
Reward Modeling from Natural Language Human Feedback

Zongqi Wang¹ Rui Wang² Yuchuan Wu² Yiyao Yu¹ Pinyi Zhang³ Shaoning Sun¹ Yujiu Yang¹
Yongbin Li²

Abstract

Reinforcement Learning with Verifiable Reward (RLVR) on preference data has become the mainstream approach for training Generative Reward Models (GRMs). Typically, GRMs generate reasoning chains ending with critiques and preference labels, with RLVR using label correctness as the training reward. However, we demonstrate that such binary classification tasks make GRMs susceptible to guessing correct outcomes without sound critiques, introducing noise into the reward signal and impairing learning effectiveness. To address this, we propose Reward Modeling from Natural Language Human Feedback (RM-NLHF), which leverages natural language feedback to obtain process reward signals. Specifically, we compute the similarity between GRM-generated and human critiques as the process reward, providing more accurate signals than outcome-only supervision. Considering that human critiques are difficult to scale, we introduce MetaRM which learns to predict process reward from datasets with human critiques and generalizes to data without them. Experiments on multiple benchmarks demonstrate that RM-NLHF consistently outperforms state-of-the-art models trained with outcome reward, confirming the superiority of natural language over binary feedback. Our code is publicly available at <https://github.com/Tongyi-ConvAI/Qwen-Character/tree/main/Character-GenRM-NLHF>.

1. Introduction

Recently, generative reward models (GRMs) have emerged as a powerful solution to enhance the general capabilities

of large language models (LLMs) (Liu et al., 2025; Wang et al., 2025d; Yuan et al., 2024; Guo et al., 2025b; Huang et al., 2025; Xu et al., 2025a). Unlike traditional scalar reward models that output a single numerical score, GRMs generate natural language rationales with detailed critiques before making judgments, leading to significantly higher robustness, generalizability, and accuracy through explicit reasoning (Mahan et al., 2024; Malik et al., 2025; Tan et al., 2024; Liu et al., 2024c; Wang et al., 2025d; Ye et al., 2024b; Zhang et al., 2025b; Shao et al., 2025b). Compared to rule-based verifiers that are restricted to verifiable tasks such as mathematical problems (Xie et al., 2025; Shao et al., 2024; Li et al., 2025c; Yu et al., 2025b; Li et al., 2025d) or code execution (Seed et al., 2025; Hui et al., 2024; Wang et al., 2025a), GRMs can effectively evaluate a much broader range of tasks including social intelligence (Yu et al., 2025a; Zhang et al., 2025a; Li et al., 2025a; Mao et al., 2026; Zhou et al., 2025a; Zhang et al., 2025a), roleplay (Zhou et al., 2025c; Lu et al., 2024; 2025; Qin et al., 2025; Nath et al., 2025), and image generation (Zhang et al., 2025c; Wu et al., 2025; Zhou et al., 2025d; Yang et al., 2025a), etc. Typically in a pairwise reward setting, GRMs produce a detailed CoT which mainly serves to derive effective **critiques**¹ of two responses, followed by an explicit **outcome** indicating which response is superior (Yuan et al., 2024; Guo et al., 2025b; Huang et al., 2025; Xu et al., 2025a).

The current GRM training in reinforcement learning with verifiable reward (RLVR) typically relies on outcome-only supervision, primarily through binary pairwise preference labels (Wang et al., 2025d; Yuan et al., 2024; Guo et al., 2025b). **However, we argue that while outcome reward are effective for mathematical tasks, they are fundamentally inadequate for pairwise rewarding tasks.** This disparity arises from the nature of their respective *solution spaces*: mathematical tasks typically have large solution spaces where a correct outcome strongly implies correct reasoning trajectory. In contrast, GRM tasks operate within a constrained binary solution space with only two possible outcomes (preference for response A or B). Such a binary classification task makes GRMs susceptible to guessing the correct preference label without generating sound critiques,

¹Tsinghua University ²Qwen-Character, Tongyi Lab, Alibaba ³East China Normal University. Correspondence to: Yujiu Yang <yang.yujiu@sz.tsinghua.edu.cn>, Yongbin Li <shuide.lyb@alibaba-inc.com>.

¹In this paper, we use “process” to refers to these critiques, instead of full reasoning process.

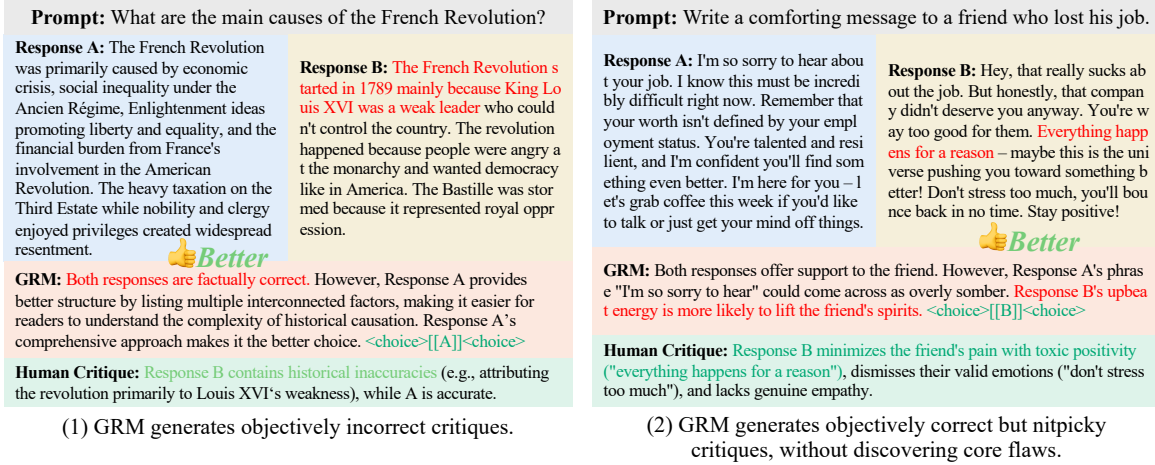


Figure 1. Examples of GRMs achieving correct outcome with flawed critiques.

e.g., objectively incorrect or nitpicky (non-essential) critiques (examples in Figure 1). We illustrate the mismatch in the proportions of process and outcome supervision between mathematical tasks and GRM tasks in Figure 2. Sometimes, mathematical tasks also exhibit small solution spaces (e.g., multiple-choice questions or true/false problems), and practitioners typically address this by either removing such data (Team et al., 2025; Cui et al., 2025) or reformulating them as fill-in-the-blank questions to expand the solution space (Albalak et al., 2025). However, in GRM tasks, the pairwise comparison formulation is inherently essential and cannot be abandoned or restructured. **As a result, given the extremely restricted solution space of binary tasks, these spurious successes introduce substantial noise into the reward signal when using RL to train GRMs, thereby impairing the learning effectiveness.** Specifically, the outcome-process misalignment resulting from outcome-only supervision might lead the model to exploit spurious correlations rather than developing genuine reasoning capabilities, causing the policy to converge toward producing erroneous critiques and thereby limiting the generalization and advancement potential of GRMs' training.

In this paper, we first empirically validate the outcome-process inconsistency issue through comprehensive analysis of mainstream GRMs, revealing that a substantial portion (20-30%) of correct preference predictions are accompanied by flawed critiques, even in state-of-the-art models (Section 2.2). We then explore various approaches for providing accurate process reward and demonstrate that leveraging the similarity of core arguments (decisive reasons that are powerful enough to justify the final choice) between human critiques and GRM-generated critiques as process reward achieves the best performance (Section 2.3).

Motivated by these observations, we first propose incorporating the similarity between *human critiques* and GRM-

generated critiques directly as an additional process reward term during GRM training, which yields significant improvements in both critique quality and outcome accuracy (Section 3.1). However, human critiques annotation faces *scalability* challenges due to high costs, and most datasets contain only outcome labels (Lai et al., 2024; Han et al., 2024; Wang et al., 2024c; Liu et al., 2024a). To address this, we propose *online MetaRM*, an auxiliary meta reward model that learns to predict process reward from limited data with human critiques and generalizes to data with outcome only. Experiments show that *online MetaRM*-based training achieves comparable performance to full human critique supervision (Section 3.2).

Our contributions can be summarized as follows:

- We demonstrate **outcome-process inconsistency** in current GRMs (Section 2.2) and show that incorporating human critique similarity can serve as an **accurate proxy for process reward** and thereby significantly improve performance (Section 2.3, Section 3.1).
- To address expensive human critique annotation, we propose an **online MetaRM** framework that learns process reward from **limited data with human critique** and generalizes to data without it, achieving remarkable performance (Section 3.2, Section 4).
- We explain four reward modeling mechanisms from a **theoretical perspective**, clarifying the evolution of reward modeling and the necessity of our method, then identify the theory-practice gaps remain (Appendix A).

Conflict of Interest Disclosure The authors Zongqi Wang, Rui Wang, Yuchuan Wu, Yongbin Li are employed by Alibaba Group, which leads the development of the Qwen model series. Some Qwen models are among those evaluated in this paper. All experiments were conducted under

fair and consistent settings across all evaluated models to ensure unbiased comparison.

2. Problem Formulation and Motivation

2.1. Problem Formulation

Pairwise rewarding tasks are designed to evaluate the comparative quality of two response candidates. A sample consists of a query q and two candidate responses, y_A and y_B , along with a preference label $l \in \{A, B\}$ indicating which response is superior. The objective of GRM π_θ is to generate reasoning with critiques for two candidates and predict a preference label $\hat{l}$ matching the ground truth. We put the detailed formulation of GRM with GRPO in Appendix F.

2.2. Motivation: Outcome–Process Inconsistency

To validate the outcome-process inconsistency, we conduct experiments on two tasks: mathematical reasoning on MATH-500 (Hendrycks et al., 2021) and pairwise rewarding on HelpSteer3 (Wang et al., 2025d). For mathematical reasoning, we evaluate outcome accuracy using a rule-based verifier and process accuracy against ground-truth solutions using gemini-2.5-pro. For pairwise rewarding, we evaluate outcome accuracy using a rule-based verifier and process accuracy via critique similarity with human annotations using gemini-2.5-pro (whose accuracy we validate in Section 2.3).

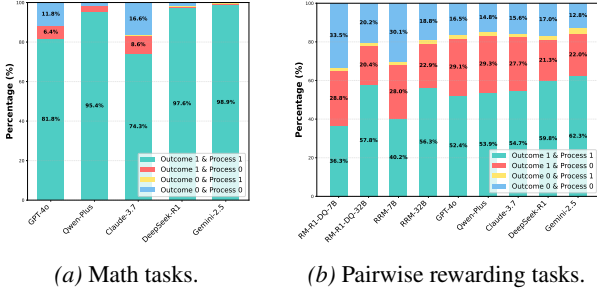


Figure 2. Inconsistency between the correctness of outcome and process across various models and tasks.

We present the results in Figure 2, which reveal striking patterns that differ between the two task types:

Mathematical Reasoning Tasks. The probability of outcome-process inconsistency is extremely low across all models. This is because mathematical reasoning has a large solution space, making it difficult to arrive at correct answers through invalid reasoning.

Pairwise Rewarding Tasks. Models frequently produce correct preference labels l but generate flawed critiques $\hat{c}$. For example, RM-R1-DeepSeek-Distilled-Qwen-7B shows this probability at 44.24%, while strong proprietary models like gemini-2.5-pro and claude-3.7-sonnet exhibit 26.10% and 33.62%, respectively. Additionally, low $P(\text{process} =$

$1 | \text{outcome} = 0)$ suggests that incorrect outcomes almost always correspond to incorrect critiques, which will also motivate our method design in Section 3.1.

Takeaway from Section 2.2

GRMs frequently produce correct outcomes despite generating invalid critiques. This high outcome-process inconsistency suggests that using outcome reward as training signals for GRMs will introduce noise, necessitating explicit process supervision. Notably, an incorrect outcome almost invariably indicates flawed critiques.

2.3. Motivation: Exploring Proxy for Process Reward

In this section, we explore how to design accurate proxy for process reward. To this end, we first carefully curate and annotate a small set of 49 samples from HelpSteer3. This dataset comprises queries q , response pairs y_a and y_b , GRM-generated critiques $\hat{c}$, human critiques h originally provided in HelpSteer3, and a label $z \in \{0, 1\}$ indicating whether the critique $\hat{c}$ is correct.

We investigate three candidate approaches for obtaining process reward: (1) **LLM-as-a-Meta-Judge**, which employs an external LLM to directly evaluate the correctness of $\hat{c}$; (2) **Similarity w/ All HC**, which uses an external LLM to extract all critiques mentioned in h and $\hat{c}$, then measures their similarity through three variants: F1, Recall, and Precision; (3) **Similarity w/ Core HC**, which employs an external LLM to identify and extract only the core arguments (excluding nitpicky critiques) in h and $\hat{c}$, then computes their similarity. For brevity, we defer the detailed experimental setup to Appendix H.

In particular, nitpicky critiques refer to irrelevant arguments introduced by human annotators that should not serve as the basis for final judgments. For example, in HelpSteer3, such critiques are often merely casual suggestions for improvement rather than substantive differentiators. We therefore filter only truly irrelevant points, while retaining minor but still valid critiques. The filtering strategy is based on the signals provided by human annotators in the critiques within HelpSteer3. Since human critiques naturally exhibit this characteristic, we further verify the effect of filtering such arguments in this section.

Table 1. Accuracy (%) of different methods for process reward. We adopt three variants (F1, Recall, and Precision) to calculate similarity between human and GRM-generated critiques.

Model	Cost (\$/M)	LLM-as-a-Meta-Judge	Similarity w/ All HC			Similarity w/ Core HC		
			w/ F1	w/ Recall	w/ Precision	w/ F1	w/ Recall	w/ Precision
gemini-2.5-pro	15.00	0.7347	0.8571	0.9388	0.7959	<u>0.9184</u>	<u>0.9184</u>	0.8980
gpt-5-mini	2.00	0.4898	0.7755	<u>0.8162</u>	0.7347	0.8571	0.8571	<u>0.8162</u>
gpt-4o-mini	0.60	0.2653	<u>0.7551</u>	<u>0.7551</u>	0.7755	0.7347	0.7347	<u>0.7551</u>

Table 1 presents the performance of different process reward

design approaches. We observe that the LLM-as-a-Meta-Judge approach is substantially outperformed by Similarity w/ HC-based methods across all models, with performance gaps ranging from 12% to 59%. On average, Similarity w/ Core HC achieves better performance than Similarity w/ All HC, indicating that focusing on essential critical points provides more reliable process supervision. While the highest performance (0.9388) is achieved by gemini-2.5-pro with Recall-based Similarity w/ All HC, we adopt the more cost-effective gpt-5-mini with F1-based Similarity w/ Core HC, which balances accuracy, cost-effectiveness, and training stability (Recall-based Similarity tends to cause reward hacking as discussed in Appendix D).

