ELIMINATING INDUCTIVE BIAS IN REWARD MODELS WITH INFORMATION-THEORETIC GUIDANCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Reward models (RMs) are crucial in reinforcement learning from human feedback (RLHF) to align large language models (LLMs) with human values. However, RM training data is commonly recognized as low-quality, always containing preference conflicts and inductive biases, such as response length or speaking style, which can easily lead to reward overfitting and hacking. A few recent RM debiasing methods either target merely a single specific type of preference bias or only address simple linear bias relations such as Pearson coefficients. To mitigate more complicated inductive bias of reward modeling, inspired by the information bottleneck, we introduce a novel information-theoretic debiasing method called **D**ebiasing via **I**nformation optimization for **R**M (DIR). More specifically, our method trains RMs by maximizing the mutual information (MI) between preference prediction and input response pairs, while minimizing the MI between RM outputs and biased attributes of preference inputs. With the theoretical justification of information theory, DIR can handle different types of bias with more comprehensive non-linear correlations, enlarging its real-world application scenarios. In experiments, we verify the effectiveness of DIR with three types of inductive biases: response length, sycophancy, and format. Based on the numerical results, we discover that DIR can not only effectively diminish target inductive biases but also improve RLHF performances on various benchmarks with better generalization abilities.

1 INTRODUCTION

Aligning Large Language Models (LLMs) (OpenAI, 2024; Touvron et al., 2023; Yang et al., 2024) with human values is paramount for ensuring their safe and reliable deployment, especially in open-domain conversational applications, where models must be helpful and harmless (Ouyang et al., 2022b). To this end, reinforcement learning from human feedback (RLHF) (Ouyang et al., 2022b; Rafailov et al., 2024b; DeepSeek-AI, 2025) has become the fundamental technique for encouraging LLM behavior toward human preferences, which operates by first training a reward model (RM) on a collection of human preference judgments, and then using this RM as a proxy for human values to guide the policy LLM's optimization via reinforcement learning (RL). However, the robustness and efficacy of RLHF have been continuously challenged by reward hacking, a phenomenon where the policy model exploits vulnerabilities in the reward model (RM) to achieve high rewards without satisfying the intended human objectives (Skalse et al., 2025; Gao et al., 2023; Amodei et al., 2016).

The vulnerabilities of RMs are commonly derived from the low quality of the human feedback annotation, which contains various toxic *inductive biases*. For example, annotators have always been instructed to choose more informative responses, whereas more detailed responses usually have longer response lengths. Learning on this biased human feedback dataset can lead the reward model to ignore the true response quality and only favor responses with longer lengths (Singhal et al., 2023). Besides the response length bias, stylistic and format patterns (Zhang et al., 2025), and sycophantic phrasing (Sharma et al., 2023; Denison et al., 2024) have been gradually recognized as typical inductive bias in reward modeling, which critically hinders reliability and safety of RLHF (Gao et al., 2023; Coste et al., 2023).

To mitigate inductive biases in reward modeling, recent studies have made some preliminary explorations. Bu et al. (2025), Chen et al. (2024) and Zhang et al. (2025) consider Pearson Coefficient (Benesty et al., 2009) as the bias measurement and minimize it jointly with the reward modeling loss. However, the Pearson Coefficient only captures the simplest linear correlation between RM

and the bias attributes, which are not sufficiently applicable in more general scenarios. Shen et al. (2023) adds another RM head to predict response length score, which lacks theoretical justification, and is only applicable with scalar types of inductive biases. Wang et al. (2025a) imposes overly-restrictive external constraints, such as enforcing distributional invariance via Maximum Mean Discrepancy (MMD) (Gretton et al., 2012), risk distorting the reward landscape by collapsing the scores of functionally disparate response groups. On the other hand, approaches relying on unreliable indirect internal compression, like the standard information bottleneck (Tishby et al., 2000) used in InfoRM (Miao et al., 2024), offer no guarantee of discarding a bias attribute, especially when it is strongly correlated with the true preference signal.

To address the inductive biases of RM generally with theoretical guarantees, we proposed an information-theoretic bias-disentangling framework called **D**ebiasing via **I**nformation optimization for **R**M (DIR). More specifically, we model the complicated inductive biases in human feedback with the concept of mutual information (MI) from the perspective of information theory (Kullback, 1997). Then we maximize the mutual information between the RM prediction and the preference data inputs, and minimize the mutual information between the bias attributes and the RM scores, simultaneously. To make the above objective tractable, we employ a dual-bound optimization strategy: a variational lower bound preserves essential preference information, while another variational upper bound actively suppresses information related to the bias. Furthermore, to ensure broad applicability, we design a comparative regularizer that operates on relative bias attributes between response pairs, rather than absolute values. This unique design allows DIR to robustly handle diverse and complex biases without distorting the underlying reward landscape, leading to a debiased RM that demonstrates better generalization and performance in downstream RLHF tasks. We summarize our contributions as follows:

1. We propose a novel, explicit information-theoretic framework that transforms debiasing from an indirect hope into a direct, supervised optimization objective, offering a more principled and targeted solution.
2. We design a practical and generalizable implementation that is computationally efficient and can be seamlessly adapted to mitigate diverse forms of bias without requiring architectural modifications to the base RM.
3. Extensive experiments demonstrate that our method significantly outperforms existing approaches in reducing reward hacking, leading to more robustly aligned LLMs that achieve better performance on both academic and preference-based benchmarks.

2 BACKGROUND

Reinforcement Learning from Human Feedback (RLHF) has become the essential training process to align LLMs with human values (Ouyang et al., 2022a). With a well-learned reward model (RM) $r_\phi(\boldsymbol{x}, \boldsymbol{y})$ scoring the degree of human preference of generated response $\boldsymbol{y} \in \mathcal{Y}$ given input prompt $\boldsymbol{x} \in \mathcal{X}$, RLHF optimizes the LLM policy $\pi_\theta(\boldsymbol{y}|\boldsymbol{x})$ with the follow objective:

$$\mathbb{E}_{\boldsymbol{x}\sim\mathcal{D},\boldsymbol{y}\sim\pi_\theta(\cdot|\boldsymbol{x})}\Big[r(\boldsymbol{x},\boldsymbol{y}) - \beta \cdot \mathrm{KL}[\pi_\theta(\boldsymbol{y}|\boldsymbol{x})\|\pi_{\text{ref}}(\boldsymbol{y}|\boldsymbol{x})]\Big], \tag{1}$$

where $\pi_{\text{ref}}(\boldsymbol{y}|\boldsymbol{x})$ is the initial model policy served as a reference, $\beta > 0$ controls the strength of a Kullback-Leibler (KL) divergence (Csiszár, 1975) between the reference model $\pi_{\text{ref}}(\boldsymbol{y}|\boldsymbol{x})$ and the current policy $\pi_\theta(\boldsymbol{y}|\boldsymbol{x})$. To train LLMs with the above objective, Proximal Policy Optimization (PPO) (Schulman et al., 2017) has been recognized as the mainstream optimization approach. Group Relative Policy Optimization (GRPO) further removes the critic model in PPO and uses a simplified group-related advantage approximation instead, which has shown competitive performance with practically simpler infrastructures (Shao et al., 2024).

Reward Modeling targets learning the human preference distribution via a parameterized reward model (RM) $r_\phi : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, where $r_\phi(\boldsymbol{x}, \boldsymbol{y})$ is the predicted reward score of the input prompt $\boldsymbol{x}$ and the corresponding response $\boldsymbol{y}$. For every input $\boldsymbol{x}$, given a pair of response $(\boldsymbol{y}, \bar{\boldsymbol{y}})$, we can calculate the "preference" by comparing the reward scores: if $r(\boldsymbol{x}, \boldsymbol{y}) > r(\boldsymbol{x}, \bar{\boldsymbol{y}})$, then $\boldsymbol{y}$ is predicted as a more "preferred" response than $\bar{\boldsymbol{y}}$ (denote as $\boldsymbol{y} \succ \bar{\boldsymbol{y}}$) and vice versa. We use a binary indicator $\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}}$ to representation the event of "human preference": $\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}} = 1$, if $\boldsymbol{y} \succ \bar{\boldsymbol{y}}$; and $\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}} = 0$, if

$\boldsymbol{y} \prec \bar{\boldsymbol{y}}$. Then reward model predicts the preference $\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}}$ as a conditional Bernoulli variable:

$$q_\phi\Big(\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}}=1\Big|\boldsymbol{x},\boldsymbol{y},\bar{\boldsymbol{y}}\Big)=\frac{\exp(r_\phi(\boldsymbol{x},\boldsymbol{y}))}{\exp(r_\phi(\boldsymbol{x},\boldsymbol{y}))+\exp(r_\phi(\boldsymbol{x},\bar{\boldsymbol{y}}))}=\sigma\Big(r_\phi(\boldsymbol{x},\boldsymbol{y})-r_\phi(\boldsymbol{x},\bar{\boldsymbol{y}})\Big), \quad (2)$$

where $\sigma(\cdot)$ is a Sigmoid function. Note that the ground-truth human preference distribution $p^*(\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}}|\boldsymbol{x},\boldsymbol{y},\bar{\boldsymbol{y}})$ is unknown. Instead, we can maximize the log-likelihood of q_ϕ with a group of human preference data $\mathcal{D}_{\text{Pref}}=\{(\boldsymbol{x}_i,\boldsymbol{y}_i^w,\boldsymbol{y}_i^l)\}_{i=1}^N$, where each $\boldsymbol{y}^w\succ\boldsymbol{y}^l$ is annotated by human judgment *w.r.t.* the response quality:

$$\begin{aligned}\mathcal{L}_{\text{RM}}(\phi)&=-\mathbb{E}_{\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}}\sim p^*}\Big[\log q_\phi\Big(\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}}\Big|\boldsymbol{x},\boldsymbol{y},\bar{\boldsymbol{y}}\Big)\Big]\approx-\frac{1}{N}\sum_{i=1}^N\left[\log q_\phi(\boldsymbol{y}_i^w\succ\boldsymbol{y}_i^l|\boldsymbol{x}_i,\boldsymbol{y}_i^w,\boldsymbol{y}_i^l)\right]\\&=-\frac{1}{N}\sum_{i=1}^N[\log\sigma(r_\phi(\boldsymbol{x}_i,\boldsymbol{y}_i^w)-r_\phi(\boldsymbol{x}_i,\boldsymbol{y}_i^l))].\quad\text{(by equation 2)}\end{aligned} \quad (3)$$

Equation 3 is commonly recongized as the Bradley-Terry (Bradley & Terry, 1952) ranking loss.

Information-theoretic Methods optimize deep models from the perspective of information theory (Chen et al., 2016; Hjelm et al., 2019; Yuan et al., 2021; Cheng et al., 2021). The core methodology of information-theoretic methods is to regard the feed-forward process of neural networks as an information channel transmission, where the correlation between different neural embeddings is measured by mutual information (MI) as:

$$I(\boldsymbol{x};\boldsymbol{y})=\mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}\Big[\log\frac{p(\boldsymbol{x},\boldsymbol{y})}{p(\boldsymbol{x})p(\boldsymbol{y})}\Big]=\text{KL}\Big[p(\boldsymbol{x},\boldsymbol{y})\Big\|p(\boldsymbol{x})p(\boldsymbol{y})\Big], \quad (4)$$

where $p(\boldsymbol{x},\boldsymbol{y})$ is the joint distribution, and $p(\boldsymbol{x})$ and $p(\boldsymbol{y})$ are the marginal distributions. Due to its general ability to capture arbitrary non-linear correlations, MI has achieved considerable success as a learning objective in various deep learning tasks (Chen et al., 2016; Belghazi et al., 2018; Hjelm et al., 2019). However, the exact MI value in equation 4 is challenging to compute, due to the intractable expectation *w.r.t.* $p(\boldsymbol{x},\boldsymbol{y})$, especially when only samples from $p(\boldsymbol{x},\boldsymbol{y})$ are provided. To address this, several approximation methods have been proposed to estimate MI from samples using tractable variational bounds (Oord et al., 2018; Cheng et al., 2020; Belghazi et al., 2021). Barber-Agakov (BA) bound (Barber & Agakov, 2004) provides a simple lower bound approximation of MI, by introducing a variational approximation $q_\theta(\boldsymbol{y}|\boldsymbol{x})$:

$$I(\boldsymbol{x};\boldsymbol{y})\geq\mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log q_\theta(\boldsymbol{y}|\boldsymbol{x})]+H[p]=:I_{\text{BA}}(\boldsymbol{x};\boldsymbol{y}), \quad (5)$$

where $H[p]$ is the entropy of the ground-truth distribution $p(\boldsymbol{x},\boldsymbol{y})$. Besides, Cheng et al. (2020) proposed a variational contrastive log-ratio upper bound (CLUB) also utilizing the variational approximation $q_\theta(\boldsymbol{y}|\boldsymbol{x})$:

$$I(\boldsymbol{x};\boldsymbol{y})\leq\mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log q_\theta(\boldsymbol{y}|\boldsymbol{x})]-\mathbb{E}_{p(\boldsymbol{x})p(\boldsymbol{y})}[\log q_\theta(\boldsymbol{y}|\boldsymbol{x})]=:I_{\text{CLUB}}(\boldsymbol{x};\boldsymbol{y}), \quad (6)$$

By minimizing equation 6, the amount of information between $\boldsymbol{x}$ and $\boldsymbol{y}$ is effectively reduced. We give the proof of BA bound and CLUB in Appendix B.1 and Appendix B.1, respectively.

A well-known application of information-theoretic methods is the *information bottleneck* (IB) (Tishby et al., 2000), which aims to learn a compressed but informative representation $\boldsymbol{h}$ of an input $\boldsymbol{x}$ to the output $\boldsymbol{y}$ as a trade-off between two MI terms:

$$\min I(\boldsymbol{x};\boldsymbol{h})-\lambda\cdot I(\boldsymbol{h};\boldsymbol{y}), \quad (7)$$

where hyperparameter $\lambda>0$ controls the balance between compressing the input $\boldsymbol{x}$ and retaining relevant information for the prediction $\boldsymbol{y}$. IB has been recognized as a powerful tool for representation learning and widely applied to diverse deep learning scenarios (Saxe et al., 2019; Wan et al., 2021; Federici et al., 2020).

