ConsistRM: Improving Generative Reward Models via Consistency-Aware Self-Training

Yu Liang[*], **Liangxin Liu**[*], **Longzheng Wang, Yan Wang, Yueyang Zhang**
Long Xia, Zhiyuan Sun, Daiting Shi[†]
Baidu Inc., Beijing, China
{liangyu05, liuliangxin, wanglongzheng, wangyan78}@baidu.com
{zhangyueyang, xialong01, sunzhiyuan01, shidaiting01}@baidu.com

Abstract

Generative reward models (GRMs) have emerged as a promising approach for aligning Large Language Models (LLMs) with human preferences by offering greater representational capacity and flexibility than traditional scalar reward models. However, GRMs face two major challenges: reliance on costly human-annotated data restricts scalability, and self-training approaches often suffer from instability and vulnerability to reward hacking. To address these issues, we propose **ConsistRM**, a self-training framework that enables effective and stable GRM training without human annotations. ConsistRM incorporates the **Consistency-Aware Answer Reward**, which produces reliable pseudo-labels with temporal consistency, thereby providing more stable model optimization. Moreover, the **Consistency-Aware Critique Reward** is introduced to assess semantic consistency across multiple critiques and allocates fine-grained and differentiated rewards. Experiments on five benchmark datasets across four base models demonstrate that ConsistRM outperforms vanilla Reinforcement Fine-Tuning (RFT) by an average of 1.5%. Further analysis shows that ConsistRM enhances output consistency and mitigates position bias caused by input order, highlighting the effectiveness of consistency-aware rewards in improving GRMs. Our implementation is available at https://github.com/yuliangCarmelo/ConsistRM.

1 Introduction

Large language models (LLMs) have seen growing popularity in both academia and industry due to their exceptional capability in tackling complex tasks. A core technique for aligning LLMs' outputs with human expectations is Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022; Bai et al., 2022; Dong et al., 2024; Lambert, 2025; Lai et al., 2025). The reward model in RLHF supplies supervision signals for training, and its effectiveness is widely acknowledged as a primary factor limiting the maximum achievable alignment performance(Rafailov et al., 2023; Azar et al., 2024). Recently, an increasing number of studies have focused on generative reward models (GRMs) (Mahan et al., 2024; Ye et al., 2025; Liu et al., 2025d). In contrast to traditional models that merely output scalar scores, GRMs offer greater representational capacity, enhanced generalization, and better adaptability to complex feedback.

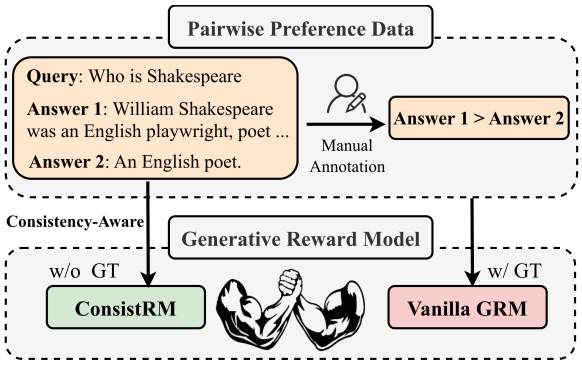

Figure 1: An overview of ConsistRM in comparison with vanilla GMRs, emphasizing its self-training framework based on consistency-aware learning to enhance model pairwise preference performance.

While GRMs have demonstrated effectiveness, their training typically requires large amounts of high-quality human-annotated label data, making the process costly and hard to scale. For instance, methods like (Chen et al., 2025) depend on human-labeled reward signals for reinforcement learning. Similarly, approaches such as SynPref-40M (Liu et al., 2025a) leverage LLMs to curate data automatically, though they still rely on initial human annotations for guidance. To reduce manual annotation, recent studies have explored unsupervised or self-training paradigms for GRMs, such as generat-

[*]Equal contribution.
[†]Corresponding authors.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 39449–39466

ing training signals from LLMs' internal entropy or confidence (Zuo et al., 2025; Shafayat et al., 2025; Wang et al., 2025c). However, these methods remain fragile because the reward signals are tightly correlated with the policy model, which easily leads to reward hacking and early overfitting to noisy pseudo-labels(Laidlaw et al., 2024; Zhang et al., 2025c). Overall, despite notable advances, achieving robust and stable GRM training without any human feedback remains a major challenge.

In this work, we propose *ConsistRM*, a novel self-training framework that leverages internally derived consistency signals without human-annotated labels. Our goal is to enhance the pairwise preference ability of GRM through two key modules: (1) the Consistency-Aware Answer Reward, which considers temporal consistency (including both current model states and historical results) to provide reliable training signals; (2) the Consistency-Aware Critique Reward, which measures semantic consistency across diverse critiques and offers fine-grained rewards. The combination of these two complementary internal signals results in a more stable and reliable self-training framework.

We evaluate the effectiveness and generalization of ConsistRM across five benchmark datasets using four base models. Experimental results show that ConsistRM achieves an average performance gain of 1.5% compared to vanilla Reinforcement Fine-Tuning (RFT). Through detailed analysis, we further demonstrate that ConsistRM enhances output consistency and mitigates position bias induced by input order.

Our contributions are summarized as follows:

- We propose ConsistRM, a framework that enables self-training for GRMs through consistency-aware mechanisms, eliminating the need for human feedback.
- Within this framework, we design a method to construct pseudo-labels by leveraging consistency between current and historical model outputs, thereby providing GRMs with more reliable learning signals.
- Empowered by the consistency-aware reward from ConsistRM, GRMs generate outputs with improved conciseness and accuracy, while significantly reducing position bias.

2 Related work

Generative Reward Model Recent experimental studies have demonstrated that GRMs represent a key new paradigm in reward modeling, primarily due to their enhanced interpretability and generalization capabilities (Liu et al., 2025a; Whitehouse et al., 2025; Saha et al., 2025). For example, DeepSeek-GRM (Liu et al., 2025d) introduces a pointwise generative approach that produces detailed critiques and self-derived evaluation rules through reinforcement learning (RL), enabling more flexible and task-agnostic scoring compared to conventional scalar RMs. This generative paradigm has inspired subsequent research. For instance, RM-R1 (Chen et al., 2025) incorporates chain-of-thought reasoning via a two-stage process: distilling high-quality reasoning traces followed by RL with verifiable rewards (Guo et al., 2025a). This approach yields sample-specific evaluation rationales, thereby enhancing both interpretability and empirical performance. Further extending the integration of reasoning, Reward Reasoning Models (Guo et al., 2025b) introduce a "reason-before-judgment" methodology without requiring annotated traces and dynamically adapt computation based on input complexity. However, a common limitation of these existing generative and reasoning-based GRMs is their dependence on supervised signals and multi-stage pipelines to obtain high-quality labels or reasoning traces, which substantially limits their further development.