Takeaway from Section 2.3

To address the outcome-process inconsistency, we need an accurate proxy for process reward. Our experiments reveal that computing the F1-based similarity between core arguments in human critiques and GRM-generated critiques serves as an effective proxy for process reward.

3. RM-NLHF

In this section, we first directly use the F1-based similarity between human critiques and GRM-generated critiques as process reward to guide GRM training, achieving promising results (Section 3.1). Next, we propose online MetaRM to address the scalability challenges of human critique annotation (Section 3.2).

3.1. Natural Language Critique as Process Reward

Motivated by the findings in Section 2.3, we use the F1-based similarity between human critiques and GRM-generated critiques as a process reward signal. In this section, we incorporate this signal into GRM training via a composite reward that combines process- and outcome-level supervision.

Reward Design. The process reward is defined as:

$$R_{\text{process}} = \begin{cases} 1, & \text{if } S(h, \hat{c}) > 0.5, \\ 0, & \text{otherwise,} \end{cases} \quad (1)$$

where $S(h, \hat{c})$ measures the F1-based similarity of core arguments of human critiques h and GRM-generated critiques $\hat{c}$ (we adopt gpt-5-mini as calculator here).

To incorporate human critiques for process supervision while maintaining outcome correctness, we design a composite reward function that balances both aspects. The final

reward is defined as:

$$R = \begin{cases} -1, & \text{if output format is invalid,} \\ 0, & \text{if } \hat{l} \neq l, \\ 1 + \lambda \cdot R_{\text{process}}, & \text{if } \hat{l} = l, \end{cases} \quad (2)$$

where $\lambda \in [0, 1]$ controls the weight of process supervision. We adopt **outcome regularization**, i.e., we exclude process reward when outcomes are incorrect since they indicate invalid critiques (Section 2.2).

Experimental Results. We train the model using GRPO for one epoch on the HelpSteer3 training set. To evaluate the impact of incorporating human critiques as process reward, we compare training with and without human critiques on the HelpSteer3 validation set (Figure 3). Detailed experimental settings are provided in Appendix I.1.

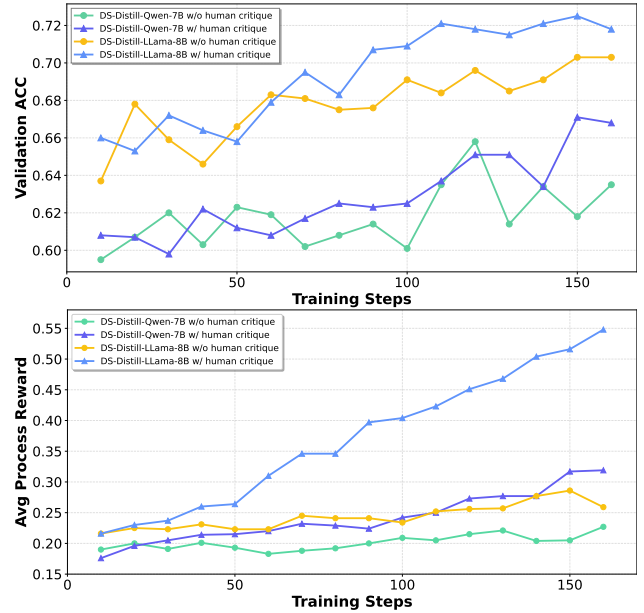


Figure 3. Comparison between outcome-only reward and combining outcome reward with F1-based similarity between GRM-generated and human critique as process reward.

We observe that, for both Llama-8B and Qwen-7B, incorporating human critiques as process reward consistently outperforms outcome-only training. Models trained with human critiques begin to show stable improvements after approximately 70 steps and maintain this advantage for the remainder of training.

Takeaway from Section 3.1

Incorporating F1-based similarity of core arguments between human and GRM-generated critiques as process reward improves GRMs' performance compared to using outcome-only reward.

3.2. Online MetaRM as Process Reward

Overview. Incorporating human critiques as process reward has been shown to be effective. However, collecting such critiques is expensive, which poses significant challenges for scaling RM-NLHF to large datasets. To address this limitation, we propose *online MetaRM*, a scalable framework that predicts process reward from limited data with human critique and generalizes to data with outcome only.

Online MetaRM. Considering that we already have a subset of data with human critiques, it is natural to use a model to fit the process rewards on this subset and generalize to the remaining data. Our MetaRM is built upon this idea. Specifically, the MetaRM M_ϕ is a regression model that learns to predict process reward of GRM-generated critique $\hat{c}$ given a query q and two candidate responses y_A and y_B .

A key design choice is our adoption of online MetaRM. The distinction between offline and online MetaRM lies in their updating strategies. The **offline MetaRM** is trained once on initial GRM-generated critiques and remains fixed during GRM training, which may suffer from distribution mismatch and reward hacking as the policy evolves. In contrast, the **online MetaRM** continuously adapts alongside policy updates, dynamically adjusting to the changing critique distribution. This online update strategy addresses the distribution shift challenge by ensuring that the reward model remains aligned with the evolving policy throughout training, thereby maintaining prediction accuracy on newly generated critiques and reducing reward hacking.

For the workflow of the online MetaRM framework, see Figure 4. For detailed information on MetaRM architecture and training, see Appendix G.1. For complete algorithm and implementation details, refer to Appendix G.2.

Experimental Results. We provide detailed experimental settings in Appendix I.2. Results shown in Figure 5 lead to the following conclusions:

(1) **Naive combination is catastrophic.** Simply applying process reward to data with human critiques while using outcome-only reward for other data performs even worse than using outcome-only reward for both datasets. We attribute this to conflicting reward signals that undermine reward consistency.

(2) **Online MetaRM outperforms offline MetaRM.** Online MetaRM consistently outperforms its offline counterpart, highlighting the importance of continuous adaptation to the evolving critique distribution.

(3) **Online MetaRM achieves performance with full human critique supervision.** Compared to using human critiques for both datasets, online MetaRM achieves comparable results while significantly reducing annotation

costs. Note that this conclusion may not hold for out-of-distribution settings, which we will thoroughly investigate in subsequent experiments.

We provide extensive ablation studies on MetaRM design in Table 11 and show the accuracy of process reward from offline/online MetaRM throughout training in Figure 7.

Takeaway from Section 3.2

We find that naively assigning different rewards to different datasets proves catastrophic to performance. In contrast, online MetaRM substantially outperforms offline methods and achieves performance comparable to full human supervision while requiring significantly fewer resources.

4. Experiments

4.1. Experimental Setup

We evaluate on 6 benchmarks released within one month before the base model’s release to avoid data contamination. Our training set contains 164K pairwise preference samples, using GRPO for RL training. We compare RM-NLHF against zero-shot LLM-as-a-Judge, scalar reward models and GRMs trained with same base models (RM-R1, RRM) and baselines. Details are in Appendix J.

4.2. Main Results

Table 2 presents the main results across multiple benchmarks, from which we observe:

(1) Compared to specialized trained GRMs with outcome-only reward sharing the same base model, RM-NLHF achieves state-of-the-art performance, demonstrating its effectiveness in producing accurate reward. Specifically, RM-NLHF-Qwen-7B attains an overall score of 0.6481, substantially outperforming RM-R1-Qwen-7B (0.5759) and RRM-Qwen-7B (0.5895).

(2) Comparing generative reward models with scalar reward models, we find that scalar reward models exhibit superior performance at the 7/8B scale. However, we observe that even when scaling up to 70B, scalar reward models show marginal performance gains. In contrast, generative reward models demonstrate substantial improvements from 7B to 32B. This indicates that generative reward models scale more effectively with model size, while scalar reward models exhibit limited scalability.

4.3. Ablation Study

In this section, we provide the main ablation study results in Table 3, addressing the following questions.

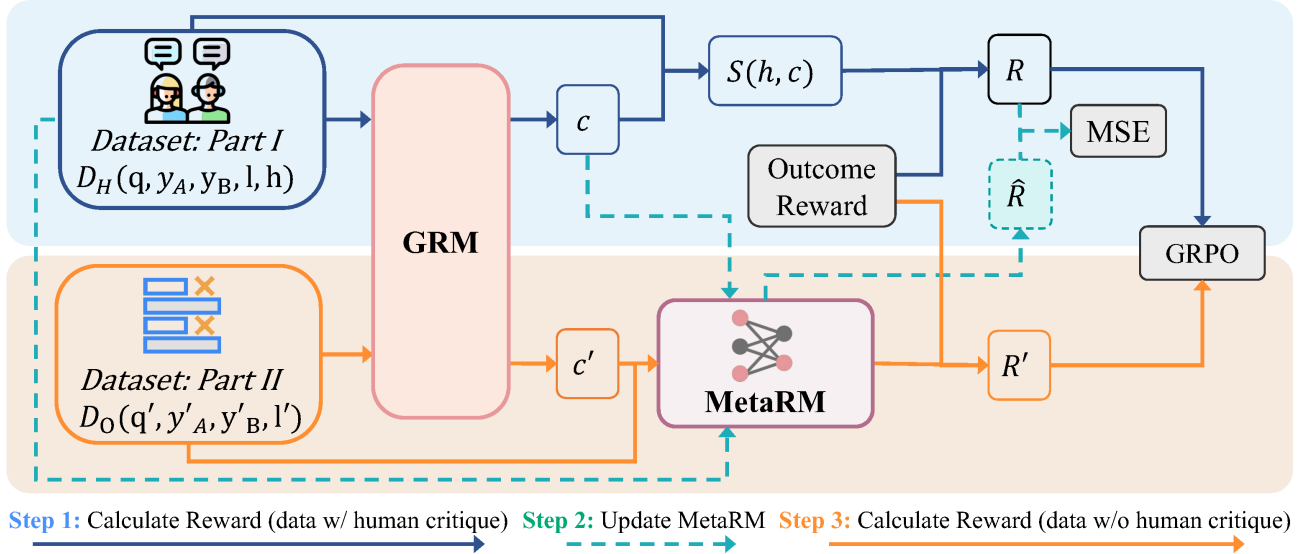


Figure 4. Workflow of online MetaRM framework. It operates in a three-step process. First, for samples in $\mathcal{D}_H$ (data with human critiques), process rewards are computed by measuring the similarity between human and GRM-generated critiques using Equation 2. Second, these computed rewards serve as supervision signals to train the MetaRM. Third, for samples in $\mathcal{D}_O$ (data with only outcome label), the trained MetaRM predicts process rewards, providing supervision where human critiques are unavailable. This design allows the GRM to be trained with complementary reward signals: human-critique-based reward for $\mathcal{D}_H$ and MetaRM-predicted reward for $\mathcal{D}_O$.

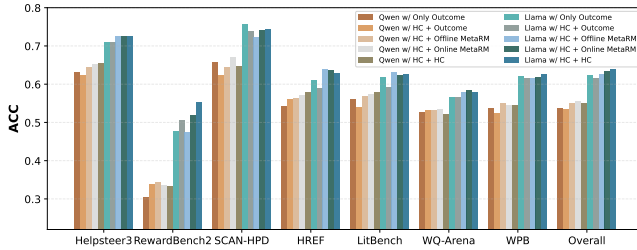


Figure 5. Comparison of different reward combinations. In all settings, $\mathcal{D}_H$ uses the similarity between GRM-generated and human critique as process reward. "+ Outcome" refers to $\mathcal{D}_O$ using only outcome reward. "+ online/offline MetaRM" refers to $\mathcal{D}_O$ using online/offline MetaRM as process reward. "+ HC" refers to $\mathcal{D}_O$ using human critique as process reward, identical to $\mathcal{D}_H$.

Ablation on Process Reward. We train a comparison model by removing process reward while keeping all other configurations (training parameters and data) same. The results show that incorporating process reward yields improvements across all benchmarks, validating the effectiveness and generalizability of process reward. Since we leverage human critiques from HelpSteer3 and generalize to other datasets, this verifies the transferability of HelpSteer3’s human critiques.

Ablation on Outcome Regularization. In this part, we try to assign process reward even when the outcome is incorrect. However, the results show a substantial decrease in accuracy, which validates the importance of outcome regularization. We visualize the training process in Figure 6b,

which demonstrates that both models are initially consistent, but the model without outcome regularization shows stagnant or even declining reward in later stages. This confirms the importance of outcome regularization for maintaining stable long-term RL training.

Generalizability of Human Critiques in HelpSteer3.

Since our training data combines HelpSteer3 with data from other sources, we verify whether the human critiques from HelpSteer3 can transfer to other datasets. Specifically, we attempt to also use $\mathcal{D}_O$ for training MetaRM, where samples with correct and incorrect outcomes are assigned training labels of 0 and $1 + \lambda/2$, respectively. This aims to adapt MetaRM to critique distributions beyond HelpSteer3. However, we find that training MetaRM solely on HelpSteer3 yields better performance, indicating that HelpSteer3 have provided broad coverage and sufficient transferability for training an effective MetaRM.

4.4. Performance of Downstream Tasks

To verify the effectiveness of GRMs on downstream tasks, beyond following prior work (Guo et al., 2025c) using Best-of-N (BoN), we additionally evaluate the quality of GRM-generated critiques through a Feedback-Edit approach. For BoN, we adopt a tournament-based approach where the pairwise GRM selects the best response from N responses sampled from the base model. For Feedback-Edit, we use RM-NLHF to select the top 2 responses, then apply GRMs to generate critiques. An edit-model (gemini-2.5-pro) subsequently synthesizes a new response based on these critiques,

Table 2. Main Results.