3 METHODOLOGY

We begin by revisiting reward modeling from a perspective of information theory. Motivated by the information bottleneck, our core idea is learning a reward model $r_\phi(\boldsymbol{x},\boldsymbol{y})$ parameterized by ϕ that is

maximally informative compressed about the true preference relation, while simultaneously being minimally informative about an inductive bias with a pre-defined bias attribute $\boldsymbol{b}$. Formally, we define our objective as follows:

$$\max_{\phi} \underbrace{I\left(\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}}; \boldsymbol{x}, \boldsymbol{y}, \bar{\boldsymbol{y}}\right)}_{\text{Preference Term}} - \lambda \cdot \underbrace{I\left(\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}}; \boldsymbol{b}\right)}_{\text{Debias Term}}, \tag{8}$$

where λ is a hyperparameter balancing the trade-off between preference learning and debiasing. Ideally, minimizing this objective should encourage the reward model r_ϕ to capture the true performance signal from the input triplet $(\boldsymbol{x}, \boldsymbol{y}, \bar{\boldsymbol{y}})$, while decreasing the reliance on the bias attribute $\boldsymbol{b}$. However, directly optimizing this mutual information-based objective is computationally intractable due to the difficulty in estimating mutual information in high-dimensional spaces.

3.1 Practical Implementation with Differentiable Losses

To render equation 8 tractable, we derive differentiable surrogate losses that approximate its constituent terms. Specifically, we minimize a total loss $\mathcal{L}_{\text{total}}$ composed of a preference loss $\mathcal{L}_{\text{pref}}$ and a debiasing loss $\mathcal{L}_{\text{debias}}$.

Preference Term. Instead of directly maximizing $I\left(\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}}; \boldsymbol{x}, \boldsymbol{y}, \bar{\boldsymbol{y}}\right)$, we can use its lower bound approximation by introducing a BA estimator of equation 5 as follows:

$$\max\ I\left(\mathbf{1}_{\boldsymbol{y}\succ\boldsymbol{y}}; \boldsymbol{x}, \boldsymbol{y}, \bar{\boldsymbol{y}}\right) \geq \max \mathbb{E}_{p^*(\boldsymbol{x},\boldsymbol{y},\bar{\boldsymbol{y}},\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}})}[\log q_\phi(\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}}|\boldsymbol{x}, \boldsymbol{y}, \bar{\boldsymbol{y}})] + H[p^*], \tag{9}$$

which is guaranteed by the non-negative nature of the KL term, and $H[p^*]$ is a constant of the ground-truth human preference distribution p^*. By equation 3, we know the other term is exactly the RM BT ranking loss. Therefore, by minimizing the RM BT ranking loss, we actually maximize the mutual information between the preference variable $\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}}$ and the input triplet $(\boldsymbol{x}, \boldsymbol{y}, \bar{\boldsymbol{y}})$, encouraging the reward model r_ϕ to assign a higher score to the preferred response $\boldsymbol{y}$. Therefore, we have $\mathcal{L}_{\text{pref}} = -\frac{1}{B}\sum_{i=1}^{B}[\log \sigma(r_\phi(\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i) - r_\phi(\boldsymbol{x}_i, \bar{\boldsymbol{y}}_i))]$ with B samples.

Debiasing Term. For maximizing the debias term $-I\left(\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}}; \boldsymbol{b}\right)$, since $(\boldsymbol{y}, \bar{\boldsymbol{y}})$ contains sufficient information to determine bias $\boldsymbol{b}$, we notice that $\boldsymbol{b} \to (\boldsymbol{x}, \boldsymbol{y}, \bar{\boldsymbol{y}}) \to \boldsymbol{H} \to \mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}}$ is a Markov Chain, where $\boldsymbol{H} = [\boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}), \boldsymbol{h}(\boldsymbol{x}, \bar{\boldsymbol{y}})]$ is the last hidden state of the reward model transformer architecture. According to the data processing inequality (DPI) and the CLUB upper bound estimator, we have

$$I\left(\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}}; \boldsymbol{b}\right) \leq I\left(\boldsymbol{H}; \boldsymbol{b}\right) \leq I_{\text{CLUB}}(\boldsymbol{H}; \boldsymbol{b}), \tag{10}$$

where

$$I_{\text{CLUB}}(\boldsymbol{H}; \boldsymbol{b}) \approx \frac{1}{B}\sum_{i=1}^{B}\left[\log q_\psi(\boldsymbol{b}_i|\boldsymbol{H}_i) - \frac{1}{B}\sum_{j=1}^{B}\log q_\psi(\boldsymbol{b}_j|\boldsymbol{H}_i)\right]. \tag{11}$$

This allows us to minimize a tighter upper bound, $I_{\text{CLUB}}(\boldsymbol{H}; \boldsymbol{b})$, shifting the debiasing pressure to the information-rich representation $\boldsymbol{H}$. Therefore, minimizing I_{CLUB} serves an effective approximation of maximizing $-I\left(\mathbf{1}_{\boldsymbol{y}\succ\bar{\boldsymbol{y}}}; \boldsymbol{b}\right)$, which minimizes mutual information by training a variational network $q_\psi(\boldsymbol{b}|\boldsymbol{H})$[1] in an adversarial manner, encouraging the reward model r_ϕ to produce representations $\boldsymbol{H}$ that are statistically independent of the bias attribute $\boldsymbol{b}$, thus the final predicted preference relation should be non-predictive of the $\boldsymbol{b}$. Moreover, as proved in Cheng et al. (2020), the better $q_\psi(\boldsymbol{b}_i|\boldsymbol{H}_i)$ approximates real $p(\boldsymbol{b}_i|\boldsymbol{H}_i)$, the more accurate I_{CLUB} serves as the MI upper bound. Therefore, we also maximize the log-likelihood of $q_\psi(\boldsymbol{b}_i|\boldsymbol{H}_i)$ with samples $\{(\boldsymbol{H}_i, \boldsymbol{b}_i)\}_{i=1}^{B}$ in addition.

A Unified Comparative Framework for Debiasing. To make the debiasing mechanism more targeted and align it with the comparative nature of preference learning, we introduce a unified comparative framework, which also handles various types of biases in a consistent and unified manner. Note that the preference task itself relies on the *difference* in rewards, which stems from the difference in representations. Therefore, instead of using the concatenated representation $\boldsymbol{H}$, which may contain redundant information from the shared prompt $\boldsymbol{x}$, we focus on the representation

[1]Practically, we find a simple Linear-ReLU-Linear network is enough to serve as q_ψ that is detailed in Appendix C.

Algorithm 1: Our DIR Training Process Within Mini-Batch.

Input : RM Dataset $\mathcal{D}_{\text{Pref}} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i^w, \boldsymbol{y}_i^l)\}_{i=1}^N$, hyper-parameter λ, pre-defined bias attribute $\boldsymbol{b}$.
Output : Trained reward model r

Initialize a reward model r with parameters ϕ;
Initialize a variational model q with parameters ψ;
while *each training iteration* **do**
  Sample a mini-batch of triplets $\{(\boldsymbol{x}_i, \boldsymbol{y}_i^w, \boldsymbol{y}_i^l)\}_{i=1}^B \sim \mathcal{D}_{\text{Pref}}$;
  Encode each $(\boldsymbol{x}_i, \boldsymbol{y}_i^w)$ and $(\boldsymbol{x}_i, \boldsymbol{y}_i^l)$ into embeddings $\boldsymbol{h}_i^w = r_{\phi_{\text{base}}}(\boldsymbol{x}_i, \boldsymbol{y}_i^w)$, $\boldsymbol{h}_i^l = r_{\phi_{\text{base}}}(\boldsymbol{x}_i, \boldsymbol{y}_i^l)$;
  Calculate each representation difference $\Delta\boldsymbol{h}_i = \boldsymbol{h}_i^w - \boldsymbol{h}_i^l$ and obtain $\boldsymbol{b}_i^{\text{rel}}$ through $(\boldsymbol{y}_i^w, \boldsymbol{y}_i^l)$ with $\boldsymbol{b}$;
  Update the variational approximation $q_\psi(\boldsymbol{b}_i^{\text{rel}}|\Delta\boldsymbol{h}_i)$ by maximizing log-likelihood with $\{(\Delta\boldsymbol{h}_i, \boldsymbol{b}_i^{\text{rel}})\}$;
  Calculate $\mathcal{L}_{\text{debias}}$ with $q_\psi(\boldsymbol{b}^{\text{rel}}|\Delta\boldsymbol{h})$ and $\{(\Delta\boldsymbol{h}_i, \boldsymbol{b}_i^{\text{rel}})\}_{i=1}^B$, and $\mathcal{L}_{\text{pref}}$ with $\{(\boldsymbol{x}_i, \boldsymbol{y}_i^w, \boldsymbol{y}_i^l)\}_{i=1}^B$;
  Learning loss $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{pref}} + \lambda \cdot \mathcal{L}_{\text{debias}}$;
  Update reward model r and variational network q by gradient descent with respect to $\mathcal{L}_{\text{total}}$;
end

difference, $\Delta\boldsymbol{h} = \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{y}) - \boldsymbol{h}(\boldsymbol{x}, \bar{\boldsymbol{y}})$, which isolates the features that distinguish the two responses. Here, with slight abuse of notation, for each input preference pair $(\boldsymbol{x}_i, \boldsymbol{y}_i)$, we use $\boldsymbol{h}_i = r_{\phi_{\text{base}}}(\boldsymbol{x}_i, \boldsymbol{y}_i)$ to denote the final hidden-state representation of the last valid token from the transformer base $r_{\phi_{\text{base}}}$ of the reward model r_ϕ.

Correspondingly, we define a *relative bias attribute*, $\boldsymbol{b}^{\text{rel}} \in \{0, 1\}$, for each pair, indicating which response exhibits more of the bias (e.g., $\boldsymbol{b}^{\text{rel}} = \mathbf{1}(\text{length}(\boldsymbol{y}) > \text{length}(\bar{\boldsymbol{y}}))$). This transforms our practical debiasing goal into minimizing $I\left(\Delta\boldsymbol{h}; \boldsymbol{b}^{\text{rel}}\right)$, where the variational network q_ψ is thus trained as a binary classifier on this more focused input-target pair: $q_\psi(\boldsymbol{b}^{\text{rel}}|\Delta\boldsymbol{h})$. By minimizing the CLUB objective within this framework, we encourage the reward model to learn representations whose differences are informative about true preference but are invariant to relative differences in the bias attribute. In summary, for a batch training data of size B, $\mathcal{L}_{\text{debias}} = \frac{1}{B}\sum_{i=1}^B [\log q_\psi(\boldsymbol{b}_i^{\text{rel}}|\Delta\boldsymbol{h}_i) - \frac{1}{B}\sum_{j=1}^B \log q_\psi(\boldsymbol{b}_j^{\text{rel}}|\Delta\boldsymbol{h}_i)]$.

Final Objective. Finally, we jointly optimize the reward model parameters ϕ and the variational parameters ψ by minimizing the complete training objective:

$$\mathcal{L}_{\text{total}}(\phi, \psi) = \mathcal{L}_{\text{pref}}(\phi) + \lambda\mathcal{L}_{\text{debias}}(\phi_{\text{base}}, \psi), \tag{12}$$

Parameters of the transformer base $\phi_{\text{base}} \subset \phi$ are updated by gradients from both losses, while the RM's prediction head and the variational network q_ψ are updated by their respective objectives. During inference, the variational network is discarded, allowing the debiased reward model r_ϕ to be used without any overhead. Given a pre-collected human preference dataset $\mathcal{D}_{\text{Pref}} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i^w, \boldsymbol{y}_i^l)\}_{i=1}^N$, we highlight the training process in Algorithm 1 and visualize our framework in Appendix D.

3.2 Discussion

Our final training $\mathcal{L}_{\text{total}}$ in equation 17 optimizes a tractable objective of the ideal information-theoretic objective in equation 8, which is theoretically grounded by several reasons. First, by targeting $I(\boldsymbol{H}; \boldsymbol{b})$, we enforce a stricter constraint on the bias information than targeting the final prediction, as justified by the DPI. Second, our comparative framework, which minimizes $I\left(\Delta\boldsymbol{h}; \boldsymbol{b}^{\text{rel}}\right)$, serves as a principled and targeted implementation of this constraint, directly addressing the representational differences that drive biased decisions. Finally, using the CLUB estimator is an empirically validated technique for minimizing a tight upper bound on mutual information. Therefore, our practical loss function guides the reward model towards the ideal objective of being maximally informative about preferences while remaining invariant to the specified bias.

4 Related Work

Reward Hacking. The issue of reward hacking, or specification gaming, has been increasingly recognized as a significant challenge for the stable and effective post-training of LLMs via RLHF (Langosco et al., 2023; Amodei et al., 2016; Hurst et al., 2024; Kaufmann et al., 2024; Skalse et al.,

2025; Zhang et al., 2025; Li et al., 2025). It occurs when an agent exploits unforeseen loopholes or proxies in a misspecified reward function to achieve high scores without fulfilling the intended goal (Pan et al., 2022). In RLHF, the learned RM serves as a proxy for true human preferences. If this RM inadvertently learns inductive bias from the preference data (e.g., a bias towards more verbose (Singhal et al., 2023), or sycophantic responses (Sharma et al., 2023)), the LLM being optimized will learn to exploit these flaws, leading to a degradation in true performance (Hurst et al., 2024). Prior work has sought to mitigate reward hacking by empowering RMs, including better data curation (Liu et al., 2024a; Wang et al., 2025b; Dubois et al., 2024), model scaling up (Wang et al., 2025b) and ensembling (Wang et al., 2024), reward post-hoc calibration (Huang et al., 2024), causal inference (Shen et al., 2023; Wang et al., 2025a), disentangled reward learning (Bu et al., 2025; Chen et al., 2024) and other additional constraints (Miao et al., 2024).