Label-Free Self-Training for LLMs To address the challenge of acquiring high-quality labels, researchers have shifted their focus toward self-training paradigms that leverage LLMs to generate supervisory signals, thereby reducing reliance on human annotation. Existing methods can be divided into two main categories: The first focuses on stabilizing outputs by encouraging low-entropy predictions or leveraging internal confidence signals to improve training efficiency (Li et al., 2025b; Zhang et al., 2025a; Li et al., 2025a). The second constructs supervision signals from the consistency of multiple output paths, providing more reliable guidance for reward modeling, which is closer to our approach. A representative example of the second category is Test-Time Reinforcement Learning (TTRL), which uses majority voting over candidate answers as pseudo-labels for RL updates (Zuo et al., 2025; Liu et al., 2025b). Extensions such as EVOL-RL (Zhou et al., 2025) and collaborative reward frameworks (Zhang et al., 2025d) have been proposed to enhance this paradigm. However, recent studies (Wang et al., 2025a) suggest

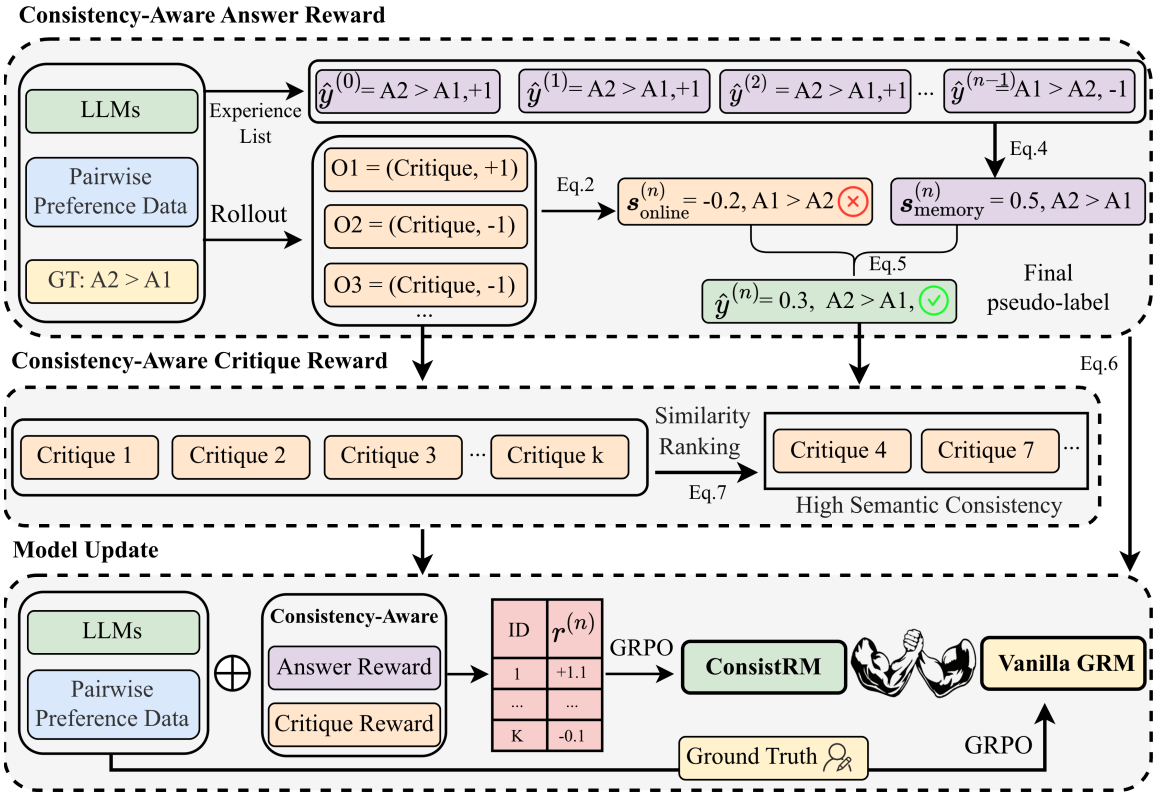

Figure 2: Overall Framework of ConsistRM. The framework aims to produce an output $o = (c, y)$, which comprises a textual critique c and a binary preference label y. Here, $y = -1$ indicates that a_1 is preferred over a_2 ($a_1 \succ a_2$)., and $y = 1$ indicates the opposite preference ($a_2 \succ a_1$). Consistency-Aware Answer Reward module integrates online-state preference and memory-driven preference to generate pseudo-labels for training. The Consistency-Aware Critique Reward module provides fine-grained rewards by measuring the similarity of multiple criteria. By combining these two rewards, ConsistRM performs self-training to progressively enhance the GRMs' ability.

that pure majority voting may suffer from spurious consensus when candidate outputs are highly correlated, offering limited guidance for fine-grained reasoning. Other approaches exploit internal model states, using confidence scores for advantage estimation or as direct feedback in preference optimization (Zhao et al., 2025; van Niekerk et al., 2025). Nevertheless, methods relying solely on within-round consensus or coarse-grained feedback are highly noise-sensitive, compromising the reliability of the resulting supervisory signals.

3 Method

We propose ConsistRM, a stable and reliable self-training framework that explicitly GRM both answer preference consistency and critique consistency. As illustrated in Figure 2, ConsistRM consists of two core components: (1) Consistency-Aware Answer Reward(CAAR), which considers online-state and memory-driven preference consistencies to produce reliable pseudo-labels; (2) Consistency-Aware Critique Reward(CACR), which encourages semantic consistency among multiple generated critiques, thereby improving the stability of the analysis process. ConsistRM leverages consistency between answers and critiques to provide fine-grained reward, thereby enhancing its pairwise preferences ability.

3.1 Pairwise Preference Formulation

We model the problem of pairwise preference comparison as a conditional generative process. Given an input $x = (q, a_1, a_2)$ consisting of a query q and two candidate responses (a_1, a_2), GRM π generates a structured output $o = (c, y)$, where c is a textual critique and y is a binary preference label, as detailed below:

$$(c, y) \sim \pi_\theta^{(n)}(c, y \mid q, a_1, a_2), \tag{1}$$

where (n) denotes the GRM training at the n-th iteration. The preference label $y \in \{-1, 1\}$ indicates the direction of preference: $y = -1$ corresponds to $a_1 \succ a_2$, while $y = 1$ corresponds to $a_2 \succ a_1$.

3.2 Consistency-Aware Answer Reward

To alleviate reward hacking(Gao et al., 2023; Laidlaw et al., 2024) and establish more reliable pseudo labels, we propose a Consistency-Aware Answer Reward (CAAR) mechanism. This mechanism constructs pseudo-labels by building the relation between two types of preferences: the Online-State Preference (derived from the current training iteration) and the Memory-Driven Preference (captured from memory experience). Consequently, the pseudo-labels produced are more robust and reliable, providing stronger support for self-training.

Online-State Consistency Preference We define the online-state preference signal as a metric that quantifies preference consistency under the current state of GRMs. Given an input $x = (q, a_1, a_2)$, the model performs K rollouts during training, generating a set of outputs: $O = [o_1, o_2, \ldots, o_k]$, and $o_j = (c_j, y_j)$ corresponds to the output of model from the j-th rollout. To quantify the online-state preference consistency for the input x at n-th iteration, we aggregate the preference prediction y_j obtained from all rollouts in O and the online-state consistency preference $s^{(n)}_{\text{online}}$ is defined as:

$$\boldsymbol{s}^{(n)}_{\text{online}} = \frac{1}{K}\sum_{j=1}^{K} y_j \tag{2}$$

where n means the n-th training iteration.