Model	HelpSteer3	Reward Bench V2	SCAN-HPD	HREF	LitBench	WQ_Arena	WPB	Overall
<i>Base Models (LLM-as-a-Judge)</i>								
gpt-5-2025-08-07	0.8245	0.8344	0.8147	0.7595	0.7617	0.6304	0.5939	0.7456
o3-2025-04-16	0.8315	0.8009	0.8147	0.7058	0.7202	0.6432	0.6234	0.7342
gemini-2.5-pro	0.8207	0.7117	0.8288	0.6934	0.7181	0.6529	0.6367	0.7232
claude-3-7-sonnet-20250219	0.8042	0.7714	0.7524	0.7384	0.7631	0.6300	0.5967	0.7223
deepseek-r1-0528	0.7827	0.7292	0.8227	0.6496	0.7000	0.6348	0.6183	0.7053
qwen-plus-latest	0.8038	0.7717	0.8048	0.6488	0.6835	0.5961	0.6278	0.7052
qwen3-max	0.8102	0.7169	0.7865	0.6219	0.6859	0.6062	0.6133	0.6916
gpt-4o-latest	0.7896	0.6478	0.7572	0.5777	0.6319	0.5799	0.5883	0.6532
deepseek-v3	0.7522	0.5436	0.7636	0.5678	0.6536	0.5848	0.6122	0.6397
DeepSeek-R1-Distill-Qwen-7B	0.5999	0.3052	0.6015	0.5487	0.5418	0.5365	0.5161	0.5214
R1-Distill-Llama-8B	0.6400	0.4095	0.6592	0.5899	0.5822	0.5650	0.5841	0.5757
R1-Distill-Qwen-32B	0.7376	0.6154	0.7492	0.6967	0.6206	0.6115	0.6389	0.6671
<i>Scalar Reward Models</i>								
ArmoRM-Llama3-8B-v0.1	0.7640	0.6897	0.6628	0.7322	0.7157	0.5855	0.4495	0.6571
URM-LLaMa-3.1-8B	0.8012	0.8004	0.7013	0.7347	0.6302	0.5814	0.4884	0.6768
Skywork-Reward-Llama-3.1-8B-v0.2	0.7950	0.7907	0.7205	0.7347	0.6593	0.5743	0.5322	0.6867
INF-ORM-Llama3.1-70B	0.8075	0.8066	0.7684	0.7364	0.7141	0.5978	0.5967	0.7182
<i>Specialized Generative Reward Models</i>								
RM-R1-Qwen-7B	0.6499	0.4679	0.6438	0.6504	0.5750	0.5181	0.5261	0.5759
RRM-Qwen-7B	0.6794	0.5082	0.6645	0.6777	0.5383	0.5421	0.5161	0.5895
RRM-Qwen-32B	0.7942	0.7340	0.7604	0.7273	0.6746	0.5875	0.6283	0.7009
RM-R1-Qwen-32B	0.7818	0.7260	0.7795	0.7099	0.6934	0.5988	0.6367	0.7037
<i>Our Generative Reward Models</i>								
RM-NLHF-Qwen-7B	0.7381	0.5757	0.6822	0.6926	0.6583	0.5416	0.5521	0.6481
RM-NLHF-Qwen-32B	0.8315	0.7867	0.7888	0.7165	0.7492	0.6161	0.6183	0.7296

Table 3. Main Ablation Study.

Model	HS3	RB-V2	SCAN	HREF	LB	WQA	WPB	AVG
RM-NLHF	0.7381	0.5757	0.6822	0.6926	0.6583	0.5416	0.5521	0.6481
w/o Process Reward	0.7125	0.5440	0.6709	0.6650	0.6540	0.5314	0.5325	0.6158
w/o Outcome Regularization	0.7296	0.5529	0.6944	0.6818	0.6512	0.5337	0.5539	0.6282
w/ D_o for MetaRM	0.7291	0.5684	0.7109	0.6857	0.6569	0.5363	0.5456	0.6332

with the prompt explicitly requiring modifications guided solely by the critiques (see prompt in Figure 13).

As shown in Table 4, RM-NLHF consistently outperforms the outcome-only baseline across MATH500 (Lightman et al., 2023), HumanEval+ (Liu et al., 2023), and Arena-Hard-V2.0 (Li et al., 2025b). In the BoN setting, RM-NLHF demonstrates superior ranking capability, with particularly notable improvements on HumanEval+. In the Feedback-Edit setting, RM-NLHF achieves substantial gains over the baseline, validating that process reward enable the generation of higher-quality critiques for response refinement.

4.5. Performance of Downstream Tasks through Reinforcement Learning

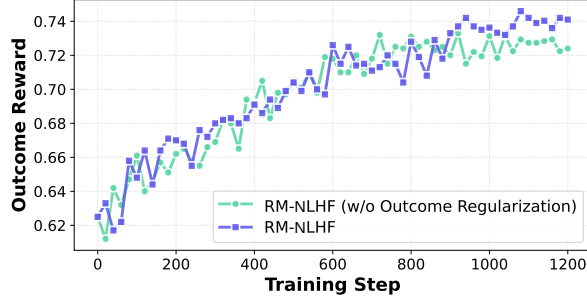
In this section, we adopt RM-NLHF-Qwen-7B and Outcome-Only-Qwen-7B as the reward models to perform RL training on DeepSeek-Distilled-Qwen-7B. We use prompts from HelpSteer3 as the training set and apply

Table 4. Evaluation results on downstream tasks.

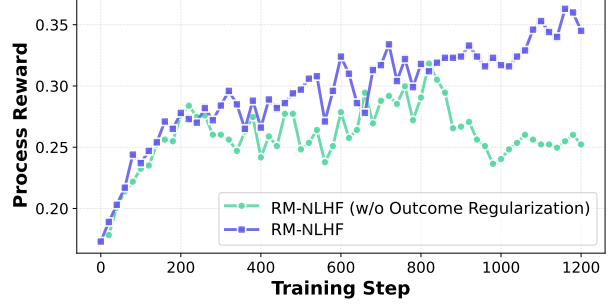
Method	MATH500	HumanEval+	Arena-Hard-V2.0
Base Model			
DeepSeek-Distilled-Qwen-7B	62.92%	77.13%	3.39%
Best-of-N (BoN)			
Outcome-only (BoN@2)	63.65%	76.30%	3.69%
RM-NLHF (BoN@2)	64.90%	76.95%	3.56%
Outcome-only (BoN@4)	65.45%	75.77%	3.93%
RM-NLHF (BoN@4)	66.80%	81.04%	3.85%
Outcome-only (BoN@8)	65.99%	75.00%	4.30%
RM-NLHF (BoN@8)	67.60%	85.98%	4.64%
Feedback-Edit			
Outcome-only	67.01%	82.32%	6.55%
RM-NLHF	68.40%	87.20%	7.03%

GRPO for 100 training steps. Since both reward models are pairwise in nature, we convert them into pointwise scores via Double-Elimination Tournament (Zhang et al., 2026).

Results presented in Table 5 demonstrate that our method holds a substantial advantage over the baseline. Across a



(a) Outcome accuracy on HelpSteer3.



(b) F1-based critique similarity on HelpSteer3.

Figure 6. Training dynamics of RM-NLHF with and without outcome regularization.

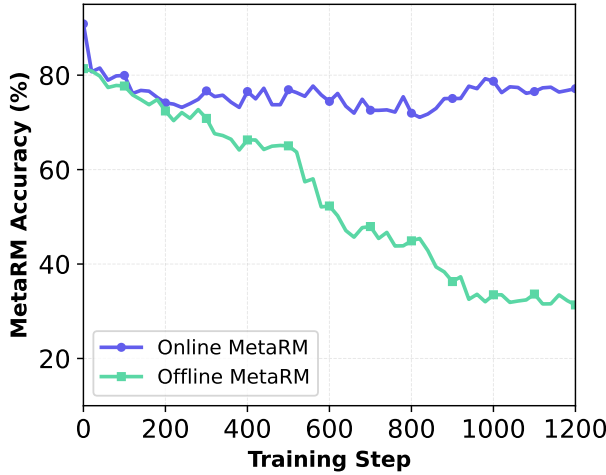


Figure 7. Effectiveness of online updating. Online MetaRM maintains higher accuracy than offline MetaRM.

Table 5. Win rates (%) of RL-trained models using RM-NLHF vs. Outcome-Only as the reward model, evaluated on Arena-Hard-V2.0 (Li et al., 2025b) with different judge models.

	Judge Model				Avg.
	gemini-3-pro	gpt-5.1	qwen3-max-thinking	kimi-k2.5	
RM-NLHF Win	57.70	54.96	49.47	47.19	52.33
Outcome-Only Win	34.40	37.67	43.98	41.58	39.41
Tie	7.90	7.37	6.55	11.23	8.26

diverse set of SOTA judge models (gemini-3-pro, gpt-5.1, qwen3-max-thinking and kimi-k2.5), our approach outperforms the baseline by an average margin of 12.92%.

Since GRM is known to exhibit a length bias (Dubois et al., 2024), we further analyze the output length throughout training, as shown in Figure 8. The model trained with RM-NLHF actually produces shorter responses than the baseline, confirming that the gains stem from improved quality rather than verbosity.

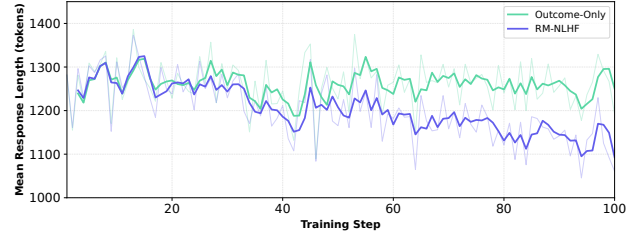


Figure 8. Mean response length during downstream RL training.

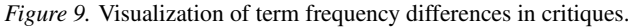
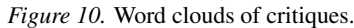
4.6. How Does MetaRM’s Online Updating Take Effect?

To demonstrate the importance of online MetaRM training, we evaluate MetaRM’s accuracy throughout the training process using the validation set from HelpSteer3. Accuracy is defined as the agreement between MetaRM-assigned scores and those from the F1-based similarity between critiques (gemini-2.5-pro). We report the results in Figure 7, which clearly demonstrates that online updates maintain high accuracy by continuously adapting to the evolving policy model’s output distribution. In contrast, while the offline MetaRM starts with comparable accuracy, its performance gradually deteriorates throughout RL training, highlighting the critical advantage of online training for maintaining the effectiveness during iterative long-term RL optimization.

4.7. How Does Human Critique Take Effect?

To examine how human critique as process reward changes GRM-generated critiques, we conduct linguistic analysis on the HelpSteer3 validation set, comparing RM-NLHF against a baseline trained with outcome-only reward.

We first examine vocabulary diversity by counting unique adjectives and verbs in the generated critiques. The results show that RM-NLHF produces critiques with 4,436 unique words, compared to only 1,633 for the baseline, indicating that **(1) RM-NLHF generates substantially more diverse and specific feedback**. We further analyze word frequency differences in Figure 9 and future visualize them



4.8. Ablation on Discretizing Process Rewards

tion in Reinforcement Learning (RL) pipelines utilizing LLM-based rewards. To empirically validate that discretization is beneficial, we conduct an ablation study comparing the discretized (binarized) variant against a continuous scoring baseline. As shown in Table 6, the discrete variant consistently outperforms the continuous one across most benchmarks, achieving an average improvement of over 2.2%. This result confirms that discretization serves as a crucial component for robust performance in our framework.

Method	HS3	RB-V2	SCAN	HREF	LB	WQA	WPB	AVG
Discrete Score	0.6723	0.3358	0.6840	0.5554	0.5686	0.5309	0.5582	0.5579
Continuous Score	0.6293	0.3314	0.6486	0.5701	0.5204	0.5204	0.5259	0.5352

Several recent works have explored using language feedback for reward modeling, but our approach differs fundamentally. CCloud (Ankner et al., 2024) uses off-the-shelf LLMs to generate critiques and trains a scalar reward model on them, while we specifically train the model generating critiques. Seq2Seq Reward Modeling (Zhou et al., 2025b) also proposes using language feedback for reward modeling. However, our method differs fundamentally in that their RM does not score responses directly but instead generates an "ideal response" and derives the reward through comparison with the generated response. RLBBF (Wang et al., 2025c) extracts a set of principles that can be assessed via binary judgments (e.g., "yes/no") from language feedback and provides them as prior knowledge to the RM. In contrast, we internalize language feedback within the RM itself rather than requiring it to be pre-specified. Moreover, we are not limited to binary judgments. FCP (Luo et al., 2025) uses language feedback directly for training policy models, whereas we focus on learning robust reward models.

In this work, we identified and addressed the outcome-process inconsistency problem in generative reward models. We incorporated human critiques as process reward to improve critique quality and preference accuracy. To address scalability challenges, we proposed MetaRM, a meta reward model that learns from limited human critiques and generalizes to unlabeled data. We further introduced an online MetaRM that adapts to distribution shifts during training. Experiments show RM-NLHF achieves performance higher than outcome-only supervision. We highly recommend readers refer to our discussion on how to understand reward modeling in Appendix A, and view code at <https://github.com/Tongyi-ConvAI/Qwen-Character/tree/main/Character-GenRM-NLHF>.

Acknowledgement

This work was partly supported by the National Natural Science Foundation of China (Grant No. 62576191), the Shenzhen Science and Technology Program (ZDCY20250901103533010) and Tsinghua SIGS KA Co-operation Fund.

Impact Statement

This paper presents work on improving reward models for large language models, with the goal of advancing the field of Machine Learning. Our method enhances the alignment of language models with human preferences through more accurate reward signals. The broader impacts of this work are consistent with general research on LLM alignment and safety. We believe this work contributes positively to developing more reliable and helpful AI systems, with no specific ethical concerns that must be highlighted here.