More related to our work, several recent efforts have sought to mitigate biases in reward modeling. A prominent line of work focuses on minimizing simple statistical correlations. For instance, methods like ODIN (Chen et al., 2024), ALBM (Bu et al., 2025), and the approach by Zhang et al. (2025) aim to reduce length or format bias by directly penalizing the Pearson correlation coefficient between the reward score and the bias attribute. While effective for simple associations, these methods are fundamentally limited as they only capture linear relationships, failing to address more complex, non-linear dependencies. Similarly, PoE (Shen et al., 2023) employs a specialized two-head architecture for length bias, but its attempt at causal disentanglement is purely heuristic. PoE does not explicitly model the preference-bias relationship, relying instead on the network to implicitly learn this separation from data, which offers neither formal guarantees nor generality.

In contrast, more principled frameworks have been proposed. CRM (Wang et al., 2025a) uses counterfactual invariance enforced by MMD, which may be overly restrictive and risk distorting the reward landscape. Closer to our approach, InfoRM (Miao et al., 2024) employs an information-theoretic framework to compress the entire latent representation, indirectly removing spurious information. Distinct from all these methods, our work is also motivated by information theory (Tishby et al., 2000) but introduces a more direct and targeted mechanism. By explicitly minimizing the mutual information that is capable of capturing arbitrary non-linear dependencies between the model's internal representation and the known bias attribute, DIR provides a principled and robust debiasing framework that is both general and effective, avoiding the limitations of linear metrics and the risks of purely heuristic, data-driven approaches.

Debias Methods aims at preventing models from learning and perpetuating undesirable biases present in training data (He et al., 2019; Nam et al., 2020; Blodgett et al., 2020). A prominent line of work involves learning representations that are invariant to sensitive or spurious attributes (Chuang et al., 2020). Methodologies to achieve this include adversarial training, where a discriminator attempts to predict the bias attribute from the model's representation (Nam et al., 2020); causal inference techniques that aim to disentangle causal factors from spurious ones (Zhou et al., 2023a); and information-theoretic approaches (Liu et al., 2023; Tartaglione et al., 2021). Our work falls into the latter category, where we introduce a novel and principal information-theoretic debiasing method for eliminating inductive bias in reward modeling.

5 Experiment

We evaluate the effectiveness of our debiased reward modeling framework on three practical bias settings, length, sycophancy, and format, by applying the method separately to each. Then we explore whether our method can alleviate the concurrent multi-bias situation.

5.1 Length Bias

Previous work demonstrates that reward models often tend to favor longer responses, leading them to assign higher reward scores for verbose completions rather than for substantive content (Singhal et al., 2023; Dubois et al., 2024). As a result, the aligned policy model would in turn learn to exploit this inductive bias, which is incentivized to generate unnecessarily verbose, repetitive, or circuitous text to maximize its expected reward, a behavior that directly contradicts the goal of aligning with nuanced human preferences for quality and conciseness. See detailed settings in Appendix E.1.

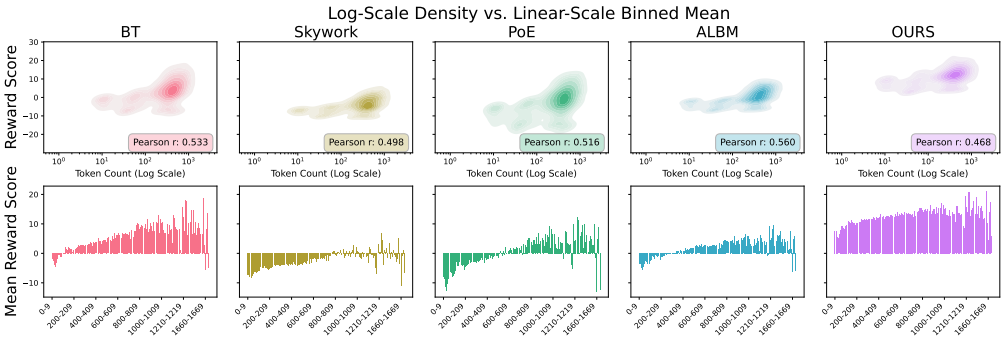

Figure 1: Evaluation of length bias in Reward Models on the RM-Bench. We compare the correlation between response length and reward score for RMs trained with different methods. Our approach yields the lowest Pearson correlation coefficient (0.468), proving its effective ability in assigning more uniform reward scores.

Table 1: We adopt the official evaluation implementation of the evalscope package by using 0-Shot, except for GSM8K, Race, and TriviaQA. Baseline: Llama3.1-8B-Instruct / OpenRLHF-Llama3-8B-SFT. **Bold** is the best. underline is the second-best. The Δ row indicates the performance change relative to the respective Baseline.

Benchmark	Llama3.1-8B-Instruct							OpenRLHF-Llama3-8B-SFT						
	Base	SK	PoE	LP	ALBM	InfoRM	Ours	Base	SK	PoE	LP	ALBM	InfoRM	Ours
$GSM8K_{acc}$-4shots	83.93	84.61	83.62	75.97	84.08	83.78	**84.84**	74.83	78.17	77.79	77.18	78.85	76.74	**79.08**
$Hellaswag_{acc}$	77.21	76.42	77.08	73.15	77.21	76.78	**77.33**	72.51	**74.76**	72.51	72.51	74.63	72.12	74.52
$IFeval_{acc}$	72.83	70.06	71.72	65.47	73.57	74.12	**78.00**	44.92	45.10	49.72	46.21	46.21	46.21	**52.31**
$MMLU_{acc}$	72.31	72.33	71.97	65.13	72.55	72.22	**72.64**	54.45	52.40	54.77	54.45	**55.25**	54.97	54.30
$ProcessBench_{acc}$	25.39	**29.49**	28.50	24.91	26.12	26.25	27.73	4.46	10.31	9.68	7.84	10.85	3.24	**13.82**
$Race_{acc}$-3shots	66.50	53.89	60.03	**78.90**	59.00	65.20	62.02	79.21	78.82	**81.39**	80.30	80.69	78.72	80.32
BBH_{acc}	64.52	65.69	60.50	61.10	64.84	66.13	**67.27**	61.20	62.68	62.69	62.28	61.10	61.62	**62.99**
$Humaneval_{pass@1}$	**70.12**	68.29	66.46	60.37	65.85	**70.12**	**70.12**	60.98	57.32	59.76	59.76	60.37	57.32	**63.41**
$TriviaQA_{acc}$-5shots	32.64	49.01	48.41	47.20	52.09	30.56	**55.86**	48.53	**52.86**	52.34	48.32	51.52	48.16	52.52
Avg. Performance	62.83	63.31	63.14	61.36	63.92	62.80	**66.20**	55.68	56.94	57.85	56.54	57.72	55.34	**59.25**
Δ	-	↑ 0.48	↑ 0.31	↓ 1.47	↑ 1.09	↓ 0.03	↑ 3.37	-	↑ 1.26	↑ 2.17	↑ 0.86	↑ 2.04	↓ 0.34	↑ 3.57

Reward Model Evaluation, Results and Analysis. We first evaluate the inherent length bias of RMs by analyzing the correlation between their scores and response lengths on the RM-Bench (Liu et al., 2024b). As visualized in Figure 1, the standard BT RM exhibits a strong, undesirable positive correlation between length and reward (Pearson r = 0.533). This confirms that even without an explicit preference for length in the training data[2], the model still learns a spurious "longer is better" heuristic, highlighting a fundamental issue in standard BT: the objective itself is susceptible to capturing such simple, non-causal patterns. While other debiasing methods show some improvement, our approach demonstrates an effective ability to mitigate this bias. Our RM achieves a Pearson correlation of just 0.468, the lowest among all evaluated methods. This quantitative advantage is further illustrated in the binned mean reward plots; the curve for our model is visibly flatter, confirming that it does not disproportionately reward longer responses. By learning to assign scores more uniformly across different lengths, our method produces a more reliable RM, preventing the policy from being misguided into generating unnecessarily verbose outputs during subsequent fine-tuning. We report the performance on RM-Bench in Appendix E.1

PPO Evaluation, Results and Analysis. We compare the performance of different PPO-optimized policies based on the corresponding RMs across several popular benchmarks. Table 1 demonstrates that mitigating length bias does not compromise, and ideally enhances, the policy's core reasoning and knowledge-based capabilities. On the Llama3.1-8B-Instruct backbone, our model ("OURS") achieves the highest average performance of 66.20, significantly outperforming strong baselines. This trend of improved performance is consistent across different base models, as our method also secures the top average score (59.25) on the OpenRLHF-Llama3-8B-SFT backbone, which shows that our fine-tuning strategy successfully improves objective performance by alleviating the length bias.

We also assess the user preference for policies fine-tuned using different reward models and compare average response length on the ArenaHard-v0.1 benchmark (Li et al., 2024). Figure 2 shows the head-to-head win rates of these challenger policies against strong opponents, as judged by Qwen3-

[2]Average token number of $(\boldsymbol{x}, \boldsymbol{y}^w)$ in the SK training set is less than $(\boldsymbol{x}, \boldsymbol{y}^l)$ ones (622.86 vs. 707.24).

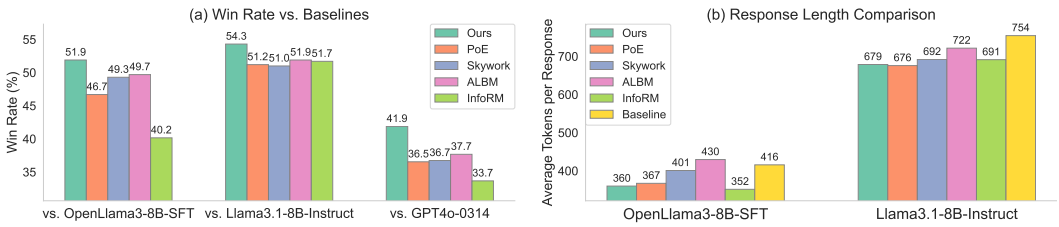

Figure 2: Evaluation on ArenaHard-v0.1 for policies fine-tuned with different RMs. (a) Head-to-head win rates. Policies are PPO fine-tuned from specified base models (from left to right: OpenLlama3-8B-SFT, Llama3.1-8B-Instruct, and Llama3.1-8B-Instruct, respectively) using five different RMs, which then act as challengers against opponents. (b) Average response length comparison.

235B-A22B-2507[3]. The policy trained with our RM ("Ours") consistently demonstrates the highest win rate across all conditions. For instance, in Figure 2 (a), when fine-tuned on Llama3.1-8B-Instruct, it achieves a remarkable 54.3% win rate against the baseline and 41.9% against GPT-4o-0314. Crucially, Figure 2 (b) reveals that this improved preference is achieved with expected conciseness. The policy guided by our RM produces shorter responses (e.g., 679 tokens on the Llama3.1 base) compared to policies guided by other RMs like ALBM (722 tokens) and the verbose original baseline (754 tokens). This combination of a relatively higher win rate and lower verbosity provides definitive evidence that our length-debiased reward model successfully guides PPO to produce a more efficient and human-aligned policy, effectively overcoming the common "longer is better" bias. Moreover, by directly targeting the known bias, DIR can effectively remove the spurious component while preserving preference-relevant information, thereby achieving both better direct debiasing and higher preference quality. In contrast, unsupervised debiased approaches like InfoRM do not consistently improve over the base models on either backbone and shows noticeably lower win rates, indicating a weaker trade-off between debiasing and performance.

In addition, we report the Win Rate performance on MT-Bench (Zheng et al., 2023) and Length Control Alpaca (Dubois et al., 2025) in Appendix E.1, from which we observe that DIR can yield policies that are preferred more often.

RM Training Cost Analysis. We analyze the computational overhead in terms of GPU memory consumption and training time, with a detailed comparison presented in Table 2. We use 8 GPU cards with full parameter training and DeepSpeed Zero-1(Rajbhandari et al., 2020). Our approach demonstrates highly comparable resource efficiency to existing methods. Specifically, the GPU memory usage of our method (57.22GB) is only marginally higher than the baseline (56.80GB) and on par with other techniques like ALBM (56.88GB). Regarding training time, while our method (67.09 minutes) requires a moderate increase compared to the simpler baseline (50.46 minutes), it remains competitive and aligns closely with other advanced methods such as ALBM (68.21 minutes). This analysis confirms that the significant performance improvements offered by our approach do not come at the expense of prohibitive computational costs, establishing it as a practical and efficient solution.

Table 2: Training cost comparison.

Method	GPU Memory	Training Time
Baseline	55.08GB	50.46m
PoE	56.80GB	55.35m
ALBM	57.22GB	78.21m
InfoRM	57.99GB	75.21m
Ours	56.88GB	67.09m

PPO Monitoring, Ablation Study, and Case Study. We visualize the PPO training dynamics metrics like RLHF Reward, KL divergence between the policy and the base, and KL divergence between following updated policies in Appendix E.1, which demonstrates that our RM helps make PPO training more stable with higher reward. In addition, we give a detailed ablation study on λ and representation difference in Appendix E.2, where the performance demonstrates the trade-off effects between preference learning and debiasing, and shows the effectiveness of representation difference than concatenation. We also provide specific case analysis in Appendix G.

Combination with Direct Preference Optimization (DPO) In addition to PPO, DPO (Rafailov et al., 2024a) has emerged as a powerful post-alignment method that directly trains the policy to increase the log-probability of the preferred response relative to the rejected one. Here, we explore whether our DIR can be effectively combined with DPO. Conceptually, DIR operates at the reward

[3] https://huggingface.co/Qwen/Qwen3-235B-A22B-Instruct-2507

Table 3: Evaluation on ArenaHard-v0.1 for policies fine-tuned with DPO, DPO+LC and DPO+OURS.

vs Base	OpenRLHF-Llama-3-8B-SFT Win Rate (%)	Length
DPO	38.63	436.55
+LC	40.96	407.23
+OURS	**45.27**	**404.61**
vs Base	Meta-Llama3.1-8B-Instruct Win Rate (%)	Length
DPO	44.06	700.87
+LC	46.57	691.43
+OURS	**49.09**	**684.67**

Table 4: Benchmark Performance under DPO training. Evaluation setting is the same as Table 1.