Memory-Driven Consistency Preference. Although online-state preference consistency reflects the preferences expressed by the current GRM π, the estimate can be unreliable in the early stages of training, which may cause the GRM to converge toward suboptimal local optima(Zhang et al., 2025c). To enhance the robustness and reliability of pseudo-labels, we propose a memory-driven consistency preference. Specifically, for each input sample x, we maintain a dynamically updated experience list $\mathcal{E}$ that stores the pseudo-labels collected across training iterations:

$$\mathcal{E}^{(n-1)} = [\hat{y}^{(0)}, \hat{y}^{(1)}, \ldots, \hat{y}^{(n-1)}] \tag{3}$$

where $\hat{y}^{(n-1)}$ denotes the pseudo-label constructed for input x at the $(n-1)$-th training iteration. Notably, the initialization pseudo-label $\hat{y}^{(0)}$ is generated by the initial GRM $\pi^{(0)}_\theta$. To quantify the memory-driven consistency preference for input x, we aggregate all pseudo-labels accumulated in $\mathcal{E}$ by computing their average. This simple averaging strategy serves as a stable semantic anchor over historical predictions, rather than emphasizing any individual pseudo-label. At the n-th training iteration, the memory-driven preference consistency $s^{(n)}_{\text{memory}}$ is defined as:

$$\boldsymbol{s}^{(n)}_{\text{memory}} = \frac{1}{n-1}\sum_{i=0}^{n-1} \hat{y}^{(i)} \tag{4}$$

Pseudo-Label with Temporal Consistency We derive the final pseudo-label $\hat{y}$ for the current iteration by jointly leveraging consistency signals at multiple levels, which leads to more reliable pseudo-labels that are better aligned with the training dynamics. Specifically, for each input sample x at the n-th training iteration, the pseudo-label is constructed as follows:

$$\hat{y}^{(n)} = sgn(s^{(n)}_{\text{online}} + s^{(n)}_{\text{memory}}) \tag{5}$$

where $\text{sgn}(\cdot)$ denotes the *sign function*. This ternary labeling strategy explicitly assigns $\hat{y} = 0$ in the presence of inconsistent preference signals, which prevent low-confidence supervision from dominating the optimization. Although early-stage pseudo-labels may be noisy, their influence naturally diminishes as more iterations are accumulated, making the overall estimate increasingly robust over time, as further validated by our analysis of CAAR in Appendix B.1.

Finally, for each input x, we define the Consistency-Aware Answer Reward $r^{(a)}_j$, which measures the agreement between the constructed pseudo-label $\hat{y}^{(n)}$ and the preference prediction $y^{(n)}_j$ generated in the j-th rollout at n-th training iteration. The reward is computed as:

$$\begin{aligned} r^{(a,n)}_j = &\mathbb{I}\left[\hat{y}^{(n)} = y^{(n)}_j\right] \\ &- \mathbb{I}\left[\hat{y}^{(n)} \neq y^{(n)}_j \ \wedge \ \hat{y}^{(n)} \neq 0\right] \end{aligned} \tag{6}$$

where $\mathbb{I}[\cdot]$ denotes the *indicator function*, and $r^{(a,n)}_j \in \{-1, 0, 1\}$.

3.3 Consistency-Aware Critique Reward

Rather than relying solely on outcome-level supervision, we introduce a Consistency-Aware Critique Reward (CACR) in ConsistRM to deliver richer reward signals and stabilize the outputs of GRMs. We posit that high semantic similarity among multiple generated critiques implies that their corresponding responses lie within a similar high-reward semantic region, which can be leveraged to guide model

optimization more effectively. When the GRM π_θ produces critiques with strong semantic consistency for the same input x, the resulting reward signal becomes more stable and less sensitive to sampling variability. Based on this observation, we use consistency among critiques as a proxy for reward signal reliability, and that reinforcing such consistency improves model alignment with stable and representative evaluations.

Concretely, following (Zhou et al., 2025), each critique $c_j^{(n)}$ is encoded into a dense vector using the Qwen3-4B-Embedding (Zhang et al., 2025b) and computes the cosine similarity matrix $S^{(n)}$. Based on this matrix, we rank the critiques $c_j^{(n)}$ according to their semantic consistency, and select the top-p critiques to receive the reward. The corresponding reward is defined as:

$$r_j^{(c,n)} = \begin{cases} 0.1, & \text{if } y_j^{(n)} = \hat{y}^{(n)} \text{ and } \mathrm{rank}(c_j^{(n)}) \leq p; \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

where $r_j^{(c,n)}$ represents the consistency-aware critique reward for input x.

3.4 Final Reward

We define the final reward for each generated response by integrating three components: a mandatory format constraint, a consistency-aware answer reward, and a consistency-aware critique reward. The format constraint is first applied to enforce structural validity. The correctness and stability reward components are computed only if this constraint is satisfied. Thus, for an input x at the j-th rollout of the n-th training iteration, the final reward is computed as follows:

$$r^{(n)} = \begin{cases} -5, & \text{if invalid;} \\ r_j^{(a,n)} + r_j^{(c,n)}, & \text{if } \hat{y}^{(n)} \neq 0; \\ 0, & \text{if } \hat{y}^{(n)} = 0. \end{cases} \tag{8}$$

Details of the ConsistRM algorithm are provided in the appendix C.

4 Experiments

4.1 Setup

Benchmark To thoroughly assess ConsistRM's performance on pairwise preference tasks, we utilize five reward model benchmarks that contain multilingual instructions and responses from diverse LLMs: (1) **RewardBench** (Lambert et al., 2025) includes 2,985 evaluation samples from 23 data sources, categorized into four main groups: chat, chat-hard, safety, and reasoning. (2) **Preference Proxy Evaluations (PPE)** (Frick et al., 2025): we focus on its Preference subset with Chatbot Arena human preference pairs, covering 20 LLMs across 121+ languages. (3) **RM-Bench** (Liu et al., 2025c): a benchmark designed to assess reward model robustness through sensitivity to content differences and style biases. (4) **RMB** (Zhou et al., 2024) is a comprehensive benchmark with 49 real-world task categories focused on helpfulness and harmlessness. It uses synthetically generated preference pairs and GPT-4 for pointwise ratings based on query-specific principles. Human verification ensures dataset quality. (5) **JudgeBench** (Tan et al., 2025): we report results for the subset generated by GPT-4o and use position-consistent accuracy, considering a sample correct only if the judge gives the right verdict in both response orders.

Implementation details We use Qwen3 as our testbed and fine-tune it for 4 epochs using the GRPO (Shao et al., 2024). The fine-tuning employs a global batch size of 64 and a learning rate of 1e-6, with a maximum generation length of 1024 tokens. GRPO involves a proximal policy optimization (PPO) phase; during this phase, we use a mini-batch size of 32 and perform 8 rollouts per update step. The generation temperature during training is set to 1.0. To stabilize training and prevent excessive policy drift, we apply KL regularization with a coefficient of 0.001. For inference, following (Whitehouse et al., 2025; Saha et al., 2025), we set the maximum generation length to 2048 tokens and the temperature to 0 (more detailed configuration can be found in Appendix A.1).

Baselines We perform the following experiments to evaluate our approach comprehensively:

- **Base**: We evaluate the Qwen3-4B (Yang et al., 2025) and Qwen3-8B models on benchmark datasets without any fine-tuning.
- **SFT**: We fine-tune the base Qwen3 models using supervised fine-tuning on the HelpSteer3 dataset (Wang et al., 2025b).
- **RFT**: we further fine-tune the Qwen3 models using the GRPO algorithm (Shao et al., 2024) for reinforcement fine-tuning (for more dataset detailed can be found in Appendix A.2).