References

- Albalak, A., Phung, D., Lile, N., Rafailov, R., Gandhi, K., Castricato, L., Singh, A., Blagden, C., Xiang, V., Mahan, D., et al. Big-math: A large-scale, high-quality math dataset for reinforcement learning in language models. *arXiv preprint arXiv:2502.17387*, 2025.
- Ankner, Z., Paul, M., Cui, B., Chang, J. D., and Ammanabrolu, P. Critique-out-loud reward models. *arXiv preprint arXiv:2408.11791*, 2024.
- Anugraha, D., Tang, Z., Miranda, L. J. V., Zhao, H., Farhan-syah, M. R., Kuwanto, G., Wijaya, D., and Winata, G. I. R3: Robust rubric-agnostic reward models. *arXiv preprint arXiv:2505.13388*, 2025.
- Chakrabarty, T., Laban, P., and Wu, C.-S. Ai-slop to ai-polish? aligning language models through edit-based writing rewards and test-time computation. *arXiv preprint arXiv:2504.07532*, 2025.
- Chen, X., Li, G., Wang, Z., Jin, B., Qian, C., Wang, Y., Wang, H., Zhang, Y., Zhang, D., Zhang, T., et al. Rm-r1: Reward modeling as reasoning. *arXiv preprint arXiv:2505.02387*, 2025.
- Cui, G., Yuan, L., Wang, Z., Wang, H., Li, W., He, B., Fan, Y., Yu, T., Xu, Q., Chen, W., et al. Process reinforcement through implicit rewards. *arXiv preprint arXiv:2502.01456*, 2025.
- Dubois, Y., Galambosi, B., Liang, P., and Hashimoto, T. B. Length-controlled alpaca-eval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 2024.
- Fein, D., Russo, S., Xiang, V., Jolly, K., Rafailov, R., and Haber, N. Litbench: A benchmark and dataset for reliable evaluation of creative writing. *arXiv preprint arXiv:2507.00769*, 2025.
- Frick, E., Li, T., Chen, C., Chiang, W.-L., Angelopoulos, A. N., Jiao, J., Zhu, B., Gonzalez, J. E., and Stoica, I. How to evaluate reward models for rlhf. In *The Thirteenth International Conference on Learning Representations*, 2025.
- Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025a.
- Guo, J., Chi, Z., Dong, L., Dong, Q., Wu, X., Huang, S., and Wei, F. Reward reasoning model. *arXiv preprint arXiv:2505.14674*, 2025b.
- Guo, J., Chi, Z., Dong, L., Dong, Q., Wu, X., Huang, S., and Wei, F. Reward reasoning model. *arXiv preprint arXiv:2505.14674*, 2025c.
- Han, S., Rao, K., Ettinger, A., Jiang, L., Lin, B. Y., Lambert, N., Choi, Y., and Dziri, N. Wildguard: Open one-stop moderation tools for safety risks, jailbreaks, and refusals of llms. *Advances in Neural Information Processing Systems*, 37:8093–8131, 2024.
- Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. Measuring mathematical problem solving with the math dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.
- Hu, J., Zhang, Y., Han, Q., Jiang, D., Zhang, X., and Shum, H.-Y. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*, 2025.
- Huang, H., He, Y., Zhou, H., Zhang, R., Liu, W., Wang, W., Su, W., Zheng, B., and Liu, J. Think-j: Learning to think for generative llm-as-a-judge. *arXiv preprint arXiv:2505.14268*, 2025.
- Hui, B., Yang, J., Cui, Z., Yang, J., Liu, D., Zhang, L., Liu, T., Zhang, J., Yu, B., Lu, K., et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.
- Kim, S., Suk, J., Longpre, S., Lin, B. Y., Shin, J., Welleck, S., Neubig, G., Lee, M., Lee, K., and Seo, M. Prometheus 2: An open source language model specialized in evaluating other language models. *arXiv preprint arXiv:2405.01535*, 2024.

- Lai, X., Tian, Z., Chen, Y., Yang, S., Peng, X., and Jia, J. Step-dpo: Step-wise preference optimization for long-chain reasoning of llms. *arXiv preprint arXiv:2406.18629*, 2024.
- Lambert, N., Morrison, J., Pyatkin, V., Huang, S., Ivison, H., Brahman, F., Miranda, L. J. V., Liu, A., Dziri, N., Lyu, S., et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.
- Lambert, N., Pyatkin, V., Morrison, J., Miranda, L. J. V., Lin, B. Y., Chandu, K., Dziri, N., Kumar, S., Zick, T., Choi, Y., et al. Rewardbench: Evaluating reward models for language modeling. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1755–1797, 2025.
- Li, H., Tjandrasuwita, M., Fung, Y. R., Solar-Lezama, A., and Liang, P. P. Mimeqa: Towards socially-intelligent nonverbal foundation models. *arXiv preprint arXiv:2502.16671*, 2025a.
- Li, T., Chiang, W.-L., Frick, E., Dunlap, L., Wu, T., Zhu, B., Gonzalez, J. E., and Stoica, I. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. In *Forty-second International Conference on Machine Learning*, 2025b.
- Li, X., Bao, K., Ma, Y., Li, M., Wang, W., Men, R., Zhang, Y., Feng, F., Liu, D., and Lin, J. Mtr-bench: A comprehensive benchmark for multi-turn reasoning evaluation. *arXiv preprint arXiv:2505.17123*, 2025c.
- Li, X., Zou, H., and Liu, P. Torl: Scaling tool-integrated rl. *arXiv preprint arXiv:2503.23383*, 2025d.
- Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let’s verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.
- Liu, C. Y., Zeng, L., Liu, J., Yan, R., He, J., Wang, C., Yan, S., Liu, Y., and Zhou, Y. Skywork-reward: Bag of tricks for reward modeling in llms. *arXiv preprint arXiv:2410.18451*, 2024a.
- Liu, J., Xia, C. S., Wang, Y., and Zhang, L. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. *Advances in Neural Information Processing Systems*, 36:21558–21572, 2023.
- Liu, T., Xiong, W., Ren, J., Chen, L., Wu, J., Joshi, R., Gao, Y., Shen, J., Qin, Z., Yu, T., et al. Rrm: Robust reward model training mitigates reward hacking. *arXiv preprint arXiv:2409.13156*, 2024b.
- Liu, Y., Yao, Z., Min, R., Cao, Y., Hou, L., and Li, J. Rm-bench: Benchmarking reward models of language models with subtlety and style. *arXiv preprint arXiv:2410.16184*, 2024c.
- Liu, Z., Wang, P., Xu, R., Ma, S., Ruan, C., Li, P., Liu, Y., and Wu, Y. Inference-time scaling for generalist reward modeling. *arXiv preprint arXiv:2504.02495*, 2025.
- Lou, X., Yan, D., Shen, W., Yan, Y., Xie, J., and Zhang, J. Uncertainty-aware reward model: Teaching reward models to know what is unknown. *arXiv preprint arXiv:2410.00847*, 2024.
- Lu, J., Li, J., Shen, G., Gui, L., An, S., He, Y., Yin, D., and Sun, X. Rolemrc: A fine-grained composite benchmark for role-playing and instruction-following. *arXiv preprint arXiv:2502.11387*, 2025.
- Lu, K., Yu, B., Zhou, C., and Zhou, J. Large language models are superpositions of all characters: Attaining arbitrary role-play via self-alignment. *arXiv preprint arXiv:2401.12474*, 2024.
- Luo, R., Liu, Z., Liu, X., Du, C., Lin, M., Chen, W., Lu, W., and Pang, T. Language models can learn from verbal feedback without scalar rewards. *arXiv preprint arXiv:2509.22638*, 2025.
- Lyu, X., Wang, Y., Hajishirzi, H., and Dasigi, P. Href: Human response-guided evaluation of instruction following in language models. *arXiv preprint arXiv:2412.15524*, 2024.
- Mahan, D., Van Phung, D., Rafailov, R., Blagden, C., Lile, N., Castriato, L., Fränken, J.-P., Finn, C., and Albalak, A. Generative reward models. *arXiv preprint arXiv:2410.12832*, 2024.
- Malik, S., Pyatkin, V., Land, S., Morrison, J., Smith, N. A., Hajishirzi, H., and Lambert, N. Rewardbench 2: Advancing reward model evaluation. *arXiv preprint arXiv:2506.01937*, 2025.
- Mao, X., Tao, E. X., and Wang, L. Mindvote: When ai meets the wild west of social media opinion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 32492–32500, 2026.
- Minghao Yang, Chao Qu, X. T. Inf-orm-llama3.1-70b, 2024. URL [<https://huggingface.co/infly/INF-ORM-Llama3.1-70B>] (<https://huggingface.co/infly/INF-ORM-Llama3.1-70B>).
- Nath, A., Graff, C., and Krishnaswamy, N. Collaborate, deliberate, evaluate: How llm alignment affects coordinated multi-agent outcomes. *arXiv preprint arXiv:2509.05882*, 2025.

- Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.
- Park, J., Jwa, S., Meiyang, R., Kim, D., and Choi, S. Offset-bias: Leveraging debiased data for tuning evaluators. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 1043–1067, 2024.
- Qin, H., Zhang, J., Zhang, W., Lu, K., Zhou, M., Liao, H., and Mao, R. R-char: A metacognition-driven framework for role-playing in large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 26984–27002, 2025.
- Saha, S., Levy, O., Celikyilmaz, A., Bansal, M., Weston, J., and Li, X. Branch-solve-merge improves large language model evaluation and generation. *arXiv preprint arXiv:2310.15123*, 2023.
- Saha, S., Li, X., Ghazvininejad, M., Weston, J., and Wang, T. Learning to plan & reason for evaluation with thinking-llm-as-a-judge. *arXiv preprint arXiv:2501.18099*, 2025.
- Seed, B., Zhang, Y., Su, J., Sun, Y., Xi, C., Xiao, X., Zheng, S., Zhang, A., Liu, K., Zan, D., et al. Seed-coder: Let the code model curate data for itself. *arXiv preprint arXiv:2506.03524*, 2025.
- Shao, R., Li, S. S., Xin, R., Geng, S., Wang, Y., Oh, S., Du, S. S., Lambert, N., Min, S., Krishna, R., et al. Spurious rewards: Rethinking training signals in rlvr. *arXiv preprint arXiv:2506.10947*, 2025a.
- Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y., Wu, Y., et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.
- Shao, Z., Luo, Y., Lu, C., Ren, Z., Hu, J., Ye, T., Gou, Z., Ma, S., and Zhang, X. Deepseekmath-v2: Towards self-verifiable mathematical reasoning. *arXiv preprint arXiv:2511.22570*, 2025b.
- Sheng, G., Zhang, C., Ye, Z., Wu, X., Zhang, W., Zhang, R., Peng, Y., Lin, H., and Wu, C. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv:2409.19256*, 2024.
- Sun, H., Shen, Y., and Ton, J.-F. Rethinking reward modeling in preference-based large language model alignment. In *The Thirteenth International Conference on Learning Representations*, 2025a.
- Sun, S., Yu, J., Wang, Z., Yang, X., Gu, T., and Yang, Y. S2j: Bridging the gap between solving and judging ability in generative reward models. *arXiv preprint arXiv:2509.22099*, 2025b.
- Tan, S., Zhuang, S., Montgomery, K., Tang, W. Y., Cuadron, A., Wang, C., Popa, R. A., and Stoica, I. Judgebench: A benchmark for evaluating llm-based judges. *arXiv preprint arXiv:2410.12784*, 2024.
- Team, K., Du, A., Gao, B., Xing, B., Jiang, C., Chen, C., Li, C., Xiao, C., Du, C., Liao, C., et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025.
- Wang, H., Xiong, W., Xie, T., Zhao, H., and Zhang, T. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. In *EMNLP*, 2024a.
- Wang, P., Xu, A., Zhou, Y., Xiong, C., and Joty, S. Direct judgement preference optimization. *arXiv preprint arXiv:2409.14664*, 2024b.
- Wang, Y., Li, H., Zhang, X., Wu, J., Liu, X., Hu, W., Guo, Z., Huang, Y., Xin, Y., Yang, Y., et al. Epicoder: Encompassing diversity and complexity in code generation. *arXiv preprint arXiv:2501.04694*, 2025a.
- Wang, Z., Dong, Y., Delalleau, O., Zeng, J., Shen, G., Egert, D., Zhang, J., Sreedhar, M. N., and Kuchaiev, O. Helpsteer 2: Open-source dataset for training top-performing reward models. *Advances in Neural Information Processing Systems*, 37:1474–1501, 2024c.
- Wang, Z., Gu, T., Gong, C., Tian, X., Bao, S., and Yang, Y. Scan: Structured capability assessment and navigation for llms, 2025b. URL <https://arxiv.org/abs/2505.06698>.
- Wang, Z., Zeng, J., Delalleau, O., Evans, E., Egert, D., Shin, H.-C., Soares, F., Dong, Y., and Kuchaiev, O. Rlbff: Binary flexible feedback to bridge between human feedback & verifiable rewards. *arXiv preprint arXiv:2509.21319*, 2025c.
- Wang, Z., Zeng, J., Delalleau, O., Shin, H.-C., Soares, F., Bukharin, A., Evans, E., Dong, Y., and Kuchaiev, O. Helpsteer3-preference: Open human-annotated preference data across diverse tasks and languages, 2025d. URL <https://arxiv.org/abs/2505.11475>.
- Wu, T., Yuan, W., Golovneva, O., Xu, J., Tian, Y., Jiao, J., Weston, J., and Sukhbaatar, S. Meta-rewarding language models: Self-improving alignment with llm-as-a-meta-judge. *arXiv preprint arXiv:2407.19594*, 2024.

