2017.

Cobbe, K., Kosaraju, V., Bavarian, M., et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Denison, C., MacDiarmid, M., Barez, F., Duvenaud, D., Kravec, S., Marks, S., Schiefer, N., Soklaski, R., Tamkin, A., Kaplan, J., et al. Sycophancy to subterfuge: Investigating reward-tampering in large language models. *arXiv preprint arXiv:2406.10162*, 2024.

Dubois, Y., Zhang, W., Wang, Z., et al. Alpacaeval: An automatic evaluator of instruction-following llms. *arXiv preprint arXiv:2306.05685*, 2023.

Fu, Y., Hou, R., Wu, S., Li, L., and Zhao, W. X. How robust is gpt-4? a comprehensive study of zero-shot and few-shot prompt variability. *arXiv preprint arXiv:2305.18225*, 2023.

Hendrycks, D., Burns, C., Kadavath, S., et al. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Ibarz, J., Leike, J., Pohlen, T., Irving, G., Legg, S., and Amodei, D. Reward learning from human preferences and demonstrations in atari. In *NeurIPS*, 2018.

Kumar, A., He, Y., Markosyan, A. H., Chern, B., and Arrieta-Ibarra, I. Detecting prefix bias in llm-based reward models. *arXiv preprint arXiv:2505.13487*, 2025.

Lee, K., Casper, J. U., Irving, G., and Leike, J. Adversarial reward hacking: Towards measuring robustness of rlhf. *arXiv preprint arXiv:2310.06771*, 2023.

Lightman, S., Wei, J., Chi, E., et al. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Liu, A., Zou, A., et al. Exploring the failure modes of direct preference optimization. *arXiv preprint arXiv:2402.06786*, 2024.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, N., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155*, 2022.

Shen, W., Zheng, R., Zhan, W., Zhao, J., Dou, S., Gui, T., Zhang, Q., and Huang, X. Loose lips sink ships: Mitigating length bias in reinforcement learning from human feedback. *arXiv preprint arXiv:2310.05199*, 2023.

Singhal, K., Zhang, W., Lee, K., et al. The length preference bias in reward models. *arXiv preprint arXiv:2402.03620*, 2024.

Stiennon, N., Ouyang, L., Wu, J., et al. Learning to summarize with human feedback. In *NeurIPS*, 2020.

Team, G., Anil, R., Borgeaud, S., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., Millican, K., et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Webson, A. and Pavlick, E. Do prompt-based models really understand the meaning of their prompts? *arXiv preprint arXiv:2109.01247*, 2022.

Xu, Y., Dong, H., Wang, L., Xiong, C., and Li, J. Reward models identify consistency, not causality. *arXiv preprint arXiv:2502.14619*, 2025.

Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., Zheng, C., Liu, D., Zhou, F., Huang, F., Hu, F., Ge, H., Wei, H., Lin, H., Tang, J., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Zhou, J., Lin, J., Dang, K., Bao, K., Yang, K., Yu, L., Deng, L., Li, M., Xue, M., Li, M., Zhang, P., Wang, P., Zhu, Q., Men, R., Gao, R., Liu, S., Luo, S., Li, T., Tang, T., Yin, W., Ren, X., Wang, X., Zhang, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Zhang, Y., Wan, Y., Liu, Y., Wang, Z., Cui, Z., Zhang, Z., Zhou, Z., and Qiu, Z. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zhang, S., Roller, S., Goyal, N., et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

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

Zheng, C., Zhang, Z., Zhang, B., Lin, R., Lu, K., Yu, B., Liu, D., Zhou, J., and Lin, J. Processbench: Identifying process errors in mathematical reasoning. *arXiv preprint arXiv:2412.06559*, 2024.

Zhou, Y., Arora, S., Yu, T., Goel, K., Hou, L., Mishra, S., Kaplan, J., and Zhang, L. Lima: Less is more for alignment. *arXiv preprint arXiv:2305.11206*, 2023.