ID	System	GT	RewardBench	PPE Pref	RM-Bench	RMB	JudgeBench	Overall	
								AVG	Δ (↑)
			Open Generative Reward Models						
1	EvalPlanner-Llama-8B	✓	83.0	54.3	68.1	-	30.2	-	-
2	J1-Llama-8B	✓	85.7	59.8	73.4	-	42.0	-	-
3	DeepSeek-GRM-27B	✓	88.5	67.2	-	70.3	-	-	-
			Implemented Existing Method						
4	Qwen3-4B		78.6	63.3	74.5	76.4	50.0	68.6	-
5	4 + SFT	✓	79.8	63.1	76.1	76.5	43.7	67.8	-0.8
6	4 + RFT	✓	80.0	64.7	76.6	76.7	51.1	69.8	+1.2
7	4 + TTRL	✗	**81.6**	61.8	76.3	75.6	52.3	69.5	+0.9
			Our Method						
8	4 + CAAR	✗	81.4	64.1	76.6	**77.5**	52.0	70.3	+1.7
9	8 + CACR (ConsistRM)	✗	80.3	**65.1**	**76.8**	76.5	**56.1**	71.0	**+2.4**
			Implemented Existing Method						
10	Qwen3-8B		81.6	63.8	75.8	78.8	54.3	70.9	-
11	10 + SFT	✓	82.7	65.0	77.1	76.9	51.7	70.7	-0.2
12	10 + RFT	✓	85.4	65.4	78.2	78.2	55.4	72.5	+1.6
13	10 + TTRL	✗	85.3	65.0	77.4	74.2	56.8	71.7	+0.8
			Our Method						
14	10 + CAAR	✗	84.9	64.8	77.3	78.1	56.0	72.2	+1.3
15	14 + CACR (ConsistRM)	✗	**85.6**	**67.7**	**78.3**	**79.1**	**56.9**	73.5	**+2.6**

Table 1: Overall performance of different GRMs on five benchmarks. ConsistRM achieved the best results without requiring any human-annotated training data, demonstrating our effectiveness of the self-training paradigm. **Bold** indicating the best performance and underline indicating the second-best performance.

- **TTRL**: This method uses majority voting over candidate answers as pseudo-labels for reinforcement learning updates (Zuo et al., 2025; Liu et al., 2025b).

We then compare our ConsistRM against several open-source GRMs of similar scale, including EvalPlanner-Llama-8B (Saha et al., 2025), J1-Llama-8B (Whitehouse et al., 2025), and DeepSeek-GRM-27B (Liu et al., 2025d)

4.2 Main Results

We focus our analysis on Qwen3-8B, which is representative of the overall trends observed across smaller 4B models. System (10) shows the original Qwen3-8B's performance on pairwise preference benchmarks. Systems (5) and (11) achieve only limited improvements through vanilla SFT. The core issue is that standard loss functions (e.g., cross-entropy) average over the entire dataset, smoothing out training signals and impeding targeted updates for hard samples. However, online reinforcement learning offers an effective solution to this challenge. Specifically, GRPO assigns differentiated rewards to multiple sampled outputs based on existing labels, thereby yielding stable performance improvements, as shown in Systems (6) and (12). In contrast, simple unsupervised learning results in marginal improvements, as shown in Systems (7) and (13), primarily due to the increasing bias in pseudo-labels during the later stages of training.

Additionally, Systems (14) shows that incrementally integrating the ConsistRM submodule enhances GRM performance, surpassing the contemporary vanilla RFT and TTRL method. This stems from stable and reliable learning labels derived from current and historical state decisions, enabling effective self-learning. Notably, ConsistRM significantly boosts Qwen3's pairwise preference performance compared to vanilla RFT and other advanced methods, as seen in System (15). The improvement is particularly pronounced on JudgeBench benchmarks, likely due to our ConsistRM produce more consistent responses.

4.3 Ablation Study

To better understand the contribution of each component in ConsistRM, we conduct ablation studies by removing individual modules. As shown in Table 2, removing either the CACR or the Online-State Consistency Preference module leads to an average performance drop of approximately 1.2 points, indicating that both components play essential roles in the overall framework. In contrast, ablating the Memory-Driven Consistency Preference results in the most significant degradation (-3.2),

System	RewardBench	PPE Pref	RM-Bench	RMB	JudgeBench	Avg	Δ (↑)
ConsistRM	85.6	67.7	78.3	79.1	56.9	73.5	-
w/o CACR	84.9	64.8	77.3	78.1	56.0	72.2	-1.3
w/o Online-State Consistency Preference	85.5	64.1	78.6	76.7	56.7	72.3	-1.2
w/o Memory-Driven Consistency Preference	84.3	63.1	75.4	74.2	54.8	70.4	-3.2

Table 2: Ablation study results of ConsistRM on five benchmarks.

suggesting that this component is critical for maintaining high-quality pseudo-labels. Its removal substantially reduces pseudo-label accuracy, indicating that accurate historical aggregation is crucial for stabilizing pseudo-label quality and improving performance.

5 Analysis

This section aims to address the following research questions through experiments: (1) Can ConsistRM adapt to different foundation models or other rewards of critique? (see §5.1) (2) What are the advantages of ConsistRM? (see §5.2) (3) How efficient is ConsistRM? (see §5.3) For evaluation, we primarily use Qwen3-4B as the base model and test our approach on benchmarks: RewardBench and JudgeBench, unless otherwise specified.

5.1 The Varieties of ConsistRM

Various Foundation Models Although ConsistRM shows impressive improvements on Qwen3-4B and 8B, its generalization to other foundation models remains to be validated. To address this, we apply ConsistRM to different models, including Qwen3-14B and LLaMA-3.1-8B. Under consistent experimental settings, we evaluate performance on two standard benchmarks. As shown in Table 3, ConsistRM consistently improves both models across these benchmarks (+1.3, +1.6). These results demonstrate that ConsistRM can effectively leverage the inherent consistency ability of LLMs (of varying sizes and architectures) for effective self-training without any human feedback.

Various Critique Rweard This section proposes two modifications to the Consistency-Aware Critique Reward Module (CACR) to examine its rationality. In contrast to ConsistRM, we introduce an additional reward term for samples in the bottom 50% of semantic consistency, denoted as CACR-Low. Second, we replace the critique quality estimation in CACR with a token-level confidence measure—DeepConfidence (Fu et al., 2025), which is implemented in two variants: (1) DeepConf-Tail: the average confidence of the final 128 tokens, based on the observation that reasoning quality often wanes at the end of long chains, where final steps are crucial for correctness. (2) DeepConf-Low: the average confidence of the 128 lowest-confidence tokens across the entire reasoning trajectory.

The effectiveness and necessity of CACR are supported in the Table 4. First, performance deteriorates when high rewards are assigned to low-similarity critiques, suggesting that higher-similarity critiques reflect stronger consensus and thus support more reliable preference judgments. Second, simply replacing CACR with DeepConfidence causes a significant performance drop (from 68.2 to 65.4). We attribute this phenomenon to reward hacking, in which optimization over token-level confidence biases the model toward overly confident but erroneous tokens, thereby leading to homogenized outputs and degraded overall performance. The above experiments show the rationality of our proposed consistency-aware critique reward.

5.2 The Consistency of ConsistRM

Robustness to Positional Bias In this section, we evaluate how effectively ConsistRM mitigates position bias in GRMs. Position bias is defined as the tendency of a model's preference between two responses to depend on their presentation order. In practice, for the same query, swapping the response order can lead to a reversal of the model's preference. To quantify this bias, we adopt the evaluation standard from Tan et al. (2025), where a sample is considered correctly evaluated only if the model produces consistent judgments under both orderings of the answer pair. We apply this standard to both RewardBench and JudgeBench, comparing base models, RFT models, and our ConsistRM.