- Wu, T., Zou, J., Liang, J., Zhang, L., and Ma, K. Visualquality-rl: Reasoning-induced image quality assessment via reinforcement learning to rank. *arXiv preprint arXiv:2505.14460*, 2025.
- Xie, T., Gao, Z., Ren, Q., Luo, H., Hong, Y., Dai, B., Zhou, J., Qiu, K., Wu, Z., and Luo, C. Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning. *arXiv preprint arXiv:2502.14768*, 2025.
- Xu, A., Zhou, Y., Nguyen, X.-P., Xiong, C., and Joty, S. J4r: Learning to judge with equivalent initial state group relative policy optimization. *arXiv preprint arXiv:2505.13346*, 2025a.
- Xu, Z., Jiang, F., Niu, L., Deng, Y., Poovendran, R., Choi, Y., and Lin, B. Y. Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing. In *The Thirteenth International Conference on Learning Representations*, 2025b.
- Yang, H., Zhou, Y., Han, W., and Shen, J. Self-rewarding large vision-language models for optimizing prompts in text-to-image generation. *arXiv preprint arXiv:2505.16763*, 2025a.
- Yang, W., Chen, J., Lin, Y., and Wen, J.-R. Deepcritic: Deliberate critique with large language models. *arXiv preprint arXiv:2505.00662*, 2025b.
- Ye, Z., Greenlee-Scott, F., Bartolo, M., Blunsom, P., Campos, J. A., and Gall  , M. Improving reward models with synthetic critiques. *arXiv preprint arXiv:2405.20850*, 2024a.
- Ye, Z., Li, X., Li, Q., Ai, Q., Zhou, Y., Shen, W., Yan, D., and Liu, Y. Beyond scalar reward model: Learning generative judge from preference data. *arXiv preprint arXiv:2410.03742*, 2024b.
- Ying, S., Li, Y., Qu, X., Li, X., Jin, S., Liu, M., Wen, Z., Du, X., Zheng, T., Zhang, Y., et al. Beyond correctness: Evaluating subjective writing preferences across cultures. *arXiv preprint arXiv:2510.14616*, 2025.
- Yu, H., Qi, Z., Zhao, Y., Nottingham, K., Xuan, K., Majumder, B. P., Zhu, H., Liang, P. P., and You, J. Sotopia-rl: Reward design for social intelligence. *arXiv preprint arXiv:2508.03905*, 2025a.
- Yu, Y., Chen, Z., Zhang, A., Tan, L., Zhu, C., Pang, R. Y., Qian, Y., Wang, X., Gururangan, S., Zhang, C., et al. Self-generated critiques boost reward modeling for language models. *arXiv preprint arXiv:2411.16646*, 2024.
- Yu, Y., Zhang, Y., Zhang, D., Liang, X., Zhang, H., Zhang, X., Yang, Z., Khademi, M., Awadalla, H., Wang, J., et al. Chain-of-reasoning: Towards unified mathematical reasoning in large language models via a multi-paradigm perspective. *arXiv preprint arXiv:2501.11110*, 2025b.
- Yu, Z., Zeng, J., Gu, W., Wang, Y., Wang, J., Meng, F., Zhou, J., Zhang, Y., Zhang, S., and Ye, W. Rewardanything: Generalizable principle-following reward models. *arXiv preprint arXiv:2506.03637*, 2025c.
- Yuan, W., Pang, R. Y., Cho, K., Li, X., Sukhbaatar, S., Xu, J., and Weston, J. E. Self-rewarding language models. In *Forty-first International Conference on Machine Learning*, 2024.
- Zhang, B., Ma, R., Jiang, Q., Wang, P., Chen, J., Xie, Z., Chen, X., Wang, Y., Ye, F., Li, J., et al. Sentient agent as a judge: Evaluating higher-order social cognition in large language models. *arXiv preprint arXiv:2505.02847*, 2025a.
- Zhang, L., Hosseini, A., Bansal, H., Kazemi, M., Kumar, A., and Agarwal, R. Generative verifiers: Reward modeling as next-token prediction. *arXiv preprint arXiv:2408.15240*, 2024.
- Zhang, N., Sun, R., Su, R., Ma, S., Zhang, S., Weng, X., Zhang, X., Zhan, Y., Xu, Y., Chen, Z., et al. Echo-nl: Affective rl frontier. *arXiv preprint arXiv:2512.00344*, 2025b.
- Zhang, Q., Chen, B., Zhang, F., Ding, R., Wang, S., Wang, Q., Huang, Y., Zhang, H., Zhu, R., Wang, P., et al. Arenarl: Scaling rl for open-ended agents via tournament-based relative ranking. *arXiv preprint arXiv:2601.06487*, 2026.
- Zhang, X., Zhang, X., Wu, Y., Cao, Y., Zhang, R., Chu, R., Yang, L., and Yang, Y. Generative universal verifier as multimodal meta-reasoner. *arXiv preprint arXiv:2510.13804*, 2025c.
- Zhao, J., Liu, R., Zhang, K., Zhou, Z., Gao, J., Li, D., Lyu, J., Qian, Z., Qi, B., Li, X., et al. Genprm: Scaling test-time compute of process reward models via generative reasoning. *arXiv preprint arXiv:2504.00891*, 2025.
- Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E., et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36: 46595–46623, 2023.
- Zhou, E., Zheng, G., Wang, B., Xi, Z., Dou, S., Bao, R., Shen, W., Xiong, L., Fan, J., Mou, Y., et al. Rmb: Comprehensively benchmarking reward models in llm alignment. *arXiv preprint arXiv:2410.09893*, 2024.

Zhou, J., Chen, Z., Wang, S., Dai, Q., Dong, Z., Wang, H., and Huang, M. Think socially via cognitive reasoning. *arXiv preprint arXiv:2509.22546*, 2025a.

Zhou, J., Ji, J., Dai, J., and Yang, Y. Sequence to sequence reward modeling: Improving rlhf by language feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 27765–27773, 2025b.

Zhou, L., Zhang, J., Gao, J., Jiang, M., and Wang, D. PersonaEval: Are LLM evaluators human enough to judge role-play? *arXiv preprint arXiv:2508.10014*, 2025c.

Zhou, P., Peng, X., Song, J., Li, C., Xu, Z., Yang, Y., Guo, Z., Zhang, H., Lin, Y., He, Y., et al. Opening: A comprehensive benchmark for judging open-ended interleaved image-text generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 56–66, 2025d.

A. Understanding Reward Modeling from Theoretical Perspective: From Statistical Correlation to Causal Alignment

A.1. Notation and Idealized Reward Modeling

We consider the classical pairwise preference reward task: given an input prompt x , the model generates two candidate responses $y_A, y_B \in \mathcal{Y}$, and human annotators provide a preference label $l \in \{A, B\}$.

For any triplet (x, y_A, y_B) , we assume there exists an ideal **evaluation dimension vector** $\mathbf{z} = [z_1, z_2, \dots, z_K]^\top$ and **weight vector** $\mathbf{w} = [w_1, w_2, \dots, w_K]^\top$, where each dimension z_k characterizes an interpretable quality attribute (e.g., factual accuracy, logical coherence, linguistic politeness, response length, etc). Weight $w_k \geq 0$ captures the importance of that dimension in the current comparison scenario. The combination of dimensions and weights constitutes what we refer to as the reward causality. For generative reward models, reward causality is sometimes contained in the critique rather than direct numerical computation.

Note: Both the dimension and weight vector $\mathbf{w}$ are context-dependent—only after given a specific (x, y_A, y_B) can we infer which dimensions actually participate in human judgment and their relative importance. Ideally, the weights of key dimensions should approach 1, while irrelevant dimensions should have weights approaching 0. For example: in knowledge-intensive QA tasks, the weight of "factual accuracy" far exceeds "linguistic politeness"—a factually incorrect response, no matter how politely expressed, typically cannot prevail over a factually accurate but plain-toned response.

A.2. Summary of Reward Modeling Mechanisms

Since the emergence of traditional BT reward modeling, numerous improvement schemes have emerged to address the limitations of reward modeling. Their common approach is to enhance model performance through causal alignment.

This article formalizes "causality" as a combined representation of evaluation dimensions and weights, and systematically reviews the following schemes shown in Table 7.

Table 7. Summary of Different Reward Modeling Mechanisms

Method	Modeling Approach	Modeling Formula	Learning Objective
Bradley-Terry Reward Modeling	Compress high-dimensional evaluation space into a single scalar score	$P_\phi(y_A \succ y_B x) = \sigma(r_\phi(x, y_A) - r_\phi(x, y_B))$	Learn scalar utility function through minimizing negative log-likelihood loss, such that $r_\phi(x, y) \approx \mathbf{w}^\top \mathbf{z}$
Conditional BT Reward Modeling	Explicitly model causality as part of the input	$P_\phi(y_A \succ y_B x, \mathbf{d}, \mathbf{w}) = \sigma(\mathbf{w}^\top r_\phi(x, y_A \mathbf{d}) - \mathbf{w}^\top r_\phi(x, y_B \mathbf{d}))$	Learn decomposed rewards conditioned on dimensions, based on predefined dimension description vector $\mathbf{d}$ and weight vector $\mathbf{w}$
Generative Reward Modeling with Outcome Supervision	Explicitly model causality as part of the output	$P_\theta(c, l x, y_A, y_B) = P_\theta(c x, y_A, y_B) \cdot \underbrace{P_\theta(l x, y_A, y_B, c)}_{\text{supervised}}$	Generate critique c and preference label l , using only preference label correctness as supervision signal, expecting causal capability through result correctness constraint
Generative Reward Modeling with Process Supervision	Explicitly supervise the causality in the output	$P_\theta(c, l x, y_A, y_B) = \underbrace{P_\theta(c x, y_A, y_B)}_{\text{supervised}} \cdot \underbrace{P_\theta(l x, y_A, y_B, c)}_{\text{supervised}}$	Directly optimize causal alignment for explicit critique supervision

A.3. Bradley-Terry Reward Modeling

Modeling Mechanism: Traditional reward models assume a scalar utility function $r_\phi(x, y)$. The core idea is to compress the high-dimensional evaluation space $\mathbf{z}$ into a single scalar score. The preference probability follows the Bradley-Terry (BT) model:

$$P_\phi(y_A \succ y_B | x) = \sigma(r_\phi(x, y_A) - r_\phi(x, y_B)) \quad (3)$$

The model is trained by minimizing the negative log-likelihood loss $\mathcal{L}_{\text{BT}}$. Ideally, the scalar reward should approximate the inner product of dimension features and true weights:

$$r_{\phi}(x, y) \approx \mathbf{w}^{\top} \mathbf{z} \quad (4)$$

Failure Modes: The BT model is fundamentally a statistical learning approach that estimates preference patterns from empirical data distributions. While effective at the dataset level, this statistical nature introduces two primary sources of bias that compromise its discriminative power on individual samples:

1. Sample-Level Causal Ambiguity: The BT model suffers from causal sparsity of supervision: for a sample (x, y_A, y_B, l) , the training objective only models the preference label l without capturing which dimensions determined the preference and their relative importance (Wang et al., 2024a; Minghao Yang, 2024). To compensate, the model relies on statistical aggregation extracted from patterns of similar samples in the training data. Learning from aggregated statistics rather than sample-specific criteria leads to inaccurate causal estimation at the individual sample level.

2. Dataset-Level Causal Ambiguity: Beyond similar samples, global dataset patterns also introduce bias (Liu et al., 2024b). Since the BT model uses global empirical risk minimization for optimization, if a large proportion of "better" responses in the dataset share features like "high detail" or "polite tone," the model may increase their weights globally, even when irrelevant to specific samples.

3. Lack of Reasoning Capability: Without reasoning capability, the BT model tends to capture surface-level correlations with label l , failing to model dimensions requiring deep reasoning (Chen et al., 2025; Guo et al., 2025c; Liu et al., 2025). For example, in math tasks, it struggles to learn "answer correctness" (which needs symbolic verification) and instead relies on proxy features like "solution detail."

In summary, the traditional BT model introduces causal ambiguity (or spurious correlations) for individual samples and lacks reasoning capability. It is only suitable for dimensions that are globally important and surface-observable (e.g., style, politeness, detail).

A.4. Conditional BT Reward Modeling

Modeling Mechanism: To alleviate the causal ambiguity of BT reward modeling, recent work (Wang et al., 2025c) proposes explicitly modeling evaluation dimensions into the reward model. Although existing schemes mostly focus on single-dimension scenarios (implicitly assuming the weight of this dimension is 1), they can naturally extend to more general settings:

$$P_{\phi}(y_A \succ y_B | x, \mathbf{d}, \mathbf{w}) = \sigma(\mathbf{w}^{\top} r_{\phi}(x, y_A | \mathbf{d}) - \mathbf{w}^{\top} r_{\phi}(x, y_B | \mathbf{d})) \quad (5)$$

where $\mathbf{d} = [d_1, d_2, \dots, d_K]$ is a predefined dimension description vector, $r_{\phi}(x, y | \mathbf{d}) \in \mathbb{R}^K$ is the decomposed reward conditioned on dimensions, and $\mathbf{w}$ is a manually specified weight vector.

Failure Modes: This mechanism directly addresses causal ambiguity by pre-specifying causal auxiliary evaluation. Several works (Wang et al., 2025c; Yu et al., 2025c; Wang et al., 2025b) leverage LLM’s reasoning capability for conditional generative reward modeling. Note that while these works address the lack of reasoning capability in BT models, all conditional reward modeling mechanisms face a significant limitation.

Specifically, since the dimension set $\mathbf{d}$ must be predefined, this method is only effective when key dimensions are highly determined. When facing (1) open-domain tasks with high openness such as open-domain dialogue and role-playing, and (2) tasks requiring complex and diverse process rewards such as multi-turn task planning and long-horizon reasoning, evaluation dimensions and weights are typically difficult to pre-specify, or even if pre-specified, are difficult to maintain consistency across all samples. Conditional reward modeling fails in these scenarios.

In summary, pre-specifying evaluation dimensions and weights endows BT models with the ability to incorporate causal reasoning for reward assignment. Compared to traditional BT models, they are more resistant to spurious correlations, but their applicability is narrow due to the need to pre-specify dimensions and weights. They are more suitable for tasks with clear rules, such as "syntax correctness" and "query efficiency" in SQL generation, or "citation reliability" and "argumentation completeness" in fact-checking.

A.5. Generative Reward Modeling with Outcome Supervision

Modeling Mechanism: Generative reward modeling changes the paradigm of reward modeling. Unlike conditional reward modeling which models causality in the input part, it models causality in the output part. Given input (x, y_A, y_B) , the model is required to generate a text sequence containing "critique" and "outcome". Some methods require the model to explicitly output evaluation dimensions and weights in the critique before giving the outcome (Liu et al., 2025), while others implicitly include this process in the critique without explicitly outputting related content (Wang et al., 2025d; Chen et al., 2025; Guo et al., 2025c; Huang et al., 2025; Xu et al., 2025a).

$$P_\theta(c, l|x, y_A, y_B) = P_\theta(c|x, y_A, y_B) \cdot \underbrace{P_\theta(l|x, y_A, y_B, c)}_{\text{supervised}} \quad (6)$$

where c is a natural language analysis of the superiority of the two responses (i.e., critique), which explicitly points out the evaluation dimensions $\mathbf{z}$ and local weights $\mathbf{w}$ in the semantic space. $l \in \{A, B\}$ is the final preference label derived from the critique. Training typically uses correctness of the preference label as a supervision signal to optimize the probability of the model predicting the correct label l .