Benchmark	Meta-Llama3.1-8B-Instruct			OpenRLHF-Llama3-8B-SFT		
	DPO	+LC	+OURS	DPO	+LC	+OURS
$GSM8K_{acc}$-4shots	82.11	81.43	**82.56**	75.89	76.35	**77.26**
$Hellaswag_{acc}$	74.84	**75.17**	75.10	66.56	66.41	**73.91**
$IFeval_{acc}$	73.57	74.12	**74.68**	38.26	35.86	**41.59**
$MMLU_{acc}$	71.36	**71.58**	71.54	48.92	48.92	**57.01**
$ProcessBench_{acc}$	26.75	26.70	**27.60**	4.28	4.95	**6.93**
$Race_{acc}$-3shots	69.98	70.14	**70.29**	79.27	78.68	**79.97**
BBH_{acc}	**67.22**	66.81	66.99	59.73	60.36	**62.34**
$Humaneval_{pass@1}$	62.80	65.24	**69.51**	59.15	**60.37**	59.15
$TriviaQA_{acc}$-5shots	55.12	55.15	**55.29**	47.45	47.69	**48.93**
Avg. Performance	64.86	65.04	**65.95**	53.28	53.29	**60.12**
Δ	-	$\uparrow$ 0.18	$\uparrow$ 1.09	-	$\uparrow$ 0.01	$\uparrow$ 6.84

modeling stage and should not modify the DPO objective: DPO still optimizes the standard log-sigmoid preference loss, and our method only modifies preference signals by making them less correlated with inductive bias. Specifically, we add $\lambda\mathcal{L}_{\text{debias}}(\phi_{\text{base}}, \psi)$ to DPO loss. Empirically, we conduct the corresponding experiments, which show that our method can also improve DPO's performance with controlled length. We provide training details in Appendix E.3. Results in Table 3 indicate that our method leads to a final policy with both a better win-rate and more effective length control, effectively boosting vanilla DPO and also outperforming a specialized Length Controlled DPO (DPO+LC) variant Park et al. (2024). Results in Table 4 indicate that our method also leads to a final policy with performance gains, especially for the SFT model, which has not undergone a preference alignment. In summary, experiments on Table 3 and Table 4 suggest that debiased reward signals from DIR interact smoothly with DPO and effectively remove spurious gradients induced by length bias.

5.2 Sycophancy Bias

Sycophancy bias occurs when an RM learns to favor responses that agree with or flatter the user, rather than prioritizing factual accuracy and helpfulness Sharma et al. (2023); Wang et al. (2025a), which arises from inductive bias in preference data, where agreeable language is incorrectly associated with higher quality. Consequently, the policy model is misguided during RL fine-tuning to produce superficially pleasing but substantively poor outputs, undermining genuine alignment. Detailed experimental settings can be found in Appendix E.4.

Table 5: Reward model accuracy (%) on the sycophancy bias under varying contamination settings, where our method consistently achieves higher accuracy across most settings.

Settings		All.			Nat.			Adv.		
γ	α	BT	InfoRM	Ours	BT	InfoRM	Ours	BT	InfoRM	Ours
20%	30%	86.6	89.4	**90.2**	85.5	88.9	**89.8**	91.0	91.2	**93.6**
20%	50%	85.6	**89.8**	88.7	85.7	**90.3**	88.2	84.9	87.9	**90.9**
20%	70%	84.8	86.1	**87.1**	85.2	86.0	**87.5**	83.1	**86.6**	85.1
40%	30%	87.4	89.0	**90.9**	86.0	**88.1**	87.4	88.9	90.3	**93.9**
40%	50%	86.1	87.9	**88.7**	87.0	87.7	**89.8**	84.8	88.3	**89.1**
40%	70%	83.6	86.6	**88.0**	84.4	86.3	**87.4**	82.6	87.2	**88.6**
80%	30%	89.0	90.4	**91.3**	82.3	89.5	**88.0**	90.7	91.9	**92.2**
80%	50%	85.5	87.2	**88.1**	86.3	86.3	**86.2**	85.3	87.5	**90.3**
80%	70%	81.2	84.5	**86.2**	86.4	86.4	**87.2**	79.7	84.0	**86.2**

Evaluation, Results, and Analysis. To evaluate the models' susceptibility to sycophancy, we conduct an adversarial test. We take a clean evaluation set and create two versions: a "natural" version and a "sycophantic" version where the undesirable prefix is added to the rejected responses. We then measure the model's accuracy in correctly identifying the preferred response in both scenarios. A robust model should maintain its accuracy, whereas a biased model's performance will degrade when faced with the "flattering but wrong" responses. As shown in Table 5, the performance of the reward models varies under different settings. The BT model shows vulnerability to bias, as its accuracy on natural examples is generally the lowest, particularly under high contamination. While

InfoRM also shows a clear improvement and greater resilience than the BT baseline, our method demonstrates overall higher performance improvements across natural, adversarial, and overall settings, even under high contamination ratios. This pattern indicates that our explicit debiasing mechanism is effective at mitigating the influence of sycophantic signals, enabling the model to focus more on the intrinsic quality of the response.

5.3 Format Bias

Zhang et al. (2025); Long et al. (2024) have indicated that format biases (e.g., lists, emoji, and bold) widely exist in human and powerful preference models, and reward modeling can be easily attacked by a small amount of biased data and leads to significant format biases in downstream alignment tasks. We test DIR's ability to resist such bias and detailed experimental settings are in Appendix E.5.

Baseline, Evaluation, Results, and Analysis. By following LE (Zhang et al., 2025), we evaluate Ours against three baselines: a standard BT model, BT† (trained on data with format-biased samples removed), and LE. As shown in Table 6, the standard BT model exhibits a profound format bias, with win-rates of 89.0% and 92.5% for Bold and List formats respectively, confirming it has learned to associate these formats with higher quality. The naive BT† approach proves to be a suboptimal strategy; while it lowers the format preference, its downstream performance on RewardBench degrades significantly. Both LE and Ours effectively neutralize the format bias, bringing the win-rates close to the ideal 50% mark. The key distinction, however, emerges in the downstream RewardBench evaluation. While LE shows competent generalization, Ours demonstrates notably stronger performance across the more demanding Chat Hard, Safety, and Reasoning. This indicates that our approach strikes a better trade-off, successfully eliminating the format preference while simultaneously enhancing the model's core competencies in critical areas.

Table 6: Performance on both Bold and List format debiasing and downstream evaluation tasks. BT† indicates that deleting the samples with specific patterns.

Metric	BT	BT†	LE	Ours
Win-Rate (%)				
Bold	89.0	49.0	**50.5**	51.2
List	92.5	52.5	53.0	**52.0**
RewardBench (Filtered)				
Chat	**98.3**	92.2	97.2	93.0
Chat Hard	71.4	64.4	72.8	**80.1**
Safety	83.1	75.5	82.9	**89.6**
Reasoning	85.1	81.4	89.7	**92.2**

5.4 Concurrent Multi-Biases

Real datasets often exhibit multiple concurrent biases. In this section, we explore whether DIR can simultaneously deal with concurrent multi-biases, i.e., length and sycophantic biases. Overall, we find that the multi-bias DIR reduces both biases compared to the BT baseline, and even brings better generalization in some cases, indicating that multiple debiasing terms can be combined without significant destructive optimization conflicts. We give the details in Appendix E.6.

6 Conclusion

In this work, we introduce DIR, a novel framework designed to mitigate reward hacking caused by inductive biases in RLHF by applying information-theoretic principles to reward modeling. Unlike existing methods that target single biases (e.g., length or format) or only address simple linear correlations (e.g., Pearson Coefficient), DIR directly confronts the root cause of reward hacking, inductive bias in preference data, by implementing a dual-objective to explicitly disentangle these signals. DIR guides the reward model to learn representations that are predictive of true human preference while remaining invariant to the influence of known biases. Experiments across three distinct scenarios (i.e., length, sycophancy, and format bias) demonstrate DIR's effectiveness not only in neutralizing the target biases but also in enhancing downstream RLHF performance and generalization, validating our approach as a general and practical tool for building more robustly aligned models.

Ethics Statement

This work aims to enhance the fairness and reliability of LLMs by mitigating format biases, preventing models from "gaming" evaluations based on style over substance. Our method encourages a more accurate assessment of a model's true capabilities. We acknowledge that our method only

addresses the specific format biases targeted during training and does not mitigate broader societal or demographic biases. Furthermore, our ablation studies show that an overly aggressive debiasing coefficient (λ) can create a trade-off, potentially harming performance on simpler tasks. While we use public models and datasets, we recognize they may contain their own inherent biases. We believe our contribution is a positive step towards more robust and transparent AI alignment.

Reproducibility Statement

To ensure full reproducibility, we provide all necessary artifacts in both the Section Experiment, the Appendix, and the Supplementary Materials.

The complete source code, including training and evaluation scripts, is provided in the supplementary material. All datasets and base models we used in this manuscript are public, and we provide download scripts (`scripts/auto_download_data.sh` and `scripts/auto_download_model.sh`) for automated setup. Key hyperparameters are detailed in the paper. The exact commands for reproducing our main results are available in the provided shell scripts (e.g., `scripts/train_debias_rm.sh`).

References

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety, 2016. URL https://arxiv.org/abs/1606.06565.

David Barber and Felix Agakov. The im algorithm: a variational approach to information maximization. *Advances in neural information processing systems*, 16(320):201, 2004.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and Devon Hjelm. Mutual information neural estimation. In *International conference on machine learning*, pp. 531–540. PMLR, 2018.

Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, and R Devon Hjelm. Mine: Mutual information neural estimation, 2021. URL https://arxiv.org/abs/1801.04062.

Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. Pearson correlation coefficient. In *Noise reduction in speech processing*, pp. 1–4. Springer, 2009.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. Language (technology) is power: A critical survey of "bias" in NLP. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 5454–5476, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.485. URL https://aclanthology.org/2020.acl-main.485/.

Ralph Allan Bradley and Milton E Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Yuyan Bu, Liangyu Huo, Yi Jing, and Qing Yang. Beyond excess and deficiency: Adaptive length bias mitigation in reward models for rlhf. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 3091–3098, 2025.

Lichang Chen, Chen Zhu, Davit Soselia, Jiuhai Chen, Tianyi Zhou, Tom Goldstein, Heng Huang, Mohammad Shoeybi, and Bryan Catanzaro. Odin: Disentangled reward mitigates hacking in rlhf, 2024. URL https://arxiv.org/abs/2402.07319.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas

Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *Advances in neural information processing systems*, 29, 2016.

Pengyu Cheng, Weituo Hao, Shuyang Dai, Jiachang Liu, Zhe Gan, and Lawrence Carin. Club: A contrastive log-ratio upper bound of mutual information. In *International conference on machine learning*, pp. 1779–1788. PMLR, 2020.

Pengyu Cheng, Weituo Hao, Siyang Yuan, Shijing Si, and Lawrence Carin. Fairfil: Contrastive neural debiasing method for pretrained text encoders, 2021. URL https://arxiv.org/abs/2103.06413.

Ching-Yao Chuang, Joshua Robinson, Yen-Chen Lin, Antonio Torralba, and Stefanie Jegelka. Debiased contrastive learning. *Advances in neural information processing systems*, 33:8765–8775, 2020.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Thomas Coste, Usman Anwar, Robert Kirk, and David Krueger. Reward model ensembles help mitigate overoptimization. *arXiv preprint arXiv:2310.02743*, 2023.

Imre Csiszár. I-divergence geometry of probability distributions and minimization problems. *The annals of probability*, pp. 146–158, 1975.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, Zhiyuan Liu, and Maosong Sun. Ultrafeedback: Boosting language models with scaled ai feedback, 2024. URL https://arxiv.org/abs/2310.01377.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Carson Denison, Monte MacDiarmid, Fazl Barez, David Duvenaud, Shauna Kravec, Samuel Marks, Nicholas Schiefer, Ryan Soklaski, Alex Tamkin, Jared Kaplan, et al. Sycophancy to subterfuge: Investigating reward-tampering in large language models. *arXiv preprint arXiv:2406.10162*, 2024.

Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. Rlhf workflow: From reward modeling to online rlhf, 2024. URL https://arxiv.org/abs/2405.07863.

Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacafarm: A simulation framework for methods that learn from human feedback, 2024. URL https://arxiv.org/abs/2305.14387.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B. Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators, 2025. URL https://arxiv.org/abs/2404.04475.

Marco Federici, Anjan Dutta, Patrick Forré, Nate Kushman, and Zeynep Akata. Learning robust representations via multi-view information bottleneck. *arXiv preprint arXiv:2002.07017*, 2020.

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

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(25):723–773, 2012. URL http://jmlr.org/papers/v13/gretton12a.html.

He He, Sheng Zha, and Haohan Wang. Unlearn dataset bias in natural language inference by fitting the residual, 2019. URL https://arxiv.org/abs/1908.10763.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding, 2021. URL https://arxiv.org/abs/2009.03300.

R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization, 2019. URL https://arxiv.org/abs/1808.06670.

Zeyu Huang, Zihan Qiu, Zili Wang, Edoardo M. Ponti, and Ivan Titov. Post-hoc reward calibration: A case study on length bias, 2024. URL https://arxiv.org/abs/2409.17407.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension, 2017. URL https://arxiv.org/abs/1705.03551.

Timo Kaufmann, Paul Weng, Viktor Bengs, and Eyke Hüllermeier. A survey of reinforcement learning from human feedback. 2024.

Solomon Kullback. *Information theory and statistics*. Courier Corporation, 1997.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. Race: Large-scale reading comprehension dataset from examinations, 2017. URL https://arxiv.org/abs/1704.04683.

Lauro Langosco, Jack Koch, Lee Sharkey, Jacob Pfau, Laurent Orseau, and David Krueger. Goal misgeneralization in deep reinforcement learning, 2023. URL https://arxiv.org/abs/2105.14111.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E. Gonzalez, and Ion Stoica. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline, 2024. URL https://arxiv.org/abs/2406.11939.

Zhuo Li, Yuege Feng, Dandan Guo, Jinpeng Hu, Anningzhe Gao, and Xiang Wan. Aplot: Robust reward modeling via adaptive preference learning with optimal transport, 2025. URL https://arxiv.org/abs/2510.10963.

Chris Yuhao Liu, Liang Zeng, Jiacai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. Skywork-reward: Bag of tricks for reward modeling in llms, 2024a. URL https://arxiv.org/abs/2410.18451.