As shown in Table 4, vanilla RFT yields only limited improvement in mitigating positional bias (+1.4). We hypothesize that for challenging samples, the model tends to learn shallow prompt–answer mappings rather than robust reasoning. In contrast, ConsistRM leverages its ability

Foundation Model	Method	RewardBench	PPE Pref	RM-Bench	RMB	JudgeBench	Overall	
							AVG	Δ (↑)
Qwen3-4B	Vanilla RFT	80.0	64.7	76.6	76.7	51.1	69.8	-
	ConsistRM	80.3	65.1	76.8	76.5	56.1	71.0	**+1.2**
Qwen3-8B	Vanilla RFT	85.4	65.4	78.2	78.2	55.4	72.5	-
	ConsistRM	85.6	67.7	79.2	79.2	55.9	73.5	**+1.0**
Qwen3-14B	Vanilla RFT	87.4	66.3	79.3	76.3	68.6	75.6	-
	ConsistRM	89.0	67.7	77.5	78.4	71.7	76.9	**+1.3**
Llama-3.1-8B	Vanilla RFT	74.8	55.3	63.2	62.7	30.2	57.2	
	ConsistRM	76.6	54.3	64.8	64.9	33.4	58.8	**+1.6**

Table 3: Results of ConsistRM with various foundation models. ConsistRM can be stably applied to LLMs of different sizes and architectures.

System	RewardBench	JudgeBench	Overall	
			AVG	Δ (↑)
Various Critique Rweard				
ConsistRM	80.3	56.1	68.2	-
w/ CACR-Low	81.2	50.6	65.8	-2.4
w/ DeepConf-Tail	80.4	50.3	65.4	-2.8
w/ DeepConf-Low	79.8	51.2	65.5	-2.7
Positional Bias				
Qwen3-4B	69.7	50.0	59.9	-
Vanilla RFT	71.4	51.1	61.3	+1.4
ConsistRM	74.2	56.1	65.2	+5.3

Table 4: Comparison results with different setups on GRMs. ConsistRM achieves the better Results.

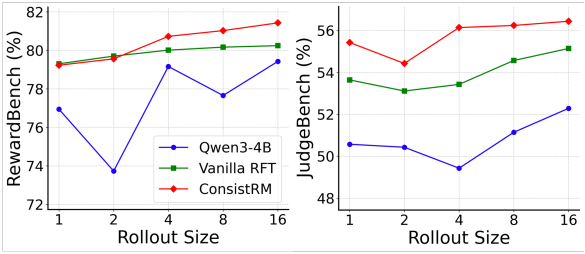

Figure 3: Comparison of the stability of different GRMs with different rollout sizes.

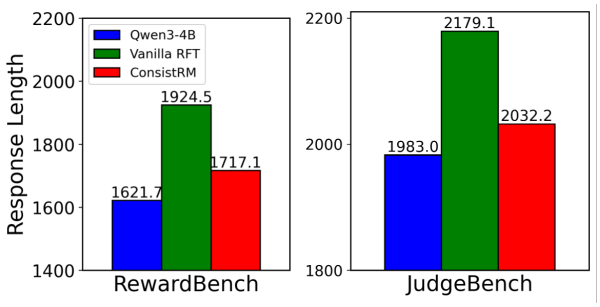

Figure 4: Changing trends of the average response length with different GRMs. ConsistRM achieves the balance between performance and efficiency.

to generate more suitable training labels, thereby promoting the training process and substantially mitigating positional bias. This leads to an improvement in the consistency score from +1.4 to +5.3, showing the robustness of ConsistRM.

Stability to Multiple Votes We further analyze the output stability of ConsistRM under high-temperature decoding, where stochasticity in generation can lead to increased variance in model outputs. To mitigate this issue, multi-round voting (Wang et al., 2023) improves answer consistency by aggregating outputs from multiple reasoning paths sampled at high temperature. We evaluate three models: the Qwen3-4B model, vanilla RFT, and our ConsistRM on RewardBench and JudgeBench, varying the number of rollout sizes (1, 2, 4, 8, 16). As shown in Figure 3, when applied to the base model, multi-round voting is observed to yield unstable gains on generative reward tasks due to output inconsistency, whereas vanilla RFT provides stable but limited improvements. In contrast, ConsistRM achieves consistent performance gains as the number of paths increases, eventually matching the performance of an 8B model while retaining the 4B parameter scale. This improvement can be attributed to ConsistRM's ability to effectively supervise the critique process, thereby enhancing response stability.

5.3 Less tokens, More power

The efficiency of ConsistRM ConsistRM is a pairwise preference reward model designed to improve both task performance and response efficiency—the latter measured by output length. To evaluate efficiency gains, we compare the average response lengths of Qwen3-4B, vanilla RFT, and ConsistRM on RewardBench and JudgeBench. As shown in Figure 4, RFT produces significantly longer responses than Qwen3-4B, likely because RFT relies solely on outcome reward without explicit length constraints. In contrast, ConsistRM

effectively controls output length while maintaining competitive performance. For example, on RewardBench, it reduces the average response length from 1,924.5 to 1,717.1. This reduction is primarily driven by consistency-aware critique rewards, which improve task performance while suppressing excessive verbosity in model outputs.

Conclusion

This paper proposes ConsistRM, a novel self-training strategy for GRMs that leverages the consistency awareness of LLMs to generate supervisory signals without relying on external human feedback. ConsistRM consists of two key modules: the Consistency-Aware Label Reward module and the Consistency-Aware Critique Reward module. The former enables self-training by explicitly modeling the model's internal preference consistency across multiple levels, while the latter derives more stable evaluation criteria to provide richer reward signals. Extensive experiments on multiple benchmarks and models of varying sizes demonstrate the effectiveness and robustness of the proposed ConsistRM. Furthermore, our analysis shows that the approach significantly improves the consistency and efficiency of answer generation while mitigating positional bias.

Limitations

Although ConsistRM demonstrates significant performance improvements, it has several limitations. First, semantic consistency is currently evaluated only at the overall critique level, which may overlook fine-grained alignment between text segments. Future work could address this by segmenting critiques into smaller semantic units and providing reward supervision at a more granular level. This direction aligns with research on process reward modeling in reinforcement learning, which aims to provide verifiable rewards and remains an open avenue for future exploration.

Ethics Statement

This work follows the ACL Ethics Policy. Our findings are based on publicly available datasets for reproducibility purposes. We acknowledge that LLMs may exhibit inherent societal biases and are prone to hallucinations. Therefore, if someone finds our work interesting and would like to use it in a specific environment, we strongly suggest the user conduct safety and bias evaluations to mitigate potential risks.

References

Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. 2024. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, pages 4447–4455. PMLR.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, and 1 others. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

Xiusi Chen, Gaotang Li, Ziqi Wang, Bowen Jin, Cheng Qian, Yu Wang, Hongru Wang, Yu Zhang, Denghui Zhang, Tong Zhang, and 1 others. 2025. Rm-r1: Reward modeling as reasoning. *arXiv preprint arXiv:2505.02387*.

Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. 2024. Rlhf workflow: From reward modeling to online rlhf. *arXiv preprint arXiv:2405.07863*.

Evan Frick, Tianle Li, Connor Chen, Wei-Lin Chiang, Anastasios Nikolas Angelopoulos, Jiantao Jiao, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. 2025. How to Evaluate Reward Models for RLHF. In *The Thirteenth International Conference on Learning Representations*.

Yichao Fu, Xuewei Wang, Yuandong Tian, and Jiawei Zhao. 2025. Deep think with confidence. *arXiv preprint arXiv:2508.15260*.