Failure Modes: Compared to conditional reward modeling, GRM has the following advantages: **(1) Automatic Causal Discovery:** By generating critique c , the model is forced to explicitly identify key dimensions of the current sample (e.g., code logic is more important than comments for this problem), achieving identification of key dimensions and dynamic weight allocation. Additionally, it can use test-time scaling to find better dimensions and weights. **(2) Resistance to Global Ambiguity:** Since GRM requires explicit causal analysis and learning for each sample, it is harder to be affected by global statistical biases.

Despite the great potential of GRM, our work identifies a critical misconception in existing approaches: they actually expect the model to learn good causal capability through outcome correctness. However, we find this introduces significant noise to training as discussed above.

A.6. Generative Reward Modeling with Process Supervision

To directly align reward modeling with causality, we introduce process supervision, i.e., supervision of critique. Our paradigm requires GRM to optimize the correctness of critiques. The modeling approach remains unchanged, but the supervision signal changes:

$$P_\theta(c, l|x, y_A, y_B) = \underbrace{P_\theta(c|x, y_A, y_B)}_{\text{supervised}} \cdot \underbrace{P_\theta(l|x, y_A, y_B, c)}_{\text{supervised}} \quad (7)$$

Important evidence supporting this argument comes from the difference in critique word frequency distributions of models trained with the two supervision methods (see Section 4.7 and Figure 9 for details).

A.7. Theory-Practice Gap

Although our theoretical framework indicates that we should conduct reward modeling by supervising critique, there are several challenges in practice:

Cost of High-Quality Human Critiques: Currently, our approach to critique supervision relies on high-quality human critiques; however, collecting such data is both difficult and resource-intensive.

Difficulty in Obtaining Accurate Supervision: We use the similarity between GRM-generated and human critiques as the supervision metric, achieving 85.71% accuracy. This imperfect signal leads to: (1) suboptimal reward model performance, and (2) training instability. Fortunately, we have addressed this issue by introducing outcome regularization, which enables stable long-term training and ultimately achieves optimal performance.

Despite these theory-practice gaps, we believe our work points to a promising direction for future reward modeling, i.e., through substantial engineering efforts to obtain higher-quality human critiques, thereby obtaining high-quality critique supervision signals and ultimately more accurate reward models. Additionally, using human critique supervision may not be necessary, but this requires further research.

B. Discussion about Future Work

Although our method is validated on GRMs, it fundamentally addresses the challenge of obtaining process-level reward. We believe our approach can be extended to two promising domains.

Multiple-Choice and True/False Questions. First, tasks such as multiple-choice or true/false questions naturally suffer from noisy outcome reward due to their restricted solution space. Our method could provide more informative process-level supervision in these settings.

Open-Ended Tasks with Verifiable Correctness. Second, our approach may benefit open-ended tasks that lack explicit outcome reward but have verifiable correctness criteria, such as mathematical proofs. In these cases, correctness can be determined by comparing against ground truth solutions, providing process-level supervision signals.

C. Discussion about Limitations

We discuss the limitations of our approach here.

Limitations on Fully Open-Ended Tasks. We note that our method may not directly transfer to fully open-ended tasks without any ground truth, as these lack the human critiques or verifiable signals that our framework relies upon. Extending to such scenarios remains an important direction for future research.

Dependency on High-Quality Human Critiques. Our method requires high-quality human critiques. Although MetaRM enables semi-supervised learning and knowledge transfer, the approach may face challenges when dealing with diverse preferences across different populations, such as regional or cultural variations. In such cases, critiques from HelpSteer3 may not transfer effectively to these datasets. In this work, we focus on general-purpose scenarios where HelpSteer3 critiques are broadly applicable.

API Cost from External Models. We need to invoke external models to compute similarity between critiques, which introduces additional costs. This is primarily due to the limited capabilities of our used 7B/32B models. For larger models with stronger capabilities, similarity computation could be performed internally, eliminating this overhead. Such a self-verification approach (such as [Shao et al. \(2025b\)](#)) represents a promising direction for future work.

D. Discussion about Reward Hacking

During our experiments, we observed multiple forms of reward hacking. We document these engineering insights here as a discussion of the reward hacking phenomenon.

Reward Hacking for Recall-based Similarity. Initially, we attempted to use recall to measure the similarity between GRM-generated critiques and human critiques. However, we found that the GRM learned to generate an excessive number of critiques to maximize recall. Although it increased the training reward, it led to degraded performance on benchmarks.

Reward Hacking for Process Reward. We experimented with F1-based similarity and MetaRM and training for an extended number of steps. However, in later stages of training, the model exhibited abnormal behavior: response lengths became excessively long, process reward increased unreasonably, and outcome reward declined. Specifically, the GRM learned to exploit the system by repeatedly generating identical critiques. Therefore, in the prompt for computing similarity between critiques, we instructed it to first check whether the model repeats the same arguments multiple times.

E. Related Work on Generative Reward Models

Generative reward models (GRMs) have emerged as a powerful alternative to traditional scalar reward models ([Ouyang et al., 2022](#); [Sun et al., 2025a](#); [Zhang et al., 2024](#)), offering greater interpretability and accuracy by generating natural language critiques and reasoning chains. The LLM-as-a-Judge paradigm ([Zheng et al., 2023](#); [Dubois et al., 2024](#); [Saha et al., 2023](#)) pioneered this approach, producing explicit rationales ([Kim et al., 2024](#); [Ankner et al., 2024](#); [Yu et al., 2024](#); [Saha et al., 2025](#)) that enhance both transparency and evaluation accuracy. Training methods have evolved from prompt engineering to supervised fine-tuning (SFT) and direct preference optimization (DPO) ([Ye et al., 2024b](#); [Wang et al., 2024b](#); [Mahan et al., 2024](#); [Ye et al., 2024a](#); [Wu et al., 2024](#); [Zhao et al., 2025](#); [Anugraha et al., 2025](#)), and more recently to reinforcement learning with verifiable reward (RLVR) ([Chen et al., 2025](#); [Guo et al., 2025c](#); [Yang et al., 2025b](#); [Xu et al., 2025a](#); [Yu et al., 2025c](#)), inspired by advances in reasoning models ([Guo et al., 2025a](#); [Team et al., 2025](#); [Hu et al., 2025](#); [Xie et al., 2025](#)).

F. Formulation of Pairwise Rewarding Task and GRM Training with Outcome Reward

In RL with outcome reward under the GRPO framework, the model receives a binary reward $R_{\text{outcome}} \in \{0, 1\}$ based on whether its predicted preference label matches the ground truth. Training proceeds by generating N_{rollout} candidate responses for the same query, computing the average reward $\bar{R}$ and standard deviation σ within the group, and then normalizing reward to obtain advantages:

$$\bar{R} = \frac{1}{N_{\text{rollout}}} \sum_{i=1}^{N_{\text{rollout}}} R_{\text{outcome}_i}, \quad \sigma = \frac{1}{N_{\text{rollout}}} \sqrt{\sum_{i=1}^{N_{\text{rollout}}} (R_{\text{outcome}_i} - \bar{R})^2}, \quad (8)$$

$$\hat{A}_i = \frac{R_{\text{outcome}_i} - \bar{R}}{\sigma}. \quad (9)$$

The advantage $\hat{A}_i$ is assigned uniformly to all tokens in a response, and the policy gradient update is computed using clipping and KL regularization to stabilize optimization:

$$\mathcal{J}(\theta) = \mathbb{E} \left[\frac{1}{G} \sum_{i=1}^G \frac{1}{|\hat{y}_i|} \sum_{t=1}^{|\hat{y}_i|} \left(\min \left(r_{i,t} \hat{A}_i, \text{clip} \left(r_{i,t}, 1 - \varepsilon, 1 + \varepsilon \right) \hat{A}_i \right) - \beta D_{\text{KL}} \right) \right]. \quad (10)$$

G. MetaRM

G.1. MetaRM Architecture and Training

Architecture. The MetaRM M_ϕ is designed as a regression model that learns to predict reward combining both outcome and process information. Given a query q , two candidate responses y_A and y_B , and a GRM-generated critique $\hat{c}$, the MetaRM outputs a scalar reward score:

$$\hat{R}_{\text{meta}} = M_\phi(q, y_A, y_B, \hat{c}) \in [0, 1 + \lambda], \quad (11)$$

where λ is the weight of process reward defined in Equation 2.

Training Objective. For $\mathcal{D}_H$ containing human critiques h , the target reward R_{target} is computed as R defined in Equation 2. The MetaRM is optimized using mean squared error (MSE):

$$\mathcal{L}_{\text{MetaRM}}(\phi) = \mathbb{E}_{(q, y_A, y_B, \hat{c}, R_{\text{target}})} \left[(M_\phi(q, y_A, y_B, \hat{c}) - R_{\text{target}})^2 \right]. \quad (12)$$

Inference. Once trained, the MetaRM can predict process reward for samples without human critiques. For a new sample (q, y_A, y_B) with GRM-generated critique $\hat{c}$, we compute:

$$R' = \begin{cases} -1, & \text{if output format is invalid,} \\ 0, & \text{if } \hat{l} \neq l, \\ 1 + \max(0, \hat{R}_{\text{meta}} - 1), & \text{if } \hat{l} = l, \end{cases} \quad (13)$$

This design aligns with Equation 2 and ensures that correct outcomes receive base reward 1, augmented by the process component predicted by MetaRM, clipped to $[0, \lambda]$. For MetaRM, we also adopt **outcome regularization**, i.e., no process reward if outcome is incorrect.

Cold-start Initialization. To initialize the MetaRM, we sample N_{sample} responses from the base model for each sample in $\mathcal{D}_H$. We then compute the reward for these samples according to Equation 2 and optimize the model using the objective function in Equation 12 to obtain the initial MetaRM $M_{\phi_{\text{cold}}}$.

G.2. Online Training Algorithm

The online MetaRM framework alternates between updating the policy GRM π_θ and the MetaRM M_ϕ , ensuring that the reward model remains aligned with the evolving generation distribution. Algorithm 1 presents the complete training procedure.

Algorithm 1 Online MetaRM Training for GRM

Require: Dataset with human critiques $\mathcal{D}_H = \{(q, y_A, y_B, l, h)\}$, dataset without critiques $\mathcal{D}_O = \{(q, y_A, y_B, l)\}$, initial GRM π_{θ_0} , initial MetaRM M_{ϕ_0} , process reward weight λ , rollout budget N_{rollout} , pre-training epochs E_{pre}

```

1: // Cold start: Initialize MetaRM
2: for epoch  $e = 1, 2, \dots, E_{\text{pre}}$  do
3:   for each sample  $(q, y_A, y_B, h) \in \mathcal{D}_H$  do
4:     Sample  $N_{\text{rollout}}$  responses  $\{\hat{y}_i\}_{i=1}^{N_{\text{rollout}}}$  from  $\pi_{\theta_0}(q)$ 
5:     Evaluate critique quality:  $R_{\text{process}} = \mathbb{I}[S(h, \hat{c}) > 0.5]$ 
6:     Compute target reward  $R_{\text{target}}$  using Equation 2
7:   end for
8:   Update MetaRM  $M_{\phi_{\text{cold}}}$  using MSE loss (Equation 12)
9: end for
10:
11: // GRPO with online MetaRM
12: for step  $t = 1, 2, \dots, T$  do
13:   // Step 1: Generate rollouts from current policy
14:   for each query  $q$  in minibatch do
15:     Sample  $N_{\text{rollout}}$  responses  $\{\hat{y}_i\}_{i=1}^{N_{\text{rollout}}}$  from  $\pi_{\theta}(q)$ 
16:   end for
17:   // Step 2: Compute reward for  $\mathcal{D}_H$  using human critiques
18:   for each sample  $(q, y_A, y_B, h) \in \mathcal{D}_H$  do
19:     Evaluate critique quality:  $R_{\text{process}} = \mathbb{I}[S(h, \hat{c}) > 0.5]$ 
20:     Compute reward  $R$  using Equation 2
21:   end for
22:   // Step 3: Update MetaRM using  $\mathcal{D}_H$  and reward  $R$ 
23:   Update MetaRM using Equation 12
24:   // Step 4: Compute reward for  $\mathcal{D}_O$  using updated MetaRM
25:   for each sample  $(q, y_A, y_B) \in \mathcal{D}_O$  do
26:     Predict meta-reward  $\hat{R}_{\text{meta}}$  using Equation 11
27:     Assign reward  $R'$  for this sample using Equation 13
28:   end for
29:   // Step 5: Update policy model using reward from both datasets
30:   Compute advantages  $\hat{A}_i$  for all rollouts using reward  $\{R, R'\}$ 
31:   Update policy model (GRM) with GRPO loss using Equation 10
32: end for
    
```

Key Design Principles. The online training procedure incorporates several important design choices:

- **Dual Data Streams:** We maintain two data streams: $\mathcal{D}_H$ with human critiques for MetaRM supervision, and $\mathcal{D}_O$ without critiques for scalable policy learning.
- **MetaRM-First Update:** In each iteration, we first update the MetaRM using fresh samples from $\mathcal{D}_H$ (Step 3), then use the updated MetaRM to score samples in $\mathcal{D}_O$ (Step 4). This ensures the MetaRM is fully on-policy before providing reward.
- **Continuous Adaptation:** Unlike offline training where the MetaRM is frozen after initialization, online training continuously adapts the MetaRM to the evolving critique distribution as the policy improves.

H. Experiment Setups for Exploring Process Reward Design

For dataset construction, we randomly sample 49 data instances from HelpSteer3, each containing a query q , two responses y_a and y_b , GRM-generated critiques $\hat{c}$, and human critiques h . We manually annotate each critique $\hat{c}$ with a binary label $z \in \{0, 1\}$, where $z = 1$ indicates the critique is correct (i.e., accurately identifies issues in the response), and $z = 0$ otherwise. This results in a small but carefully curated evaluation set for studying process supervision methods.