Dugang Liu, Pengxiang Cheng, Hong Zhu, Zhenhua Dong, Xiuqiang He, Weike Pan, and Zhong Ming. Debiased representation learning in recommendation via information bottleneck. *ACM Transactions on Recommender Systems*, 1(1):1–27, 2023.

Yantao Liu, Zijun Yao, Rui Min, Yixin Cao, Lei Hou, and Juanzi Li. Rm-bench: Benchmarking reward models of language models with subtlety and style, 2024b. URL https://arxiv.org/abs/2410.16184.

Do Xuan Long, Hai Nguyen Ngoc, Tiviatis Sim, Hieu Dao, Shafiq Joty, Kenji Kawaguchi, Nancy F Chen, and Min-Yen Kan. Llms are biased towards output formats! systematically evaluating and mitigating output format bias of llms. *arXiv preprint arXiv:2408.08656*, 2024.

Yuchun Miao, Sen Zhang, Liang Ding, Rong Bao, Lefei Zhang, and Dacheng Tao. Inform: Mitigating reward hacking in rlhf via information-theoretic reward modeling. *Advances in Neural Information Processing Systems*, 37:134387–134429, 2024.

Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure: Training debiased classifier from biased classifier, 2020. URL https://arxiv.org/abs/2007.02561.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

OpenAI. Gpt-4 technical report, 2024.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022a.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35: 27730–27744, 2022b.

Alexander Pan, Kush Bhatia, and Jacob Steinhardt. The effects of reward misspecification: Mapping and mitigating misaligned models, 2022. URL https://arxiv.org/abs/2201.03544.

Ryan Park, Rafael Rafailov, Stefano Ermon, and Chelsea Finn. Disentangling length from quality in direct preference optimization. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 4998–5017, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.297. URL https://aclanthology.org/2024.findings-acl.297/.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. Instruction tuning with gpt-4. *arXiv preprint arXiv:2304.03277*, 2023.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model, 2024a. URL https://arxiv.org/abs/2305.18290.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024b.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models, 2020. URL https://arxiv.org/abs/1910.02054.

Andrew M Saxe, Yamini Bansal, Joel Dapello, Madhu Advani, Artemy Kolchinsky, Brendan D Tracey, and David D Cox. On the information bottleneck theory of deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124020, 2019.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R Bowman, Newton Cheng, Esin Durmus, Zac Hatfield-Dodds, Scott R Johnston, et al. Towards understanding sycophancy in language models. *arXiv preprint arXiv:2310.13548*, 2023.

Wei Shen, Rui Zheng, Wenyu Zhan, Jun Zhao, Shihan Dou, Tao Gui, Qi Zhang, and Xuanjing Huang. Loose lips sink ships: Mitigating length bias in reinforcement learning from human feedback. *arXiv preprint arXiv:2310.05199*, 2023.

Prasann Singhal, Tanya Goyal, Jiacheng Xu, and Greg Durrett. A long way to go: Investigating length correlations in rlhf. *arXiv preprint arXiv:2310.03716*, 2023.

Joar Skalse, Nikolaus H. R. Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward hacking, 2025. URL https://arxiv.org/abs/2209.13085.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. Challenging big-bench tasks and whether chain-of-thought can solve them, 2022. URL https://arxiv.org/abs/2210.09261.

Enzo Tartaglione, Carlo Alberto Barbano, and Marco Grangetto. End: Entangling and disentangling deep representations for bias correction. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13503–13512. IEEE, June 2021. doi: 10.1109/cvpr46437.2021.01330. URL http://dx.doi.org/10.1109/CVPR46437.2021.01330.

Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method, 2000. URL https://arxiv.org/abs/physics/0004057.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

Zhibin Wan, Changqing Zhang, Pengfei Zhu, and Qinghua Hu. Multi-view information-bottleneck representation learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 10085–10092, 2021.

Chaoqi Wang, Zhuokai Zhao, Yibo Jiang, Zhaorun Chen, Chen Zhu, Yuxin Chen, Jiayi Liu, Lizhu Zhang, Xiangjun Fan, Hao Ma, et al. Beyond reward hacking: Causal rewards for large language model alignment. *arXiv preprint arXiv:2501.09620*, 2025a.

Haoxiang Wang, Wei Xiong, Tengyang Xie, Han Zhao, and Tong Zhang. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 10582–10592, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.620. URL https://aclanthology.org/2024.findings-emnlp.620/.

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

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Siyang Yuan, Pengyu Cheng, Ruiyi Zhang, Weituo Hao, Zhe Gan, and Lawrence Carin. Improving zero-shot voice style transfer via disentangled representation learning. In *International Conference on Learning Representations*, 2021.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence?, 2019. URL https://arxiv.org/abs/1905.07830.

Xuanchang Zhang, Wei Xiong, Lichang Chen, Tianyi Zhou, Heng Huang, and Tong Zhang. From lists to emojis: How format bias affects model alignment. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 26940–26961, Vienna,

Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1308. URL https://aclanthology.org/2025.acl-long.1308/.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Processbench: Identifying process errors in mathematical reasoning, 2025. URL https://arxiv.org/abs/2412.06559.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023. URL https://arxiv.org/abs/2306.05685.

Fan Zhou, Yuzhou Mao, Liu Yu, Yi Yang, and Ting Zhong. Causal-debias: Unifying debiasing in pretrained language models and fine-tuning via causal invariant learning. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4227–4241, Toronto, Canada, July 2023a. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.232. URL https://aclanthology.org/2023.acl-long.232/.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models, 2023b. URL https://arxiv.org/abs/2311.07911.

A USAGE OF LLMS

In the preparation of this paper, we utilized Large Language Models (LLMs) solely for the purpose of grammatical polishing and text refinement of the manuscript content. Specifically, the LLMs were only used to optimize the clarity, fluency, and grammatical accuracy of the written text.

All content polished by LLMs underwent thorough manual review and verification by the authors. We carefully checked the polished text to ensure its consistency with the original research intent, accuracy of scientific facts, and compliance with academic integrity standards. We confirm that we take full responsibility for all contents of the paper under our names, including the parts that underwent LLM-assisted grammatical polishing.

B BOUND PROOF

B.1 PROOF OF THE BARBER-AGAKOV (BA) BOUND.

The goal is to prove that for any variational distribution $q_\theta(\boldsymbol{y}|\boldsymbol{x})$, the mutual information $I(\boldsymbol{x};\boldsymbol{y})$ is lower-bounded by $\mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log q_\theta(\boldsymbol{y}|\boldsymbol{x})] + H(\boldsymbol{y})$, for any two random variables $\boldsymbol{x}$ and $\boldsymbol{y}$. We begin with the definition of mutual information:

$$I(\boldsymbol{x};\boldsymbol{y}) = H(\boldsymbol{y}) - H(\boldsymbol{y}|\boldsymbol{x}),$$

where $H(\boldsymbol{y})$ is the marginal entropy of $\boldsymbol{y}$, and $H(\boldsymbol{y}|\boldsymbol{x})$ is the conditional entropy. The conditional entropy is defined as:

$$H(\boldsymbol{y}|\boldsymbol{x}) = -\mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log p(\boldsymbol{y}|\boldsymbol{x})].$$

Substituting this into the definition of MI, we get:

$$\begin{aligned} I(\boldsymbol{x};\boldsymbol{y}) &= H(\boldsymbol{y}) - (-\mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log p(\boldsymbol{y}|\boldsymbol{x})]) \\ &= H(\boldsymbol{y}) + \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log p(\boldsymbol{y}|\boldsymbol{x})]. \end{aligned} \tag{13}$$

Now, we introduce the variational approximation $q_\theta(\boldsymbol{y}|\boldsymbol{x})$ by considering the Kullback-Leibler (KL) divergence between the true conditional distribution $p(\boldsymbol{y}|\boldsymbol{x})$ and our approximation $q_\theta(\boldsymbol{y}|\boldsymbol{x})$, averaged over all $\boldsymbol{x} \sim p(\boldsymbol{x})$:

$$\mathbb{E}_{p(\boldsymbol{x})}[\mathrm{KL}(p(\boldsymbol{y}|\boldsymbol{x}) \,||\, q_\theta(\boldsymbol{y}|\boldsymbol{x}))] \geq 0.$$

By expanding the definition of KL divergence, we have:

$$\begin{aligned} 0 &\leq \mathbb{E}_{p(\boldsymbol{x})}\left[\sum_{\boldsymbol{y}} p(\boldsymbol{y}|\boldsymbol{x}) \log \frac{p(\boldsymbol{y}|\boldsymbol{x})}{q_\theta(\boldsymbol{y}|\boldsymbol{x})}\right], \\ 0 &\leq \sum_{\boldsymbol{x}} p(\boldsymbol{x}) \sum_{\boldsymbol{y}} p(\boldsymbol{y}|\boldsymbol{x}) \log \frac{p(\boldsymbol{y}|\boldsymbol{x})}{q_\theta(\boldsymbol{y}|\boldsymbol{x})}, \\ 0 &\leq \sum_{\boldsymbol{x},\boldsymbol{y}} p(\boldsymbol{x},\boldsymbol{y}) \left(\log p(\boldsymbol{y}|\boldsymbol{x}) - \log q_\theta(\boldsymbol{y}|\boldsymbol{x})\right), \\ 0 &\leq \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log p(\boldsymbol{y}|\boldsymbol{x})] - \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log q_\theta(\boldsymbol{y}|\boldsymbol{x})]. \end{aligned}$$

Rearranging this inequality gives us a lower bound for the expected log-likelihood under the true distribution:

$$\mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log p(\boldsymbol{y}|\boldsymbol{x})] \geq \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log q_\theta(\boldsymbol{y}|\boldsymbol{x})]. \tag{14}$$

Finally, by substituting this inequality equation 14 back into our expanded definition of mutual information equation 13, we obtain the Barber-Agakov bound:

$$\begin{aligned} I(\boldsymbol{x};\boldsymbol{y}) &= H(\boldsymbol{y}) + \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log p(\boldsymbol{y}|\boldsymbol{x})] \\ &\geq H(\boldsymbol{y}) + \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log q_\theta(\boldsymbol{y}|\boldsymbol{x})]. \end{aligned}$$

Let $H[p]$ to represent the marginal entropy $H(\boldsymbol{y})$, we arrive at the final expression:

$$I(\boldsymbol{x};\boldsymbol{y}) \geq \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log q_\theta(\boldsymbol{y}|\boldsymbol{x})] + H[p] =: I_{\mathrm{BA}}(\boldsymbol{x};\boldsymbol{y}),$$

which completes the proof. The bound becomes tight (i.e., the inequality becomes an equality) if and only if the variational approximation perfectly matches the true conditional distribution, $q_\theta(\boldsymbol{y}|\boldsymbol{x}) = p(\boldsymbol{y}|\boldsymbol{x})$ for all $\boldsymbol{x}, \boldsymbol{y}$.

B.2 Proof of the CLUB Upper Bound

We aim to prove that for any variational distribution $q_\theta(\boldsymbol{y}|\boldsymbol{x})$, the mutual information $I(\boldsymbol{x};\boldsymbol{y})$ is upper-bounded by $I_{\text{CLUB}}(\boldsymbol{x};\boldsymbol{y})$. We begin with the definition of mutual information:

$$I(\boldsymbol{x};\boldsymbol{y}) = \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}\left[\log p(\boldsymbol{y}|\boldsymbol{x})\right] - \mathbb{E}_{p(\boldsymbol{y})}\left[\log p(\boldsymbol{y})\right] \tag{15}$$

Let's focus on the second term, which is the negative marginal entropy $+H(\boldsymbol{y})$. We can express the marginal distribution $p(\boldsymbol{y})$ by marginalizing out $\boldsymbol{x}$:

$$p(\boldsymbol{y}) = \mathbb{E}_{p(\boldsymbol{x}')}[p(\boldsymbol{y}|\boldsymbol{x}')]$$

where $\boldsymbol{x}'$ is a random variable drawn from the same distribution as $\boldsymbol{x}$, but is independent of the $\boldsymbol{x}$ in the first term of equation 15. Substituting this into the entropy term:

$$-\mathbb{E}_{p(\boldsymbol{y})}\left[\log p(\boldsymbol{y})\right] = -\mathbb{E}_{p(\boldsymbol{y})}\left[\log \mathbb{E}_{p(\boldsymbol{x}')}[p(\boldsymbol{y}|\boldsymbol{x}')]\right].$$

Since the logarithm is a concave function, we can apply Jensen's inequality, which states that $\mathbb{E}[\log(Z)] \leq \log(\mathbb{E}[Z])$. This implies $-\log(\mathbb{E}[Z]) \leq -\mathbb{E}[\log(Z)]$. Applying this, we get:

$$\begin{aligned} -\mathbb{E}_{p(\boldsymbol{y})}\left[\log \mathbb{E}_{p(\boldsymbol{x}')}[p(\boldsymbol{y}|\boldsymbol{x}')]\right] &\leq -\mathbb{E}_{p(\boldsymbol{y})}\left[\mathbb{E}_{p(\boldsymbol{x}')}[\log p(\boldsymbol{y}|\boldsymbol{x}')]\right] \\ &= -\mathbb{E}_{p(\boldsymbol{x}')p(\boldsymbol{y})}[\log p(\boldsymbol{y}|\boldsymbol{x}')]. \end{aligned}$$

Now, substituting this inequality back into our original MI expression equation 15, we obtain an upper bound on the mutual information:

$$I(\boldsymbol{x};\boldsymbol{y}) \leq \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log p(\boldsymbol{y}|\boldsymbol{x})] - \mathbb{E}_{p(\boldsymbol{x})p(\boldsymbol{y})}[\log p(\boldsymbol{y}|\boldsymbol{x})]. \tag{16}$$

Note that the second expectation is over the product of marginals $p(\boldsymbol{x})p(\boldsymbol{y})$. The inequality equation 16 holds for the true conditional distribution $p(\boldsymbol{y}|\boldsymbol{x})$. The CLUB bound replaces $p(\boldsymbol{y}|\boldsymbol{x})$ with the variational approximation $q_\theta(\boldsymbol{y}|\boldsymbol{x})$. The key insight from Cheng et al. (2020) is that the difference between the true bound and the variational bound is an expectation of KL-divergences, and this variational form serves as a practical, sample-based upper bound for minimization. Therefore, we use the variational form as our tractable objective:

$$I(\boldsymbol{x};\boldsymbol{y}) \leq \mathbb{E}_{p(\boldsymbol{x},\boldsymbol{y})}[\log q_\theta(\boldsymbol{y}|\boldsymbol{x})] - \mathbb{E}_{p(\boldsymbol{x})p(\boldsymbol{y})}[\log q_\theta(\boldsymbol{y}|\boldsymbol{x})] =: I_{\text{CLUB}}(\boldsymbol{x};\boldsymbol{y}).$$

This completes the justification for using I_{CLUB} as an upper bound for mutual information minimization.