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

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025a. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Jiaxin Guo, Zewen Chi, Li Dong, Qingxiu Dong, Xun Wu, Shaohan Huang, and Furu Wei. 2025b. Reward reasoning model. *arXiv preprint arXiv:2505.14674*.

Hanyu Lai, Xiao Liu, Junjie Gao, Jiale Cheng, Zehan Qi, Yifan Xu, Shuntian Yao, Dan Zhang, Jinhua Du, Zhenyu Hou, and 1 others. 2025. A survey of post-training scaling in large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2771–2791.

Cassidy Laidlaw, Shivam Singhal, and Anca Dragan. 2024. Correlated proxies: A new definition and improved mitigation for reward hacking. *arXiv preprint arXiv:2403.03185*.

Nathan Lambert. 2025. Reinforcement learning from human feedback. *arXiv preprint arXiv:2504.12501*.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. 2025. Rewardbench: Evaluating Reward Models for Language Modeling. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 1755–1797, Albuquerque, New Mexico. Association for Computational Linguistics.

Tianjian Li, Yiming Zhang, Ping Yu, Swarnadeep Saha, Daniel Khashabi, Jason Weston, Jack Lanchantin, and Tianlu Wang. 2025a. Jointly reinforcing diversity and quality in language model generations. *arXiv preprint arXiv:2509.02534*.

Yi-Chen Li, Tian Xu, Yang Yu, Xuqin Zhang, Xiong-Hui Chen, Zhongxiang Ling, Ningjing Chao, Lei Yuan, and Zhi-Hua Zhou. 2025b. Generalist reward models: Found inside large language models. *arXiv preprint arXiv:2506.23235*.

Chris Yuhao Liu, Liang Zeng, Yuzhen Xiao, Jujie He, Jiacai Liu, Chaojie Wang, Rui Yan, Wei Shen, Fuxiang Zhang, Jiacheng Xu, Yang Liu, and Yahui Zhou. 2025a. Skywork-reward-v2: Scaling preference data curation via human-ai synergy. *arXiv preprint arXiv:2507.01352*.

Jia Liu, ChangYi He, YingQiao Lin, MingMin Yang, FeiYang Shen, and ShaoGuo Liu. 2025b. Ettrl: Balancing exploration and exploitation in llm test-time reinforcement learning via entropy mechanism. *arXiv preprint arXiv:2508.11356*.

Jiawei Liu, Thanh Nguyen, Mingyue Shang, Hantian Ding, Xiaopeng Li, Yu Yu, Varun Kumar, and Zijian Wang. 2024. Learning code preference via synthetic evolution. *arXiv preprint arXiv:2410.03837*.

Yantao Liu, Zijun Yao, Rui Min, Yixin Cao, Lei Hou, and Juanzi Li. 2025c. RM-Bench: Benchmarking Reward Models of Language Models with Subtlety and Style. In *The Thirteenth International Conference on Learning Representations*.

Zijun Liu, Peiyi Wang, Runxin Xu, Shirong Ma, Chong Ruan, Peng Li, Yang Liu, and Yu Wu. 2025d. Inference-time scaling for generalist reward modeling. *arXiv preprint arXiv:2504.02495*.

Dakota Mahan, Duy Van Phung, Rafael Rafailov, Chase Blagden, Nathan Lile, Louis Castricato, Jan-Philipp Fränken, Chelsea Finn, and Alon Albalak. 2024. Generative reward models. *arXiv preprint arXiv:2410.12832*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.

Swarnadeep Saha, Xian Li, Marjan Ghazvininejad, Jason Weston, and Tianlu Wang. 2025. Learning to plan & reason for evaluation with thinking-llm-as-a-judge. *arXiv preprint arXiv:2501.18099*.

Sheikh Shafayat, Fahim Tajwar, Ruslan Salakhutdinov, Jeff Schneider, and Andrea Zanette. 2025. Can large reasoning models self-train? *arXiv preprint arXiv:2505.21444*.

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. *arXiv preprint arXiv:2402.03300*.

Sijun Tan, Siyuan Zhuang, Kyle Montgomery, William Yuan Tang, Alejandro Cuadron, Chenguang Wang, Raluca Popa, and Ion Stoica. 2025. JudgeBench: A Benchmark for Evaluating LLM-Based Judges. In *The Thirteenth International Conference on Learning Representations*.

Carel van Niekerk, Renato Vukovic, Benjamin Matthias Ruppik, Hsien-chin Lin, and Milica Gašić. 2025. Post-training large language models via reinforcement learning from self-feedback. *arXiv preprint arXiv:2507.21931*.

Weiqin Wang, Yile Wang, Kehao Chen, and Hui Huang. 2025a. Beyond majority voting: Towards fine-grained and more reliable reward signal for test-time reinforcement learning. *arXiv preprint arXiv:2512.15146*.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *The Eleventh International Conference on Learning Representations*.

Zhilin Wang, Jiaqi Zeng, Olivier Delalleau, Hoo-Chang Shin, Felipe Soares, Alexander Bukharin, Ellie Evans, Yi Dong, and Oleksii Kuchaiev. 2025b. Helpsteer3-preference: Open human-annotated preference data across diverse tasks and languages. *arXiv preprint arXiv:2505.11475*.

Zihan Wang, Xingle Xu, Hao Wang, Yiwen Ye, Yuchen Li, Linhao Wang, Hongze Tan, Peidong Wang, Shi Feng, Guoqing Chen, and 1 others. 2025c. A survey

on entropy mechanism in large reasoning models. *Authorea Preprints*.

Chenxi Whitehouse, Tianlu Wang, Ping Yu, Xian Li, Jason Weston, Ilia Kulikov, and Swarnadeep Saha. 2025. J1: Incentivizing thinking in llm-as-a-judge via reinforcement learning. *arXiv preprint arXiv:2505.10320*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Zihuiwen Ye, Fraser David Greenlee, Max Bartolo, Phil Blunsom, Jon Ander Campos, and Matthias Gallé. 2025. Improving reward models with synthetic critiques. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 4506–4520.

Kongcheng Zhang, Qi Yao, Shunyu Liu, Yingjie Wang, Baisheng Lai, Jieping Ye, Mingli Song, and Dacheng Tao. 2025a. Consistent paths lead to truth: Self-rewarding reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.08745*.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025b. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*.

Yanzhi Zhang, Zhaoxi Zhang, Haoxiang Guan, Yilin Cheng, Yitong Duan, Chen Wang, Yue Wang, Shuxin Zheng, and Jiyan He. 2025c. No free lunch: Rethinking internal feedback for llm reasoning. *arXiv preprint arXiv:2506.17219*.

Zizhuo Zhang, Jianing Zhu, Xinmu Ge, Zihua Zhao, Zhanke Zhou, Xuan Li, Xiao Feng, Jiangchao Yao, and Bo Han. 2025d. Co-rewarding: Stable self-supervised rl for eliciting reasoning in large language models. *arXiv preprint arXiv:2508.00410*.

Xuandong Zhao, Zhewei Kang, Aosong Feng, Sergey Levine, and Dawn Song. 2025. Learning to reason without external rewards. *URL https://arxiv.org/abs/2505.19590*, 2.

Enyu Zhou, Guodong Zheng, Binghai Wang, Zhiheng Xi, Shihan Dou, Rong Bao, Wei Shen, Limao Xiong, Jessica Fan, Yurong Mou, and 1 others. 2024. Rmb: Comprehensively benchmarking reward models in llm alignment. *arXiv preprint arXiv:2410.09893*.