We investigate three approaches to automatically predict the correctness label:

(1) LLM-as-a-Meta-Judge. We prompt an external LLM to directly evaluate whether the critique $\hat{c}$ correctly identifies issues in the response, given the query q and response y . The prompt is shown in Figure 16.

(2) Similarity w/ All HC. We use an external LLM to extract all critique points from both h and $\hat{c}$, then compute their similarity using three metrics: F1, Recall, and Precision. High similarity suggests the GRM critique aligns with human judgment. The extraction prompt is shown in Figure 14.

(3) Similarity w/ Core HC. Similar to (2), but we prompt the LLM to identify and extract only the *core* critical points from h and $\hat{c}$, filtering out minor issues. We then compute similarity between these core critiques. The extraction prompt is shown in Figure 14.

I. Experiment Setups for Preliminary Results

I.1. Experiment Setups for Section 3.1.

For base models, we select Deepseek-R1-Distill-Qwen-7B (Guo et al., 2025a) and Deepseek-R1-Distill-LLama-8B (Guo et al., 2025a). For training hyperparameters, we set the training steps to 150 (1 epoch), while all other parameters follow Table 10. We employ gpt-5-mini as the scoring model to evaluate the similarity (F1 score) between human critiques and scores. For training data, we use the complete HelpSteer3 dataset.

I.2. Experiment Setups for Section 3.2.

For base models, we select Deepseek-R1-Distill-Qwen-7B (Guo et al., 2025a) and Deepseek-R1-Distill-LLama-8B (Guo et al., 2025a). For training hyperparameters, we set the training steps to 150 (1 epoch), while all other parameters follow Table 10. We employ gpt-5-mini as the scoring model to evaluate the similarity (F1 score) between human critiques and GRM-generated critiques. For training data, we randomly split the HelpSteer3 dataset into two equal parts with a 1:1 ratio, serving as $\mathcal{D}_H$ and $\mathcal{D}_O$, respectively.

J. Experiment Setups for Main Results

J.1. Benchmarks

Following previous work (Shao et al., 2025a), data contamination has a significant impact on training effectiveness and evaluation credibility, especially for the Qwen2.5 series of models, where sometimes even spurious reward can take effect. In this paper, we mitigate this issue by selecting benchmarks that are released at most 1 month before the base model.

Table 8. Base model and benchmark release timeline, domains, label sources, and usage. Benchmarks released within 1 month before the base model are excluded to mitigate data contamination concerns.

Benchmark	Release	Domain	Outcome Source	Used
DeepSeek-R1-Distill-Qwen-7B (Guo et al., 2025a)	2025.01	–	–	–
DeepSeek-R1-Distill-Llama-8B (Guo et al., 2025a)	2025.01	–	–	–
DeepSeek-R1-Distill-Qwen-32B (Guo et al., 2025a)	2025.01	–	–	–
MT-Bench (Zheng et al., 2023)	2023.06	Writing, Roleplay, Extraction, Reasoning, Math, Coding, STEM, Humanities	Majority human voting	✗
RewardBench (Lambert et al., 2025)	2024.03	Chat, Safety, Math, Code	Verifiable correctness, dataset integration	✗
RM-Bench (Liu et al., 2024c)	2024.10	Chat, Safety, Code, Math	Verifiable correctness, deliberate errors	✗
PPE (Frick et al., 2025)	2024.10	Knowledge, Math, STEM, Coding, Instruction Following, Chat	Verifiable correctness, single human	✗
JudgeBench (Tan et al., 2024)	2024.10	Knowledge, Reasoning, Math, Coding	Verifiable correctness	✗
RMB (Zhou et al., 2024)	2024.10	Helpfulness, Harmlessness	gpt-4o	✗
HREF (Lyu et al., 2024)	2024.12	General instruction following	Majority human voting	✓
WQ-LMArena (Chakrabarty et al., 2025)	2025.04	General writing	Single human	✓
SCAN-HPD (Wang et al., 2025b)	2025.05	Writing, Roleplay, Knowledge, Coding, Math, Reasoning	Majority human voting	✓
HelpSteer3 (Wang et al., 2025d)	2025.05	General chat, STEM, Code, Multilingual	Majority human voting	✓
RewardBench V2 (Malik et al., 2025)	2025.06	Factuality, Instruction Following, Math, Safety, Focus	LLM-as-a-judge, verifiable correctness, human	✓
LitBench (Fein et al., 2025)	2025.07	Social media replies	Human upvote count	✓
WPB (Ying et al., 2025)	2025.10	Documents, Communication, Fiction, Poetry, Scriptwriting, Role-playing	Human-AI collaborative	✓

J.2. Training Datasets

In our experiments, we exclusively use pairwise preference data. The specific distribution of our training data is shown in Table 9.

Table 9. **Training Data Overview.** The asterisk (*) denotes data sourced from Skywork-RewardPreference-80K-v0.2 (Liu et al., 2024a).

Source	Size	Domain
<i>Process Reward: Human Critique</i>		
HelpSteer3 (Wang et al., 2025d)	36,255	General Domain
<i>Process Reward: MetaRM</i>		
Helpsteer2 (Wang et al., 2024c)*	7,221	General Domain
Magpile_Pro_Llama3.1 (Xu et al., 2025b)*	28,349	General Domain
Magpile_Pro (Xu et al., 2025b)*	2,030	General Domain
Offset_Bias (Park et al., 2024)*	8,504	General Domain
Code-Preference-Pairs	8,000	Code
Math-DPO-10K (Lai et al., 2024)	10,000	MATH
hamishivi/math_rlv_r_mixture_dpo	17,713	MATH
WebInstruct (Sun et al., 2025b)	5,910	Knowledge
WildGuard (Han et al., 2024)*	6,709	Safety
javirandor/hh-rlhf-safety-v3-dpo	9,371	Safety
Tulu-3-Pref-Personas-Instruction-Following (Lambert et al., 2024)	12,000	Instruction Following
LitBench-Train (Fein et al., 2025)	12,000	Social Media
Total	164,062	

J.3. Implementation Details

For our main experiments, we select Deepseek-R1-Distill-Qwen-7B and Deepseek-R1-Distill-Qwen-32B as the base models. To evaluate the F1-based similarity between generated critiques and human critiques, we employ the lightweight gpt-5-mini as the scoring model. For MetaRM, we utilize the same model as GRM but replace the final layer with a regression head that outputs a single scalar value.

For the cold start of MetaRM, we first sample 8 responses per prompt from the base model using HelpSteer3. We then train it for 3 epochs with a learning rate of $1e-5$, using a cosine scheduler, weight decay of 0.01, and a warmup ratio of 0.01.

We train our GRMs using the GRPO algorithm, implemented on the VeRL (Sheng et al., 2024) framework. The training is conducted on a setup with 32 NVIDIA A100 GPUs. Key hyperparameters are detailed in Table 10.

Table 10. Training hyperparameters.

Hyperparameter	Value
<i>Data Configuration</i>	
Max prompt length	6144
Max response length	8192
<i>GRPO Algorithm Configuration</i>	
Rollouts per prompt	8
Sampling temperature	0.7
Sampling top-p	1.0
KL loss	0.001
<i>GRM</i>	
Optimizer	AdamW
Learning rate	$1e-6$
Batch size	256
Mini-batch size	256
Total training steps	1200
<i>Online MetaRM during RL training</i>	
Loss	MSE loss
Learning rate	$5e-6$

J.4. Baselines

We select open-source specialized GRMs: RM-R1 and RRM, using their versions based on Deepseek-R1-Distill-Qwen-7B (Guo et al., 2025a) and Deepseek-R1-Distill-Qwen-32B (Guo et al., 2025a) respectively for fair comparison. For scalar reward model, we select URM-LLaMa-3.1-8B (Lou et al., 2024), Skywork-Reward-Llama-3.1-8B-v0.2 (Liu et al., 2024a), ArmoRM-Llama3-8B-v0.1 (Wang et al., 2024a) and INF-ORM-Llama3.1-70B (Minghao Yang, 2024). Additionally, we employ several base models as judge models through prompt engineering, including: gemini-2.5-pro, claude-3-7-sonnet-20250219, o3-2025-04-16, gpt-5-2025-08-07, gpt-4o-latest, deepseek-r1-0528-inner, deepseek-v3-inner, qwen-plus-latest, DeepSeek-R1-Distill-Qwen-7B, DeepSeek-R1-Distill-Llama-8B, and DeepSeek-R1-Distill-Qwen-32B.

K. More Results

K.1. Ablation Study on MetaRM

We conduct ablation studies on MetaRM to analyze the impact of different reward designs for the outcome dataset $\mathcal{D}_O$. Note that for the human critique dataset $\mathcal{D}_H$, all variants compute similarity with human critiques. We compare the following reward strategies for $\mathcal{D}_O$: (1) **No-MetaRM**: using only outcome-based reward; (2) **Offline-MetaRM**: reward from offline-trained MetaRM with continuous values; (3) **Online-MetaRM**: our default online MetaRM with continuous reward; (4) **Only-MetaRM**: using only MetaRM reward without any outcome signal; (5) **Binary**: reward from online MetaRM with outputs discretized to $\{0, 1\}$; (6) **Aggressive**: online MetaRM trained with higher learning rate ($1e-5$) for 2 epochs per round; (7) **Classifier**: online MetaRM using classification head instead of regression.

The ablation results reveal two critical findings. First, online training is essential for MetaRM’s effectiveness. Online-MetaRM consistently outperforms Offline-MetaRM across both base models (0.5552 vs 0.5493 for Qwen-7B; 0.6356 vs 0.6273 for Llama-8B), as online adaptation enables MetaRM to track the evolving policy distribution during RL training. The Aggressive variant, which updates MetaRM too rapidly, degrades performance (0.5407 for Qwen-7B), indicating that overly fast adaptation destabilizes training. Second, using MetaRM reward alone (Only-MetaRM) exhibits model-dependent instability. While Only-MetaRM achieves the best results for Qwen-7B (0.5579), it significantly underperforms for Llama-8B (0.6143 vs 0.6356), with a dramatic drop on RewardBench2 (0.3562 vs 0.5196). This inconsistency suggests that relying solely on MetaRM without grounding in outcome-based reward leads to unpredictable behavior across architectures. The default Online-MetaRM, which combines both signals, provides more robust performance. Additionally, Binary and Classifier variants both underperform, confirming that continuous regression-based reward are preferable for capturing nuanced quality distinctions.

Table 11. Ablation study on MetaR designs.

Method	HelpSteer3	RewardBench2	SCAN-HPD	HREF	LitBench	WQ-Arena	WPB	Overall
<i>DeepSeek-R1-Distill-Qwen-7B</i>								
No-MetaRM	0.6242	0.3375	0.6245	0.5603	0.5392	0.5329	0.5240	0.5347
Offline-MetaRM	0.6444	0.3432	0.6440	0.5628	0.5680	0.5310	0.5516	0.5493
Online-MetaRM	0.6529	0.3364	0.6712	0.5719	0.5738	0.5335	0.5464	0.5552
Only-MetaRM	0.6723	0.3358	0.6840	0.5554	0.5686	0.5309	0.5582	0.5579
Binary	0.6615	0.3307	0.6342	0.5661	0.5662	0.5359	0.5335	0.5469
Aggressive	0.6294	0.3358	0.6407	0.5504	0.5534	0.5289	0.5464	0.5407
Classifier	0.6382	0.3301	0.6384	0.5421	0.5647	0.5348	0.5315	0.5400
<i>DeepSeek-R1-Distill-Llama-8B</i>								
No-MetaRM	0.7096	0.5054	0.7396	0.5901	0.5919	0.5651	0.6156	0.6168
Offline-MetaRM	0.7252	0.4759	0.7220	0.6405	0.6310	0.5794	0.6170	0.6273
Online-MetaRM	0.7259	0.5196	0.7424	0.6364	0.6226	0.5846	0.6176	0.6356
Only-MetaRM	0.6970	0.3562	0.7109	0.6884	0.6214	0.5807	0.6456	0.6143

K.2. Analysis of Time Consumption

An important question is how much additional time overhead our method introduces. We provide a detailed breakdown of the time consumption for one RL training step in Table 12. Our scoring scheme can be executed either synchronously or asynchronously with the rollout process. In the synchronous setting (Ours(Sync)), our method increases the training time from 156s to 289s for the 7B model and from 655s to 874s for the 32B model per training step. However, with asynchronous execution (Ours(Async)), the overhead is significantly reduced: the total time becomes 196s for 7B (26% increase) and 766s for 32B (17% increase) compared to the baseline. Considering the significant improvement in final performance, the additional computation overhead (26% for 7B and 17% for 32B) introduced by our method is acceptable.

Overall, this linguistic analysis reveals a fundamental limitation of pure outcome-based reward models: they tend to generate superficial, templated and nitpicky feedback that fails to capture the true core issues. However, by incorporating explicit natural language feedback signals, we force the model to identify the key factors that truly distinguish response quality, thereby encouraging the model to produce critiques that are more precise and diverse.

Table 12. Training Time Breakdown for One Step (seconds). Processes with the same color can be executed asynchronously. We enable asynchronous execution for computing reward and computing log-probabilities by default. "Ours(Sync)" and "Ours(Async)" denote sequential and concurrent execution of Rollout and Rewarding w/ HC (Human Critique) & Outcome, respectively.

Method	Rollout	Rewarding w/ HC & Outcome	MetaRM (Update & Rewarding)	Compute LogProb & KL	Update Policy	Save Logs	Total
<i>7B</i>							
Outcome-only	95	2	-	21	36	4	156
Ours (Sync)	95	113	41	21	36	4	289
Ours (Async)	95	115	41	21	36	4	196
<i>32B</i>							
Outcome-only	376	2	-	91	184	4	655
Ours (Sync)	376	113	197	91	184	4	874
Ours (Async)	376	381	197	91	184	4	766

K.3. How Do Human Critiques Take Effect?

To demonstrate how human critiques improve critique quality, we present the proportion of cases where our model produces correct outcomes but flawed processes in Figure 11. As shown, compared to similar-sized models, our method reduces $P(\text{process} = 0 | \text{outcome} = 1)$ significantly, demonstrating the superiority of our approach in generating higher-quality critiques.