C Implementation Details of q_ψ

Our variational network q_ψ for estimating the mutual information is implemented as a lightweight two-layer Multi-Layer Perceptron (MLP). Its architecture is as follows:

```
self.variational_net = nn.Sequential(
    nn.Linear(input_dim, hidden_size),
    nn.ReLU(),
    nn.Linear(hidden_size, label_num)
)
```

The dimensions are chosen based on the following principles: input_dim: This is dynamically set to match the dimension of the final hidden state representation from the backbone LLM. For instance, in our experiments with Llama-3.1-8B-Instruct, the input_dim is 4096.

label_num: The output dimension is set to 2. This is a direct and necessary consequence of our theoretical formulation, as the relative bias attribute $\boldsymbol{b}_{\text{rel}}$ is defined as a binary variable (0, 1) indicating which of the two responses in a pair exhibits a stronger bias.

hidden_size: For the intermediate hidden layer size, we conducted an ablation study over the values [512, 1024, 2048]. We observed that hidden_size=1024 offered the best trade-off between debiasing performance and minimal computational overhead.

We intentionally designed q_ψ to be a simple and efficient network. This ensures that the observed performance gains are attributable to our method itself, rather than to the introduction of a large number of additional parameters.

D VISUALIZATION OF DIR

We visualize our DIR framework as shown in Figure 3.

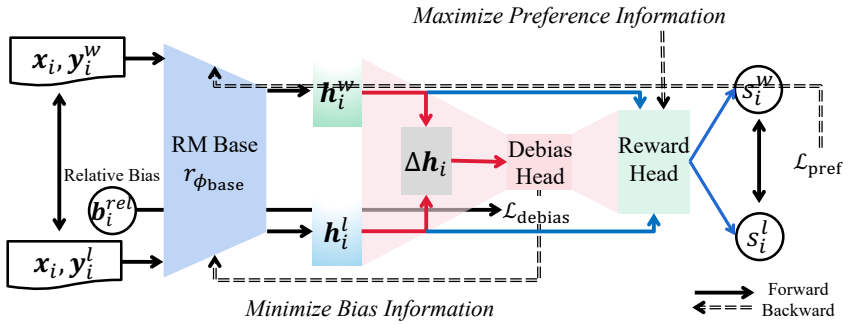

Figure 3: An overview of our DIR framework, which disentangles reward modeling into two competing pathways. A Preference Path is optimized to maximize mutual information with the true preference label. Concurrently, a Debias Path, regularized by an information bottleneck, is optimized to minimize mutual information with the bias attributes. This dual-objective forces the shared RM Base to learn representations that are sensitive to preference but invariant to bias.

E EXPERIMENT

E.1 LENGTH BIAS

Dataset, and Model. We train reward models on Skywork-Preference-80K-v0.2 (SK) dataset[4] based on Llama3.1-8B-Instruct. With the reward model, we then train Llama3.1-8B-Instruct and OpenRLHF-Llama3-8B-SFT polices with the PPO implementation for one epoch.

Training Settings. For our reward model training, we adopt a full parameter tuning strategy by using HuggingFace Trainer with DeepSpeed Zero1 on 8 GPU cards. Global batch size is set to 128, initialization learning rate is 2e-6 with Cosine scheduler. For our PPO experiment, we fine-tune two distinct models using 20,000 samples from the alpaca-gpt4-data-en dataset (Peng et al., 2023). The first model, Llama3.1-8B-Instruct[5], has undergone post-training that includes both DPO and RLHF. The second, OpenRLHF-Llama3-8B-SFT[6], is an instruction-following version built upon Llama3-8B-Base, without the RLHF post-training stage. We conduct the PPO training using the ms-swift framework[7] with its default training configuration.

Baselines. We mainly consider the following baselines due to the reproducibility: 1) Vanilla BT Baseline and popular open-source RM Skywork-Reward-Llama-3.1-8B-v0.2[8]; 2) Length Debiased RMs, including PoE (Shen et al., 2023) and ALBM (Bu et al., 2025); 3) Length Penalty that directly resharps the reward during PPO by $\tilde{r}(\boldsymbol{x}, \boldsymbol{y}) = r(\boldsymbol{x}, \boldsymbol{y}) - 0.001 * len(\boldsymbol{y})$ (Dong et al., 2024); 4) InfoRM (Miao et al., 2024) that is also designed from the information theory perspective.

Evaluations. All benchmark evaluations are subsequently performed using the ms-evalscope framework[9]. Our evaluation protocol utlize few-shot settings for GSM8K (4-shot) (Cobbe et al., 2021), Race (3-shot) (Lai et al., 2017), and TriviaQA (5-shot) (Joshi et al., 2017), while all other

[4] https://huggingface.co/datasets/Skywork/Skywork-Reward-Preference-80K-v0.2

[5] https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct

[6] https://huggingface.co/OpenRLHF/Llama-3-8b-sft-mixture

[7] https://github.com/modelscope/ms-swift

[8] https://huggingface.co/Skywork/Skywork-Reward-Llama-3.1-8B-v0.2

[9] https://github.com/modelscope/evalscope

benchmarks (i.e., Hellaswag (Zellers et al., 2019), IFeval (Zhou et al., 2023b), MMLU (Hendrycks et al., 2021), ProcessBench (Zheng et al., 2025), BBH (Suzgun et al., 2022), and Humaneval (Chen et al., 2021)) are assessed in a zero-shot setting. We report accuracy as the primary metric for all tasks, with the exception of Humaneval, for which we report the Pass@1 score.

Performance on RM-Bench. We further evaluate our debiased reward models on the RM-Bench, which assesses capabilities across various domains (Chat, Math, Code, Safety) and difficulty levels (Hard, Normal, Easy). The results, presented in Table 7, demonstrate that our DIR framework outperforms several baseline methods in terms of overall performance.

Our primary model, Ours-1.0, which corresponds to the optimal trade-off point ($\lambda = 1.0$) identified in our ablation study, achieves the second-highest total score (69.35). It exhibits a well-balanced profile, securing the top performance on the 'Math' subset (61.81) and the 'Normal' difficulty subset (73.59), while remaining highly competitive in 'Chat' (68.91). This confirms that our method can enhance the reward model's core capabilities without compromising its general performance.

When we increase the debiasing strength to $\lambda = 10.0$, the Ours-10.0 model achieves the best overall performance (70.18). The most significant improvement is observed on the 'Hard' subset, where our model's score dramatically jumps to 64.41, surpassing the next-best method by a large margin of over 16 points. This strongly suggests that by forcing the model to ignore superficial format cues, DIR enables it to focus on the more subtle and complex signals of quality inherent in difficult prompts. This specialized model also secures the top rank in the 'Chat' and 'Code' domains. However, this specialization comes at the cost of performance on the 'Easy' subset, where simpler heuristics might be sufficient and our strong debiasing may be overly restrictive.

In summary, these results demonstrate that DIR not only enhances the overall capability of the reward model but also offers a tunable mechanism to prioritize robustness on challenging tasks over simpler ones, showcasing the flexibility and effectiveness of our approach.

Table 7: Performance comparison on RM-Bench. Best results are in **bold** and Second-performance is in underlined. We train the BT baseline in our own codebase.

Method	Chat	Math	Code	Safety	Hard	Normal	Easy	Total
BT	64.69	61.21	51.41	95.11	42.76	72.30	89.24	68.10
PoE	67.70	61.23	51.51	**95.51**	44.94	73.17	88.86	68.99
ALBM	64.57	58.48	52.34	95.21	47.88	71.50	**90.32**	67.40
Ours-1.0	68.91	**61.81**	51.56	95.13	47.88	**73.59**	88.93	69.35
Ours-10.0	**71.23**	61.59	**52.73**	94.91	**64.41**	71.29	74.85	**70.18**

Performance on MT-Bench and AlpaceEval. For MT-Bench (Zheng et al., 2023), we report the win rate of each RM-guided policy against its own base model, using the standard MT-Bench LLM-as-a-judge setup. As shown in Table 8, our method ("OURS") achieves the highest win rates on both backbones (56.25% vs. 48.75–53.75% for OpenRLHF-Llama-3-8B-SFT, and 56.88% vs. 50.63–51.88% for Meta-Llama3.1-8B-Instruct), indicating more improvements on open-ended, multi-turn dialogue quality.

Table 8: Win rate (%) performance comparison on MT-Bench.

Win Rate (%) (vs. Base)	Base Model	
	OpenRLHF-Llama-3-8B-SFT	Meta-Llama3.1-8B-Instruct
OURS	**56.25**	**56.88**
PoE	48.75	51.25
Skywork	49.38	51.25
ALBM	53.75	50.63
InfoRM	46.88	51.88

For Length Controlled AlpacaEval, we follow the length-controlled protocol of Dubois et al. (2025) and report both raw win rate and length-controlled win rate over the base model. On Meta-Llama3.1-8B-Instruct, OURS achieves the highest scores on both metrics. On OpenRLHF-Llama-3-8B-SFT, Skywork attains a slightly higher raw win rate, but OURS achieves the best length-controlled win rate. This pattern is consistent with our goal: once the confounding effect of response length is controlled for, our debiased RMs yield policies that are preferred more often, demonstrating better alignment that is not driven by verbosity. We will include these MT-Bench and AlpacaEval results and their analysis in the revised version.

Table 9: Win rate (%) performance comparison on Length Controlled AlpacaEval against *gpt4_1106_preview*.

Base Model: Meta Llama3.1-8B-Instruct		
Methods	Raw Win Rate (%)	Length Control Win Rate (%)
OURS	**31.30**	**19.66**
PoE	26.58	11.41
Skywork	29.38	13.21
ALBM	26.83	10.61
InfoRM	25.22	11.02
Base Model: OpenRLHF Llama-3-8B-SFT		
Methods	Raw Win Rate (%)	Length Control Win Rate (%)
OURS	9.50	**5.46**
PoE	7.14	3.28
Skywork	**10.19**	3.93
ALBM	8.88	5.08
InfoRM	5.84	3.65

PPO Training Monitoring. Figure 4 presents three key metrics for monitoring the PPO training process. The left plot (RLHF Reward) evaluates the final quality score of the model's outputs, with higher values being better. The middle plot (KL Divergence) measures how much the learned policy has deviated from the initial reference model, indicating the extent of exploration. The right plot (Approx. KL) shows the magnitude of each policy update, serving as a critical indicator of training stability. Our policy model demonstrates a better balance across these metrics by achieving a top reward score that significantly outperforms all baselines. Concurrently, our KL divergence is maintained at a moderate level, suggesting effective exploration without catastrophic deviation from the base model's capabilities. Most importantly, our method exhibits the lowest and most stable Approx. KL, which proves that the training process is exceptionally smooth and reliable. In summary, our approach successfully boosts performance while ensuring unparalleled training stability.

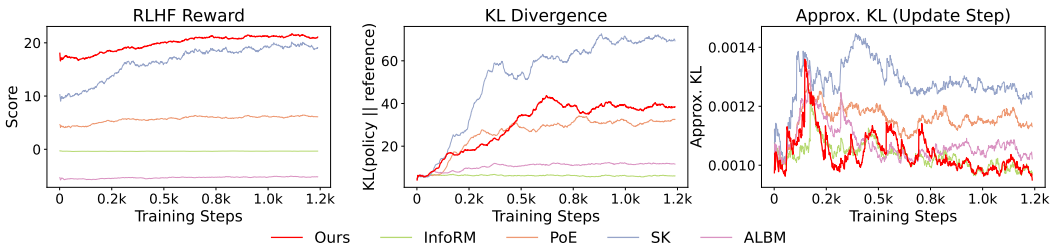

Figure 4: PPO training dynamics across key metrics. Our RM obtains a higher policy score and demonstrates better training stability.

E.2 Ablation Studies under Length Debias

Ablation Study on Representation for Debiasing. A core design choice in our framework is the use of representation difference ($\Delta \boldsymbol{h} = \boldsymbol{h}^w - \boldsymbol{h}^l$) as input to the variational network, rather than representation concatenation ($[\boldsymbol{h}^w; \boldsymbol{h}^l]$). We conduct an ablation study to validate this choice, evaluating both approaches on the RewardBench-v1 and RM-Bench benchmark suites. As detailed in Table 10, our empirical results strongly support the effectiveness of using representation difference.

Theoretically, this choice is motivated by two factors. (1) Alignment with Preference Learning: The Bradley-Terry objective itself operates on the *difference* of reward scores. By feeding the representation *difference* to the debiasing module, we align the supervisory signal for debiasing with the primary learning objective. (2) Signal Purity: The difference operator effectively cancels out redundant information from the shared prompt $\boldsymbol{x}$, forcing the debiasing network q_ψ to focus exclusively on the features that distinguish $\boldsymbol{y}^w$ from $\boldsymbol{y}^l$.

Our experiments confirm these theoretical advantages. The difference-based method shows notable performance gains across a wide range of capabilities, particularly in conversational and reasoning tasks. For instance, on RewardBench-v1, our approach improves performance on the challenging 'Chat Hard' subset from 78.9% to 83.6% and on 'Reasoning' from 88.8% to 90.0%. Similar gains are observed on RM-Bench, where the 'chat' score increases from 63.9% to 66.8%. While performance on other sub-categories remains largely comparable, the overall trend indicates a clear advantage for the difference-based approach.