Yujun Zhou, Zhenwen Liang, Haolin Liu, Wenhao Yu, Kishan Panaganti, Linfeng Song, Dian Yu, Xiangliang Zhang, Haitao Mi, and Dong Yu. 2025. Evolving language models without labels: Majority drives selection, novelty promotes variation. *arXiv preprint arXiv:2509.15194*.

Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Yuchen Zhang, Xinwei Long, Ermo Hua, and 1 others. 2025. Ttrl: Test-time reinforcement learning. *arXiv preprint arXiv:2504.16084*.

A Training Configuration

A.1 Hyperparameter Settings

We implement our method using the Volcano Engine Reinforcement Learning for LLMs framework and conduct our experiments on two recent open-source base models: Qwen3-4B and Qwen3-8B. All models are optimized using the GRPO algorithm for 4 training epochs. We adopt a global batch size of 64 and a learning rate of 1e-6. The maximum generation length is set to 1024 tokens and the maximum prompt length is set to 4096 tokens. During training, we use a PPO mini-batch size of 32 and perform 8 rollouts per update step. The generation temperature is fixed at 1.0 and the top-p is set to 0.8 to encourage exploration. We additionally enable KL regularization to stabilize training, with a KL loss coefficient of 0.001.

Hyperparameter	**Value**
Training Epochs	4
Global Batch Size	64
Learning Rate	1e-6
Maximum Prompt Length	4096 tokens
Maximum Generation Length	1024 tokens
PPO Mini-Batch Size	32
Rollouts per Update Step	8
Clip Ratio	0.2
Training Temperature	1.0
Top-p	0.8
Use KL Loss	True
KL Loss Coefficient	0.001

Table 5: Implementation Details of Our Method

A.2 Training Data Construction for ConsistRM

In this study, rather than utilizing the full HelpSteer3 training dataset, we selectively curated the training data by leveraging the model itself for data filtering. Specifically, we first identified and removed examples for which the model produced outputs identical to the ground truth across all 8 rollouts. The remaining subset of data was subsequently used to construct the final training dataset. The detailed statistics of the resulting training data are summarized in Table 6.

Model	Number of Examples
Qwen3-4B	13,236
Qwen3-8B	12,098
Qwen3-14B	10,340

Table 6: Statistics of the final training dataset

A.3 Computational Resources

All experiments reported in this paper were conducted on a single server with 8×NVIDIA A800 GPUs, while Qwen3-14B training adopts two servers.

B More Analysis of ConsistRM

B.1 Effectiveness of CAAR

To further demonstrate the effectiveness of CAAR, Table 7 shows that our method effectively combines Online-State Consistency Preference and Memory-Driven Consistency Preference, leading to a steady improvement in the accuracy of consistency-aware pseudo-labels (from 80.25% to 91.5%) as training progresses. This trend demonstrates that the proposed mechanism can provide increasingly reliable and dynamically adjusted supervision signals, enabling more stable learning based on high-quality pseudo-labels in the later stages of training.

	0 steps	80 steps	160 steps	240 steps	320 steps
Acc (%)	80.25	83.45	87.31	88.25	91.50

Table 7: Accuracy of consistency-aware pseudo-labels over training steps.

B.2 Generalization of ConsistRM

To assess the generalization of ConsistRM, we train the model on the general-purpose HelpSteer3 and evaluate its performance on the code-specific preference benchmark, CodePrefBench (Liu et al., 2024). The results shown in Table 8 indicate that ConsistRM consistently outperforms the RFT baseline under training–test distribution mismatch. Despite being trained on a general-domain dataset, it achieves substantial gains on this code-oriented benchmark, highlighting its strong generalization under distribution shift. Moreover, ConsistRM consistently outperforms RFT across model scales when trained on HelpSteer3.

Model	CodePrefBench	Δ
Qwen3-4B + RFT	62.1	–
Qwen3-4B + ConsistRM	66.6	**+4.5**
Qwen3-8B + RFT	69.7	–
Qwen3-8B + ConsistRM	72.1	**+2.4**

Table 8: Performance on CodePrefBench.

B.3 Robustness of ConsistRM

To investigate the impact of incorrect pseudo-labels (i.e., false consistency), we analyze samples that conflict with the gold preference labels, as such inconsistencies naturally arise in consistency-aware pseudo-labeling. To assess their impact on model performance, we remove these inconsistent samples during training and retrain the model under identical settings. As illustrated in Table 9, this removal results in only a marginal performance change (from 73.5 to 73.3), indicating that such samples provide limited but non-negligible learning signals. This is because these erroneous samples constitute only a small fracion of the training data in later stages (approximately 9.5%), thereby limiting their overall influence on model optimization.

System	Reward Bench	PPE Pref	RM-Bench	RMB	Judge Bench	Avg
ConsistRM	85.6	67.7	78.3	79.1	56.9	73.5
w/ removal	86.1	67.9	76.2	79.2	57.2	73.3

Table 9: Effect of eliminating incorrect pseudo-labels on model performance.

C Algorithm of ConsistRM

We present the detailed pseudo-code of ConsistRM in Algorithm 1. The algorithm outlines the full consistency-aware self-training pipeline, including pseudo-label construction and reward computation.

Algorithm 1 ConsistRM: Consistency-Aware Self-Training for Generative Reward Models

Input: $x = (q, a_1, a_2)$, rollout count K, experience buffer $\mathcal{E}$, GRM π_θ
Output: Updated $\mathcal{E}$, rewards $\{r_j^{(n)}\}$
for $j = 1$ to K **do**
    $(c_j^{(n)}, y_j^{(n)}) \sim \pi_\theta(x)$
end for
$s_{\text{online}}^{(n)} \leftarrow \frac{1}{K} \sum_j y_j^{(n)}$
if $|\mathcal{E}| > 0$ **then**
    $s_{\text{memory}}^{(n)} \leftarrow \frac{1}{|\mathcal{E}|} \sum_{\hat{y} \in \mathcal{E}} \hat{y}$
else
    $s_{\text{memory}}^{(n)} \leftarrow 0$
end if
$\hat{y}^{(n)} \leftarrow \text{sgn}(s_{\text{online}}^{(n)} + s_{\text{memory}}^{(n)})$
$\mathcal{E} \leftarrow \mathcal{E} \cup \{\hat{y}^{(n)}\}$
Compute embeddings $e_j^{(n)}$ for each $c_j^{(n)}$
Compute similarity scores and obtain ranking $\text{rank}(c_j^{(n)})$
for $j = 1$ to K **do**
    if $c_j^{(n)}$ is invalid **then**
        $r_j^{(n)} \leftarrow -5$
    else if $\hat{y}^{(n)} = 0$ **then**
        $r_j^{(n)} \leftarrow 0$
    else
        $r_j^{(a,n)} \leftarrow \begin{cases} 1 & \text{if } y_j^{(n)} = \hat{y}^{(n)} \\ -1 & \text{if } y_j^{(n)} \neq \hat{y}^{(n)} \end{cases}$
        $r_j^{(c,n)} \leftarrow \begin{cases} 0.1 & \text{if } y_j^{(n)} = \hat{y}^{(n)} \wedge \text{rank}(c_j^{(n)}) \leq p \\ 0 & \text{otherwise} \end{cases}$
        $r_j^{(n)} \leftarrow r_j^{(a,n)} + r_j^{(c,n)}$
    end if
end for
return $\mathcal{E}$, $\{r_j^{(n)}\}$

D Prompts and Model Completions

This section presents sample prompts and the responses generated by the models.