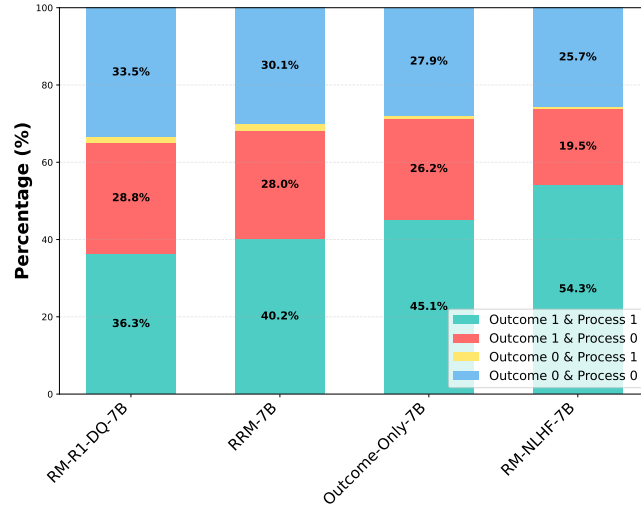


Figure 11. Effectiveness of process reward. $P(\text{process} = 0 | \text{outcome} = 1)$ is substantially lower than baselines.

K.4. Ablation Study: Direct LLM Scoring vs. Human Critiques

To further validate the efficacy of human critiques, we conduct an additional ablation study. Specifically, we evaluate a baseline where gpt-5-mini directly scores critique quality. This experimental setup aligns with the configuration in Figure 5, utilizing a training duration of 1 epoch (150 steps).

The results, summarized in Table 13, demonstrate that incorporating human-annotated critiques yields superior performance compared to relying solely on an LLM-as-a-meta-judge. The advantage is most pronounced on the HS3 (HelpSteer3) dataset, which explicitly provides human-annotated critiques. Here, our approach achieves a performance gain of up to 3.25% over the direct LLM scoring baseline. This significant margin underscores the critical role of human feedback in refining critique quality and enhancing overall model alignment.

Table 13. Ablation results evaluated by gpt-5-mini.

	HS3	RB-V2	SCAN	HREF	LB	WQA	WPB	AVG
Similarity w/ Core HC (F1)	0.6723	0.3358	0.6840	0.5554	0.5686	0.5309	0.5582	0.5579
LLM-as-a-Meta-Judge	0.6398	0.3373	0.6512	0.5545	0.5534	0.5255	0.5389	0.5429

K.5. Comparison with CLOUD

CLOUD (ANKNER ET AL., 2024) estimates rewards by generating critiques and passing them to a scalar head. As shown in Table 14, our 32B model surpasses the 70B CLOUD-RM baseline significantly. This highlights the effectiveness of our framework, achieving state-of-the-art results with substantially fewer parameters (32B vs. 70B).

Table 14. Comparison with CLOUD-RM (70B). Our 32B model achieves better performance with higher efficiency.

	HS3	RB-V2	SCAN	HREF	LB	WQA	WPB	AVG
CLOUD-RM-70B	0.7368	0.6123	0.7891	0.6347	0.7181	0.5881	0.6456	0.6750
RM-NLHF-32B	0.8315	0.7867	0.7888	0.7165	0.7492	0.6161	0.6183	0.7296

K.6. Expanded Validation for design of Meta-Judge

To further support the conclusions for design of Meta-Judge, we expand the validation set and re-verify the effectiveness of our Meta-Judge design.

Expanded Validation Set. We expand the validation set to 289 samples. The samples are drawn from HelpSteer3, a diverse dataset spanning rich domains. The critiques are sampled in a balanced manner from 6 LLMs: DeepSeek-R1, Gemini2.5-Pro, Qwen3-Max, Qwen2.5-7B-Instruct, GPT-5, and Claude3.7. We recruit 5 annotators with a Master’s degree in relevant fields. Each sample is independently labeled by 2 annotators, and samples with disagreement are discarded to ensure inter-annotator agreement. The remaining data undergoes an independent check by 1 annotator. The overview of annotation principles is: the critique is correct and no key critiques are missing. We will include the complete annotation principles in the final revision, and open-source this annotated dataset upon acceptance.

Re-verification Results. The results on the expanded validation set, shown in Table 15, demonstrate conclusions consistent with those reported previously.

Table 15. Results on the expanded validation set of 289 samples.

Method	ACC
LLM-as-a-Meta-Judge	0.4360
Similarity w/ All HC (F1)	0.7093
Similarity w/ All HC (Recall)	0.7439
Similarity w/ All HC (Precision)	0.6194
Similarity w/ Core HC (F1)	0.7751
Similarity w/ Core HC (Recall)	0.7785
Similarity w/ Core HC (Precision)	0.7439

Threshold Ablation. We further conduct an ablation study over the threshold, which justifies our choice of 0.5 previously.

K.7. Sensitivity to the LLM used for argument extraction and similarity scoring

We evaluate five backbone LLMs for scoring, reporting accuracy (ACC) on 289 validation samples and agreement rate with Qwen3-32B in Tab 17. The results show high consistency both within and across model families, indicating that our method is robust to the choice of LLM backbone for scoring.

Table 16. Ablation study over the similarity threshold.

Threshold	0.2	0.3	0.4	0.5	0.6	0.7	0.8
ACC	0.7716	0.7716	0.7716	0.7751	0.7751	0.7024	0.6920

Table 17. ACC and agreement rate of different scorer models.

Model	ACC	Agreement Rate
Qwen3-8B	0.6609	0.7057
Qwen3-32B	0.7785	1.0000
Qwen3-235B	0.8028	0.7962
GPT-5-mini	0.7751	0.7698
DeepSeek-R1	0.7324	0.8113

We conduct an ablation study on model scale. As shown in Tab. 18, Qwen3-8B underperforms noticeably, while models of 32B and above show negligible differences, suggesting that a 32B model is sufficient. This indicates that our approach does not rely on a strong scoring model—moderate capability suffices.

Table 18. Ablation study on different scoring models.

Model	HS3	RB-V2	Scan	Ref	PB	WQA	WPB	Avg
Qwen3-8B	0.6309	0.2973	0.6581	0.5493	0.5696	0.5166	0.5478	0.5385
Qwen3-32B	0.6445	0.3409	0.6641	0.5766	0.5657	0.5657	0.5570	0.5592
Qwen3-235B	0.6712	0.3501	0.6778	0.5623	0.5601	0.5443	0.5589	0.5607

L. Prompt

Prompt: Template of Generative Reward Models

Please act as an impartial judge and evaluate the quality of the responses provided by two AI Chatbots to the Client question displayed below.

[Client Question]

{conv_hist}

[The Start of Chatbot A's Response]

{response_A}

[The End of Chatbot A's Response]

[The Start of Chatbot B's Response]

{response_B}

[The End of Chatbot B's Response]

Output your final verdict by strictly following this format:

<critics>

[Provide a brief summary of your reasoning for the choice]

</critics>

<choice>

[[A]]

</choice>

Note: Use [[A]] if A is better, or [[B]] if B is better.

Figure 12. Prompt: Template of Generative Reward Models.

Prompt: Edit-only Refinement Instruction based on Critiques

[SYSTEM RULE: EDIT-ONLY MODE ENGAGED]

You are a text-processing bot. You are FORBIDDEN from answering the question. Your only function is to apply edits.

PRIMARY DIRECTIVE: Follow the <critique>. Nothing else.

1. **THE EXCEPTION RULE:** IF the <critique> states something is factually or mathematically wrong, you are authorized to fix **ONLY THAT SINGLE PIECE OF INFORMATION**. Do not explain. Do not add context. Just replace the wrong data with the correct data.
2. **THE DEFAULT RULE:** For everything else, if the <critique> does not explicitly order a change, you **MUST NOT** change it. Do not fix other errors. Do not improve style. Do not add information.
3. **THE RE-CHECK RULE:** Before you respond, you must check again whether the <critique> contain the content you modify.

Failure to follow these rules means you fail the task.

[Client Question]

{conv_his}

[The Start of Chatbot A's Response]

{response_A}

[The End of Chatbot A's Response]

[The Start of Chatbot B's Response]

{response_B}

[The End of Chatbot B's Response]

[The Start of Critique]

{critique}

[The End of Critique]

Figure 13. Prompt: Edit-only Refinement Instruction based on Critiques.

Prompt: Calculation of Similarity w/ Core Human Critiques

I will provide you with a generated evaluation content and a reference evaluation content. Your task is to analyze the similarity between the <Generated Evaluation Content> and the <Reference Evaluation Content> by calculating F1 scores based on their key arguments.

Core Principle: Focus exclusively on "Core Arguments" - decisive reasons that are powerful enough to justify the final choice on their own. Identify these key justifications, not minor points.

Part 1: First check

First check if the generated critique repeats the same point across multiple times. If yes, directly output without conducting part 2:

```
<thinking>
  Put here how the generated critique repeats points.
</thinking>
```

```
<scores>
  <critique_f1>0</critique_f1>
  <critique_precision>0</critique_precision>
  <critique_recall>0</critique_recall>
</scores>
```

Part 2: Steps for F1 Score Calculation

1. Count Reference Key Arguments (N_ref):

- Check if the reference identifies a **fatal error** (critical factual error, harmful statement, or fundamental misunderstanding).
 - If yes:** Only this fatal error counts. Set **N_ref = 1**.
 - If no:** Count all unique Key Arguments (decisive reasons that could justify the choice by themselves). Set **N_ref** to this count.

2. Count Generated Key Arguments (N_gen):

- Identify all unique Key Arguments in the generated evaluation.
- Set **N_gen** to this count.

3. Count True Positives (TP):

- Initialize **TP = 0**.
- For each reference key argument, search for a match in generated key arguments.
- Matching Rule:** Both **semantic meaning** and **stance** (which response and positive/negative) must align.
 - Example: "Response A is more detailed" only matches with similar praise of Response A, not Response B.
 - For fatal errors: Generated must identify the same error in the same response.
- Each generated argument can only match once.
- Increment **TP** by 1 for each valid match.

4. Calculate Scores:

- Precision_critique:** TP / N_{gen} (0 if $N_{gen} = 0$)
- Recall_critique:** TP / N_{ref} (0 if $N_{ref} = 0$)
- CritiqueScore:** $2 * (Precision * Recall) / (Precision + Recall)$ (0 if $sum = 0$)

Output Format (rounded to 4 decimal places):

```
<thinking>
  Put the thinking process here.
</thinking>

<scores>
  <critique_f1>CritiqueScore</critique_f1>
  <critique_precision>Precision_critique</critique_precision>
  <critique_recall>Recall_critique</critique_recall>
</scores>

<Generated Evaluation Content>
{critiques}
</Generated Evaluation Content>

<Reference Evaluation Content>
{reference_critiques}
</Reference Evaluation Content>
```

Figure 14. Prompt: Calculation of Similarity w/ Core Human Critiques.

Prompt: Calculation of Similarity w/ All Human Critiques

I will provide you with a generated evaluation content and a reference evaluation content. Your task is to analyze the similarity between the <Generated Evaluation Content> and the <Reference Evaluation Content> by calculating F1 scores based on their all arguments.

Part 1: First check

First check if the generated critique repeats the same point across multiple times. If yes, directly output without conducting part 2:

```
<thinking>
  Put here how the generated critique repeats points.
</thinking>
```

```
<scores>
  <critique_f1>0</critique_f1>
  <critique_precision>0</critique_precision>
  <critique_recall>0</critique_recall>
</scores>
```

Part 2: Steps for F1 Score Calculation

1. Count Reference All Arguments (N_ref):

- Check if the reference identifies a **fatal error** (critical factual error, harmful statement, or fundamental misunderstanding).
 - If yes:** Only this fatal error counts. Set **N_ref = 1**.
 - If no:** Count all unique Arguments (decisive reasons that could justify the choice by themselves). Set **N_ref** to this count.

2. Count Generated All Arguments (N_gen):

- Identify all unique Arguments in the generated evaluation.
- Set **N_gen** to this count.

3. Count True Positives (TP):

- Initialize **TP = 0**.
- For each reference argument, search for a match in generated arguments.
- Matching Rule:** Both **semantic meaning** and **stance** (which response and positive/negative) must align.
 - Example: "Response A is more detailed" only matches with similar praise of Response A, not Response B.
 - For fatal errors: Generated must identify the same error in the same response.
- Each generated argument can only match once.
- Increment **TP** by 1 for each valid match.

4. Calculate Scores:

- Precision_critique:** TP / N_{gen} (0 if $N_{gen} = 0$)
- Recall_critique:** TP / N_{ref} (0 if $N_{ref} = 0$)
- CritiqueScore:** $2 * (Precision * Recall) / (Precision + Recall)$ (0 if $sum = 0$)

Output Format (rounded to 4 decimal places):

```
<thinking>
  Put the thinking process here.
</thinking>

<scores>
  <critique_f1>CritiqueScore</critique_f1>
  <critique_precision>Precision_critique</critique_precision>
  <critique_recall>Recall_critique</critique_recall>
</scores>

<Generated Evaluation Content>
{critiques}
</Generated Evaluation Content>

<Reference Evaluation Content>
{reference_critiques}
</Reference Evaluation Content>
```

Figure 15. Prompt: Calculation of Similarity w/ All Human Critiques.

Prompt: LLM-as-a-Meta-Judge

You are an expert evaluator tasked with assessing the quality of critiques comparing two responses.
You will be given:

1. A conversation history
2. Response A
3. Response B
4. One or more critiques comparing Response A and Response B

Your task is to evaluate whether the critique(s) are accurate and correct based on the actual content of the responses. Assign a score between 0 and 1, where higher scores indicate more accurate critiques.

You must provide your response in the following XML format:

```
<thinking>
  Put your detailed analysis here. Examine the critique
  against the actual responses and explain your reasoning
  for the score.
</thinking>
```

```
<scores>
  <critique_f1>Score</critique_f1>
  <critique_precision>Score</critique_precision>
  <critique_recall>Score</critique_recall>
</scores>
```

Note: Assign the same score to all three metrics (critique_f1, critique_precision, and critique_recall).

```
<Conversation History>
{conv_his}
</Conversation History>
```

```
<Response A>
{response_A}
</Response A>
```

```
<Response B>
{response_B}
</Response B>
```

```
<Critiques>
{critiques}
</Critiques>
```

Figure 16. Prompt: LLM-as-a-Meta-Judge.