Beyond performance, the difference operator offers practical benefits. Using concatenation doubles the input dimension to the variational network q_ψ (i.e., from embedding size to embedding size $\times$ 2). This not only increases the number of parameters and computational complexity for the debiasing module but also leads to a slightly higher GPU memory footprint during training. Therefore, we conclude that using representation difference is more effective both in principle and in practice, and we adopt it as the default setting for our DIR framework.

Table 10: Ablation study on the representation format for the debiasing module. We report accuracy (%) on RewardBench-v1 and RM-Bench. The difference-based approach consistently outperforms concatenation, especially on challenging conversational and reasoning tasks. Best results are in **bold**.

	RewardBench-v1 (Acc %)				RM-Bench (Acc %)			
Method	Chat	Chat Hard	Safety	Reasoning	Chat	Math	Code	Safety
Concat ($[\boldsymbol{h}^w; \boldsymbol{h}^l]$)	93.3	78.9	**90.9**	88.8	65.9	60.8	**52.6**	95.0
Difference ($\Delta \boldsymbol{h}$)	**94.1**	**83.6**	89.7	**90.0**	**67.8**	**61.1**	52.4	**95.2**

Ablation Study on Debiasing Coefficient λ. The hyperparameter λ in Equation 17 governs the trade-off between the standard preference learning objective ($\mathcal{L}_{\text{pref}}$) and our information-theoretic debiasing objective ($\mathcal{L}_{\text{debias}}$). To analyze its sensitivity, we tested a range of values: $\{0.1, 0.3, 0.5, 1, 2, 5, 10\}$. The results, visualized in Figure 5, reveal a clear trade-off.

As shown in the figure, when λ is too small (e.g., 0.1), the debiasing signal is insufficient. The model behaves similarly to a standard BT model, exhibiting a high bias metric (e.g., high Pearson correlation with a bias attribute) while achieving good performance on RewardBench. Conversely, when λ is too large (e.g., 10), the debiasing objective dominates the training. This "over-correction" successfully minimizes the bias but severely compromises the model's ability to learn true preference signals, leading to a significant drop in RewardBench accuracy. We observe that $\lambda = 1$ strikes an optimal balance. At this value, the bias metric is substantially reduced, while the preference learning performance on RewardBench is maximized. This indicates that our method can effectively neutralize spurious correlations without damaging, and in fact enhancing, the reward model's core capabilities. Therefore, we use $\lambda = 1$ for all main experiments in this paper.

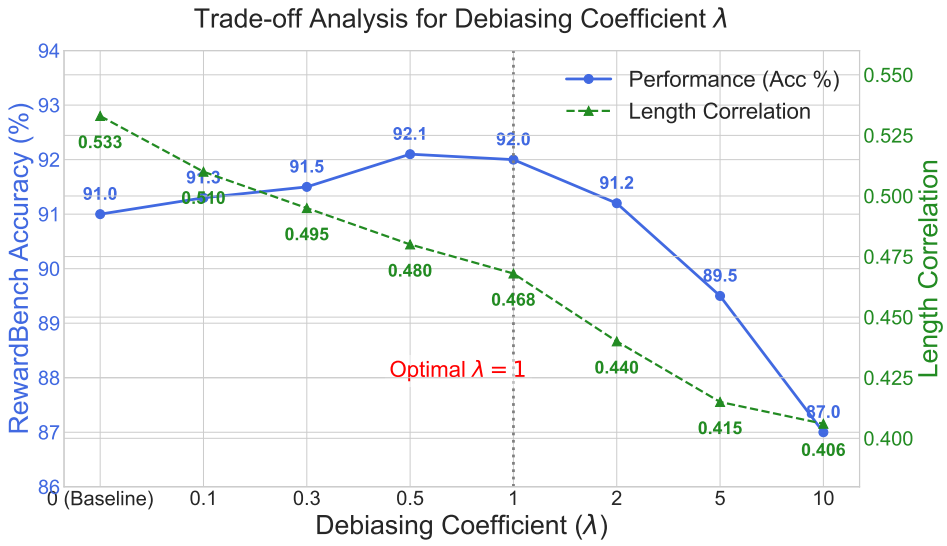

Figure 5: Ablation study on the debiasing coefficient λ. The plot shows the trade-off between preference learning performance (RewardBench Accuracy, blue) and the bias metric (e.g., Pearson r, green). $\lambda = 1$ achieves the best balance.

E.3 Experiment on DPO

Specifically, we adopt ms-swift framework with its default DPO training configuration on Human-Like-DPO-Dataset[10], based on both OpenRLHF-Llama-3-8b-SFT and Meta-Llama3.1-8B-Instruct models, where DPO $\beta = 0.1$, debias factor $\lambda = 1$. We train 1 epoch and evaluate the performance on the final checkpoint. Human-Like-DPO-Dataset is created to fine-tune LLMs toward generating more human-like responses, which includes 10,884 samples across 256 topics, containing technology, daily Life, science, history and arts. We evaluate the performance on ArenaHard-v0.1 and several popular benchmarks. For baselines, we also compare with the length-controlled DPO method Park et al. (2024), which disentangles the length from the quality to explicitly avoid the policy model from preferring the longer response DPO training.

E.4 Sycophancy Bias

Dataset and Model Motivated by Sharma et al. (2023); Wang et al. (2025a), we create a semi-sycophantic dataset by partially contaminating the HelpSteer3 dataset (Wang et al., 2025b). Specifically, we artificially inject a sycophantic prefix (i.e., "Yes, you are right.") into a proportion γ (e.g., $\gamma = 40\%$) of responses in the training dataset. Within this contaminated subset, the prefix is added to the chosen response with an α probability (e.g., $\alpha = 70\%$) and to the rejected response with a $1 - \alpha = 20\%$ probability. The remaining $1 - \gamma = 30\%$ of the dataset is left unchanged without the sycophancy. This process creates a challenging, mixed-distribution environment where the sycophantic phrase acts as a strong but unreliable reward signal. Reward models are still built upon the Llama-3.1-8B-Instruct backbone.

Training Settings. For reward model training, we adopt a full parameter tuning strategy by using HuggingFace Trainer with DeepSpeed Zero1 on 8 GPU cards. Global batch size is set to 128, initialization learning rate is 2e-6 with Cosine scheduler.

Baselines. Since other debiasing methods are either mainly designed for length bias (e.g., PoE, ALBM, and Length-Penalty) or are not open-sourced (e.g., CRM), we primarily compare our method against two key baselines: a standard BT reward model and InfoRM.

[10] https://huggingface.co/datasets/HumanLLMs/Human-Like-DPO-Dataset

E.5 FORMAT BIAS

Dataset and Model Following the data construction in LE (Zhang et al., 2025), we construct a format-biased dataset for our experiments. We start with a clean base preference dataset of 71.6K pairs, which is created by filtering the UltraFeedback dataset (Cui et al., 2024) to include only pairs with a score difference greater than 1.0. To inject format bias, this clean dataset is then "attacked" by mixing in a small, artificially generated biased dataset. Specifically, we inject 0.7% training data where a 'bold' formatted response is spuriously labeled as preferred over its identical, unformatted counterpart, and 1.4% data where a 'list' formatted response is similarly favored. The final reward model training is conducted on this combined, biased dataset. The base model for our reward model is Llama-3-8B-Instruct.

Training Settings. For reward model training, we adopt a full parameter tuning strategy by using HuggingFace Trainer with DeepSpeed Zero1 on 8 GPU cards. Global batch size is set to 128, initialization learning rate is 2e-6 with Cosine scheduler.

Baselines. By following the experimental setting of Zhang et al. (2025), we mainly consider standard BT, BT with deleted specific format training data (BT†), and LE (Zhang et al., 2025).

E.6 CONCURRENT MULTI-BIAS

Concretely, we extend DIR to length and sycophancy biases by introducing two independent Mutual-Information regularizers, each with its own q_ψ head. For each preference pair, we construct $\boldsymbol{b}_{\text{rel}}^{\text{len}}$ (which response is longer) and $\boldsymbol{b}_{\text{rel}}^{\text{syco}}$ (which response is more sycophantic), and optimize:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{pref}} + \lambda_{\text{len}}\mathcal{L}_{\text{debias\_len}} + \lambda_{\text{syco}}\mathcal{L}_{\text{debias\_syco}}, \tag{17}$$

where $\lambda_{\text{len}} = \lambda_{\text{syco}} = 1$. We follow the Table 5 setting, training Meta-Llama-3.1-8B-Instruct on the HelpSteer3 dataset under two sycophancy contamination configurations ($\gamma = 40\%/80\%, \alpha = 70\%$), where a larger γ indicates a more challenging setting. All models (BT, length-only DIR, and length+syco DIR) are trained for 1 epoch on the same data, and we report results using the final checkpoint for fairness and convenience.

As shown in Table 11, on RM-Bench, the joint model (OURS-Len-Syco) achieves the best overall performance and the largest gains on the hardest subset (e.g., at $\gamma = 40\%$, Total: $67.25 \rightarrow 69.96$, Hard: $39.88 \rightarrow 46.85$ vs. BT), while still clearly reducing the Pearson correlation with length relative to BT, confirming that length bias is mitigated even when sycophancy is also debiased. We also observe that the length-only model (OURS-Len) attains the lowest length–reward Pearson coefficient, but somewhat surprisingly, the joint length+sycophancy debiasing (OURS-Len-Syco) yields the best overall RM-Bench performance, suggesting that debiasing multiple biases together may help lead to a more balanced and effective reward model.

Table 11: Performance comparison of concurrent multi-bias experiments on RM-Bench. Best results are in **bold**.

$\gamma = 40\%, \alpha = 70\%$	Chat	Math	Code	Safety	Hard	Normal	Easy	Total	Pearson Coefficient
BT	66.58	64.17	53.70	84.53	39.88	72.82	**89.04**	67.25	0.4807
OURS-Len	66.93	64.59	53.12	**89.14**	44.25	72.43	88.65	68.44	**0.4235**
OURS-Len-Syco	**70.80**	**65.07**	**55.26**	88.71	**46.85**	**75.08**	87.96	**69.96**	0.4446
$\gamma = 80\%, \alpha = 70\%$	Chat	Math	Code	Safety	Hard	Normal	Easy	Total	Pearson Coefficient
BT	65.37	63.71	53.12	80.85	37.58	70.96	**88.75**	65.76	0.4666
OURS-Len	**68.91**	64.25	53.75	87.23	44.78	**73.09**	87.72	68.53	**0.4081**
OURS-Len-Syco	68.39	**64.92**	**54.04**	**88.06**	**45.15**	72.92	88.50	**68.85**	0.4235

As shown in Table 12, on sycophancy stress tests, debiasing only sycophancy (OURS-Syco) gives the strongest syco robustness, as expected, but the joint model (ours-Len-Syco) still substantially outperforms BT on all sycophancy metrics (All./Nat./Adv.) across both γ settings, while additionally reducing length bias. In summary, a second debiasing term leads to a controlled trade-off, not conflicting gradients: both biases are improved over BT, and overall RM quality remains strong.

Table 12: Reward model accuracy (%) on the concurrent multi-bias under varying contamination settings.

		All.			Nat.			Adv.		
γ	α	BT	OURS-Syco	ours-Len-Syco	BT	OURS-Syco	ours-Len-Syco	BT	OURS-Syco	ours-Len-Syco
40%	70%	83.6	88.0	85.6	84.4	87.4	86.4	82.6	88.6	84.4
80%	70%	81.2	86.2	85.9	86.4	87.2	86.6	79.7	86.2	85.1

F PROMPT-BASED JUSTIFICATION PROMPT

In this section, we give a Qwen3-235B-A22B-based pair-wise justification prompt shown below, which is adopted from ArenaHard's official implementation [11].

Please act as an impartial judge and evaluate the quality of the responses provided by two AI assistants to the user prompt displayed below. You will be given assistant A's answer and assistant B's answer. Your job is to evaluate which assistant's answer is better. Begin your evaluation by generating your own answer to the prompt. You must provide your answers before judging any answers. When evaluating the assistants' answers, compare both assistants' answers with your answer. You must identify and correct any mistakes or inaccurate information. Then consider if the assistant's answers are helpful, relevant, and concise. Helpful means the answer correctly responds to the prompt or follows the instructions. Note when user prompt has any ambiguity or more than one interpretation, it is more helpful and appropriate to ask for clarifications or more information from the user than providing an answer based on assumptions. Relevant means all parts of the response closely connect or are appropriate to what is being asked. Concise means the response is clear and not verbose or excessive. Then consider the creativity and novelty of the assistant's answers when needed. Finally, identify any missing important information in the assistants' answers that would be beneficial to include when responding to the user prompt. After providing your explanation, you must output only one of the following choices as your final verdict with a label:

1. Assistant A is significantly better: [[A >> B]]
2. Assistant A is slightly better: [[A>B]]
3. Tie, relatively the same: [[A=B]]
4. Assistant B is slightly better: [[B>A]]
5. Assistant B is significantly better: [[B>>A]]
Example output: "My final verdict is tie: [[A=B]]"

[User Prompt]

[Assistant A's Answer]
{*answer A*}

[Assistant B's Answer]
{*answer B*}

G CASE STUDY

In this case study, our model demonstrates its ability to generate not just a correct, but a more professionally helpful response compared to strong baselines like GPT-4o. As visualized in Figure 6 (Data UID: `95bb5011680c4ba281ad7830ffa57f92`), when asked how to set up passwordless SSH, our model provided a guide that was praised by the LLM-based judge for its clear structure and organization. In this instance, GPT-4o's response is judged to be better to all other baselines except for our response, which provides a more secure, modern, and concise solution, highlighting its advanced capabilities.

[11] `https://github.com/lmarena/arena-hard-auto`

Crucially, the judge noted that our model successfully avoids unnecessary verbosity, resulting in a response that is of higher quality and more concise. This is not merely a stylistic advantage; the substance of our model's response was also technically useful. It recommended the more modern and secure "ed25519" encryption algorithm over the RSA suggested by GPT-4o. Furthermore, our model included the critical security best practice of disabling password authentication entirely via the sshd_config file—a vital step for server hardening that GPT-4o and other models omitted. This case exemplifies our model's strength in producing responses that are not only accurate but also more secure, efficient, and pragmatically useful for real-world technical tasks.

Prompt: How do I set up ssh into a server such that I don't require a password when I ssh into it?