D.1 Prompt for Training

For all experiments, we adopted standardized templates to regulate the output format, and explicitly required the model to first define the evaluation dimensions and elaborate on the analysis process of the two answers, before presenting a clearly delimited final answer.

Prompt Template.

Initially, consider factors such as practicality, relevance, accuracy, comprehensiveness, originality, and detail richness. Include 2 to 4 criteria for high-quality responses within the <Criterion >and </Criterion >. Subsequently, based on the <Criterion>, provide a thorough evaluation of each response within the <Analysis>and </Analysis>. Avoid any positional biases and ensure that the sequence in which the responses were presented does not affect your judgement. The length of the responses should not sway your evaluation. Lastly, after completing your evaluation, deliver your final judgement strictly adhering to this format: "Response 1 is better than Response 2" OR "Response 2 is better than Response 1" within the <Result>and </Result>.
Output Format Requirements
The special placeholders must be included.
<Criterion>
2 to 4 criteria for high-quality responses
</Criterion>
<Analysis>
Response 1:
Evaluation of Response 1 based on the given Criterion
Response 2:
Evaluation of Response 2 based on the given Criterion
</Analysis>
<Result>
Based on the output results from Criteria and Analysis, only print the following: "Response 1 is better than Response 2" OR "Response 2 is better than Response 1".
</Result>
Conversation Context & Query

Responses to be Compared

[The Begin of Response 1]

[The End of Response 1]

[The Begin of Response 2]

[The End of Response 2]

D.2 Example from RewardBench Generation

Example of RewardBench.

Conversation Context & Query
does the NHI use the top bottom or bottom up approach
Response 1:
The National Health Insurance (NHI) is a system that aims to provide universal healthcare coverage to all citizens of a country. The approach used by NHI can vary depending on the specific implementation and the country in question. However, in general, NHI systems tend to use a combination of both top-down and bottom-up approaches A top-down approach in healthcare typically involves centralized decision-making and planning, with policies and guidelines set at the national level and implemented at the local level. This approach can ensure consistency, standardization, and equity in healthcare services across the country. In the context of NHI, a top-down approach might involve the government setting national health priorities, determining the benefits package, and establishing payment rates for healthcare providers. On the other hand, a bottom-up approach in healthcare emphasizes local decision-making, community engagement, and the involvement of healthcare providers and patients in the planning and implementation of healthcare services. This approach can help ensure that healthcare services are responsive to local needs and preferences and can promote greater ownership and accountability among healthcare providers and communities. In the context of NHI, a bottom-up approach might involve local healthcare providers and communities participating in the design and implementation of healthcare services, with the national government providing support and resources. In practice, many NHI systems use a combination of both top-down and bottom-up approaches. For example, the national government might set broad policy guidelines and provide funding, while local healthcare providers and communities have the flexibility to tailor healthcare services to meet local needs and preferences. In summary, the NHI can use both top-down and bottom-up approaches, depending on the specific implementation and the country in question. A combination of both approaches can help ensure that healthcare services are equitable, responsive to local needs, and efficient.
Response 2:
The National Health Insurance (NHI) system in Taiwan uses a bottom-up approach. This means that the NHI is funded primarily through premiums collected from the population, which are then used to pay for healthcare services. The system is designed to provide universal coverage and equal access to healthcare services for all citizens, with the government subsidizing the premiums for those who cannot afford them. The bottom-up approach in the context of the NHI involves the following key features: 1. **Premium Collection**: The NHI is funded through premiums that are collected from the insured population. These premiums are based on income levels, with higher-income individuals paying higher premiums. 2. **Pooling of Funds**: The collected premiums are pooled together to create a large fund that is used to pay for healthcare services. This pooling of funds helps to spread the financial risk across the entire population, ensuring that everyone has access to healthcare when needed. 3. **Service Provision**: Healthcare providers, such as hospitals and clinics, are reimbursed by the NHI for the services they provide to insured individuals. The reimbursement rates are set by the government and are based on the type of service provided. 4. **Government Subsidies**: The government provides subsidies to help cover the cost of premiums for low-income individuals and families, ensuring that everyone has access to healthcare services regardless of their ability to pay. In summary, the NHI in Taiwan uses a bottom-up approach, with premiums collected from the population being used to fund healthcare services. The system is designed to provide universal coverage and equal access to healthcare services for all citizens, with the government subsidizing the premiums for those who cannot afford them.
Ground Truth:
Response 1 is better than Response 2

Generation of Base GRM.

<Criterion>
1. Relevance and Correctness: The response should accurately answer the user's question and provide information that is factually correct regarding the National Health Insurance (NHI) and its top-down or bottom-up approach.
2. Depth and Coverage: The response should address all important aspects of the NHI system and its operational approach, offering a thorough and well-rounded explanation.
3. Organization and Readability: The response should be clearly structured, with ideas presented in a logical sequence and easy for readers to follow.
4. Detail and Exemplification: The response should include concrete details or examples where relevant, particularly when discussing specific implementations, such as the NHI in Taiwan.
</Criterion>

<Analysis>
Response 1:
Response 1 provides a broad overview of NHI systems and correctly explains the concepts of both top-down and bottom-up approaches in healthcare. It is accurate in describing the general mechanisms and benefits of each approach, and it is well-organized and clear. However, the response remains largely abstract and does not focus on any specific country or concrete implementation. As a result, while it demonstrates strong general knowledge, it lacks concrete details that would help a user understand how the NHI operates in practice, making it less actionable for someone seeking a real-world example.

Response 2:
Response 2 focuses specifically on the NHI in Taiwan and provides a detailed explanation of how the bottom-up approach is implemented in practice. It accurately describes key mechanisms, including premium collection, pooling of funds, service reimbursement, and government subsidies, linking each to the bottom-up operational model. The response is structured clearly, presenting information in a logical sequence, and offers concrete examples that enhance understanding. While it does not discuss the top-down approach, the specificity, practical relevance, and clarity make it more directly useful and easier to apply to a real-world context compared with Response 1.
</Analysis>

<Result>Response 2 is better than Response 1 </Result>

Generation After CosistRM Training.

<Criterion>
1. **Accuracy and Relevance**: The response must correctly address the user's question and provide accurate information relevant to the National Health Insurance (NHI) and the top-bottom or bottom-up approach.
2. **Comprehensiveness**: The response should cover all relevant aspects of the NHI system and its approach, providing a thorough explanation.
3. **Clarity and Structure**: The response must be clearly organized, with logical flow and easy to understand.
4. **Specificity**: The response should provide specific examples or details where applicable, especially when discussing specific implementations like the NHI in Taiwan.
</Criterion>

<Analysis>
Response 1:
Response 1 provides a comprehensive overview of the NHI system and explains both top-down and bottom-up approaches in the context of healthcare. It correctly identifies that NHI systems can use a combination of both approaches, and it offers clear explanations of each. The response is accurate, well-structured, and relevant to the user's query. However, it is somewhat general and does not specify any particular country's NHI system, which could have been more informative given the nature of the question.

Response 2:
Response 2 is more specific, as it focuses on the NHI in Taiwan and clearly explains how it uses a bottom-up approach. It provides detailed features of the system, including premium collection, pooling of funds, service provision, government subsidies, and universal coverage. The response is accurate, relevant, and clearly structured. However, it assumes that the user is referring to the NHI in Taiwan without explicitly stating this, which may limit its applicability to other NHI systems around the world. It also does not mention the top-down approach at all, which could be seen as a limitation in addressing the broader question.
</Analysis>

<Result>Response 1 is better than Response 2 </Result>