ALBM (2486 chars): Setting up Passwordless SSH**\n\nTo set up passwordless SSH, you'll need to generate a pair of SSH keys on your local machine and add the public key to the server. This way, you can authenticate without entering a password.\n\n### Step 1: Generate SSH Keys on Your Local Machine\n\nOpen a terminal on your local machine and run the following command:\n\n```bash\nssh-keygen -t rsa -b 4096\n```\n\nThis will generate a pair of SSH keys in the `~/.ssh` directory.\n\n### Step 2: Copy the Public Key to the Server\n\nYou'll need to copy the public key to the server. You can do this by running the following command:\n\n```bash\nssh-copy-id user@server\n```\n\nReplace `user` with your username on the server and `server` with the hostname or IP address of the server.\n\n### Step 3: Verify Passwordless SSH\n\nTry connecting to the server using SSH without a password:\n\n```bash\nssh user@server\n```\n\nIf everything is set up correctly, you should be able to connect to the server without entering a password.\n\n### Step 4: Add the Public Key to the Server (Alternative Method)\n\nIf you're using an older version of SSH or don't have `ssh-copy-id` available, you can add the public key manually. Run the following commands on your local machine:\n\n```bash\nssh-keygen -t rsa -b 4096\ncat ~/.ssh/id_rsa.pub | ssh user@server \"mkdir -p ~/.ssh; cat >> ~/.ssh/authorized_keys\"\n```\n\nThis will create the `~/.ssh` directory on the server if it doesn't exist and append the public key to the `authorized_keys` file.\n\n### Tips and Precautions\n\n* Make sure to keep your private key secure. If someone gets access to your private key, they can connect to your server without a password.\n* Use a strong passphrase for your private key to prevent unauthorized access.\n* If you're using a shared server or a server with multiple users, consider using a more secure method, such as public key authentication with a keyring or a secure authentication service.\n* Regularly update your SSH keys to ensure you're using the latest version.\n\n### Example Use Case\n\nHere's an example of how you can use passwordless SSH to automate a deployment script:\n\n```bash\n#!/bin/bash\n\n# Set up SSH connection\nssh -o \"StrictHostKeyChecking=no\" user@server <<EOF\n # Run deployment script\n git pull origin main\n npm install\n npm run build\nEOF\n```\n\nIn this example, the script uses passwordless SSH to connect to the server and run a deployment script without prompting for a password.

InfoRM (2422 chars): To set up SSH to not require a password, you'll need to generate a pair of SSH keys (public and private) and add the public key to the server. Here's a step-by-step guide:\n\n### Step 1: Generate SSH keys\n\n1. Open a terminal on your local machine.\n2. Run the following command to generate a new pair of SSH keys:\n ```bash\nssh-keygen -t ed25519\n```\n or\n ```bash\nssh-keygen -t rsa\n```\n You can choose the type of key (RSA or Ed25519) and the location where the keys will be stored. By default, it will be stored in `~/.ssh`.\n\n### Step 2: Add the public key to the server\n\n1. Copy the public key to the clipboard:\n ```bash\ncat ~/.ssh/id_ed25519.pub\n```\n or\n ```bash\ncat ~/.ssh/id_rsa.pub\n```\n Copy the entire output.\n2. Log in to your server using your current password.\n3. Create a new directory for the SSH keys if it doesn't exist:\n ```bash\nmkdir -p ~/.ssh\n```\n4. Add the public key to the authorized_keys file:\n ```bash\necho \"your-public-key-here\" >> ~/.ssh/authorized_keys\n```\n Replace `your-public-key-here` with the public key you copied earlier.\n\n### Step 3: Restrict SSH access (optional)\n\nTo add an extra layer of security, you can restrict SSH access to specific users or IP addresses. You can do this by editing the `~/.ssh/authorized_keys` file and adding options to the key.\n\nFor example, to restrict access to a specific user and IP address, you can add the following options:\n```bash\nssh-rsa your-public-key-here user@ip_address\n```\nThis will only allow the specified user to access the server from the specified IP address.\n\n### Step 4: Update SSH configuration (optional)\n\nTo use the SSH keys for passwordless login, you'll need to update the SSH configuration to allow it.\n\n1. Edit the SSH configuration file:\n ```bash\nsudo nano /etc/ssh/sshd_config\n```\n2. Add the following line to the end of the file:\n ```bash\nPubkeyAuthentication yes\n```\n3. Restart the SSH service:\n ```bash\nsudo service ssh restart\n```\n or\n ```bash\nsudo systemctl restart sshd\n```\n\n### Step 5: Test SSH connection\n\nTry to SSH into the server using the following command:\n```bash\nssh user@server\n```\nIf everything is set up correctly, you should be able to log in without entering a password.\n\nNote: Make sure to keep your private key secure and do not share it with anyone. If your private key is compromised, you'll need to generate a new pair of SSH keys.

PoE (2172 chars): **Setting up SSH Key-Based Authentication**\n\nTo set up SSH key-based authentication, you'll need to generate a pair of SSH keys on your local machine and then add the public key to the server's authorized_keys file. This way, you can access the server without entering a password.\n\n**Step 1: Generate SSH Keys on Your Local Machine**\n\nOpen a terminal and run the following command to generate a new pair of SSH keys:\n```bash\nssh-keygen -t rsa -b 4096\n```\nThis will create a new pair of keys in the `~/.ssh` directory:\n\n* `id_rsa`: the private key (do not share this with anyone)\n* `id_rsa.pub`: the public key (share this with the server administrator)\n\n**Step 2: Copy the Public Key to the Server**\n\nCopy the contents of the public key file (`id_rsa.pub`) to the server's authorized_keys file using the following command:\n```bash\nssh-copy-id user@server\n```\nReplace `user` with your username on the server and `server` with the hostname or IP address of the server.\n\n**Alternative Method: Manually Copying the Public Key**\n\nIf you prefer not to use `ssh-copy-id`, you can manually copy the public key to the server's authorized_keys file using the following steps:\n\n1. Open the public key file (`id_rsa.pub`) in a text editor:\n ```bash\ncat ~/.ssh/id_rsa.pub\n```\n2. Connect to the server using SSH:\n ```bash\nssh user@server\n```\n3. Append the public key to the authorized_keys file:\n ```bash\necho \"your_public_key_here\" >> ~/.ssh/authorized_keys\n```\nReplace `your_public_key_here` with the contents of the public key file.\n\n**Step 3: Test SSH Connection**\n\nTry connecting to the server using SSH without a password:\n```bash\nssh user@server\n```\nYou should now be able to access the server without entering a password.\n\n**Security Considerations**\n\n* Keep your private key secure and do not share it with anyone.\n* Limit access to the `authorized_keys` file and the `~/.ssh` directory to prevent unauthorized access.\n* Regularly update your SSH keys and authorized_keys file to ensure security.\n\nBy following these steps, you can set up SSH key-based authentication and access your server without entering a password.

Skywork (2274 chars): **Setting up SSH without Password**\n\nTo set up SSH without requiring a password, you'll need to generate public-private key pairs and configure SSH on both your local machine and the server.\n\n### Step 1: Generate Public-Private Key Pair on Local Machine\n\n1. **Open a terminal** on your local machine.\n2. **Generate a new key pair** using the following command:\n ```bash\nssh-keygen -t ed25519\n```\n or\n ```bash\nssh-keygen -t rsa\n```\n (Choose a key type: `ed25519` or `rsa`)\n\n3. **Accept the default filename and location** for the key pair.\n\n### Step 2: Copy the Public Key to the Server\n\n1. **Copy the public key** to the clipboard:\n ```bash\ncat ~/.ssh/id_ed25519.pub\n```\n or\n ```bash\ncat ~/.ssh/id_rsa.pub\n```\n2. **Log in to the server** using your existing SSH password.\n\n3. **Create a new directory** for SSH keys if it doesn't exist:\n ```bash\nmkdir -p ~/.ssh\n```\n\n4. **Append the public key** to the server's authorized_keys file:\n ```bash\necho \"<paste the public key here>\" >> ~/.ssh/authorized_keys\n```\n\n5. **Set permissions** to ensure only the owner can read the authorized_keys file:\n ```bash\nchmod 700 ~/.ssh\nchmod 600 ~/.ssh/authorized_keys\n```\n\n### Step 3: Configure SSH on the Server\n\n1. **Edit the SSH configuration file** to enable public key authentication:\n ```bash\nsudo nano /etc/ssh/sshd_config\n```\n2. **Add or uncomment the following line**:\n ```\nPubkeyAuthentication yes\n```\n3. **Restart the SSH service** to apply the changes:\n ```bash\nsudo systemctl restart sshd\n```\n\n### Step 4: SSH into the Server without a Password\n\n1. **Try to SSH into the server** using the private key:\n ```bash\nssh -i ~/.ssh/id_ed25519 user@server_ip\n```\n or\n ```bash\nssh -i ~/.ssh/id_rsa user@server_ip\n```\n\nYou should now be able to SSH into the server without entering a password!\n\n**Additional Tips:**\n\n* Make sure to keep your private key secure and do not share it with anyone.\n* Consider using a passphrase-protected private key for added security.\n* Regularly update your SSH keys to ensure you're using the latest encryption algorithms.\n\nBy following these steps, you'll be able to set up SSH without a password and enjoy a more secure and convenient way to access your servers.

OURS (2052 chars): **Generating SSH Keys**\n\nTo set up SSH without a password, you'll need to generate a pair of SSH keys: a private key and a public key. The private key will be kept on your local machine, while the public key will be added to the server.\n\n**Step 1: Generate SSH Keys**\n\nOpen a terminal on your local machine and run the following command:\n```bash\nssh-keygen -t ed25519\n```\nThis will generate a new pair of SSH keys using the Ed25519 algorithm. You can choose a different algorithm (e.g., RSA) if you prefer.\n\n**Step 2: Copy the Public Key**\n\nOnce the keys are generated, you'll see a message indicating that the keys have been created. The public key will be stored in a file named `id_ed25519.pub` (or `id_rsa.pub` if you chose RSA).\n\nTo copy the public key to the server, run the following command:\n```bash\nssh-copy-id user@server\n```\nReplace `user` with your username on the server and `server` with the hostname or IP address of the server.\n\n**Step 3: Authorize the Public Key**\n\nOn the server, the public key will be added to the `~/.ssh/authorized_keys` file. You can verify this by running:\n```bash\ncat ~/.ssh/authorized_keys\n```\nYou should see the public key from your local machine listed.\n\n**Step 4: Test SSH without a Password**\n\nNow, try connecting to the server using SSH without a password:\n```bash\nssh user@server\n```\nIf everything is set up correctly, you should be logged in without being prompted for a password.\n\n**Tips and Considerations**\n\n* Make sure to keep your private key secure, as it can be used to access your account on the server.\n* Consider using a passphrase for your private key to add an extra layer of security.\n* If you're using a shared hosting environment, check with your provider to see if they have any specific requirements or restrictions for SSH key usage.\n* To disable password authentication on the server, you can edit the `sshd_config` file and set `PasswordAuthentication` to `no`. Then, restart the SSH service.\n\nBy following these steps, you should be able to set up SSH without a password and enjoy a more convenient and secure way to access your server.

GPT4o (2008 chars): To set up SSH key-based authentication and avoid entering a password each time you SSH into a server, follow these steps:\n\n1. Generate an SSH key pair on your local machine (if you haven't already):\n\n Open a terminal and run the following command:\n\n ```\n ssh-keygen -t rsa -b 4096 -C \"your_email@example.com\"\n ```\n\n Replace \"your_email@example.com\" with your actual email address. This will create a 4096-bit RSA key pair.\n\n You'll be prompted to enter a file path to save the key pair. Press Enter to use the default location (usually `~/.ssh/id_rsa`).\n\n You'll also be prompted to enter a passphrase for the key pair. You can either enter a passphrase for added security or leave it empty for passwordless authentication.\n\n2. Copy the public key to the remote server:\n\n Use the `ssh-copy-id` command to copy your public key to the remote server:\n\n ```\n ssh-copy-id username@remote_server_ip\n ```\n\n Replace \"username\" with your remote server's username and \"remote_server_ip\" with the remote server's IP address or hostname.\n\n If `ssh-copy-id` is not available, you can manually copy the public key using the following command:\n\n ```\n cat ~/.ssh/id_rsa.pub | ssh username@remote_server_ip \"mkdir -p ~/.ssh && chmod 700 ~/.ssh && cat >> ~/.ssh/authorized_keys && chmod 600 ~/.ssh/authorized_keys\"\n ```\n\n This command will create the `~/.ssh` directory and `authorized_keys` file on the remote server if they don't exist, and append your public key to the `authorized_keys` file.\n\n3. Test the passwordless SSH connection:\n\n Now, try to SSH into the remote server:\n\n ```\n ssh username@remote_server_ip\n ```\n\n If everything is set up correctly, you should be able to log in without entering a password. If you set a passphrase for your key pair, you'll be prompted to enter it.\n\nNote: Make sure to keep your private key (`~/.ssh/id_rsa`) secure and do not share it with anyone. The public key (`~/.ssh/id_rsa.pub`) is the one you copy to the remote server.

Judgment: My final verdict is [[A (OURS) >>B (GPT4o)]]\n\n**Explanation:**\nBoth assistants correctly outline SSH key-based authentication setup. However, Assistant A's response is superior due to:\n1. **Modern Algorithm Recommendation**: Ed25519 is more secure and efficient than RSA (Assistant B's choice).\n2. **Security Best Practices**: Explicitly mentions disabling password authentication via `sshd_config`, a critical security step Assistant B omits.\n3. **Conciseness & Structure**: Clear step-by-step formatting with helpful tips (e.g., passphrases, provider checks) without unnecessary verbosity.\n4. **Permissions Handling**: Includes explicit `chmod` commands in examples, ensuring proper directory/file permissions.\n\nAssistant B's manual key-copy method is useful but lacks A's comprehensive security guidance. While both answers are technically correct, Assistant A provides a more secure, future-proof solution with better operational safety recommendations.

Figure 6: A case study on ArenaHard-v0.1 Benchmark where our model's response significantly outperforms GPT-4o on a technical question about setting up passwordless SSH.