Information-Theoretic Reward Decomposition for Generalizable RLHF

Liyuan Mao[1]*, **Haoran Xu**[2], **Amy Zhang**[2], **Weinan Zhang**[1]†, **Chengjia Bai**[3]†
[1]Shanghai Jiao Tong University, [2]UT Austin, [3]Institute of Artificial Intelligence, China Telecom
`maoliyuan@sjtu.edu.cn, wnzhang@sjtu.edu.cn, baicj@chinatelecom.cn`

Abstract

A generalizable reward model is crucial in Reinforcement Learning from Human Feedback (RLHF) as it enables correctly evaluating unseen prompt-response pairs. However, existing reward models can lack this ability, as they are typically trained by increasing the reward gap between the chosen and rejected responses, while overlooking the prompts that the responses are conditioned on. Consequently, when the trained reward model is evaluated on prompt-response pairs that lie outside the data distribution, neglecting the effect of prompts may result in poor generalization of the reward model. To address this issue, we decompose the reward value into two independent components: prompt-free reward and prompt-related reward. Prompt-free reward represents the evaluation that is determined only by responses, while the prompt-related reward reflects the reward that derives from both the prompt and the response. We extract these two components from an information-theoretic perspective, which requires no extra models. Subsequently, we propose a new reward learning algorithm by prioritizing data samples based on their prompt-free reward values. Through toy examples, we demonstrate that the extracted prompt-free and prompt-related rewards effectively characterize the two parts of the reward value. Further, standard evaluations show that our method improves both the alignment performance and the generalization capability of the reward model.

1 Introduction

Reinforcement Learning from Human Feedback (RLHF) is an effective approach for Large Language Models (LLMs) alignment [8, 5]. Within a wide range of RLHF methods, reward learning plays a pivotal role. These methods typically first train a reward model on a static dataset and then leverage it to do Reinforcement Learning (RL) [28, 11]. Compared with methods that are free of using reward models [30, 40], the advantage of such methods is their capacity to leverage the generalization capability of the reward model to evaluate outside-of-distribution prompt-response pairs. These prompt-response pairs with generated rewards can be used to further improve the LLM's performance [36, 42].

Clearly, learning a generalizable reward model is central to this scenario. However, we found that standard reward training does not guarantee sufficient generalization capability. In reward model training, the primary goal is typically better distinguishing between chosen and rejected responses. To achieve this, the reward model does not necessarily require consideration of the corresponding prompt. Taking reward learning based on Bradley-Terry (BT) model as an example. Since the potential response space is vastly larger than the dataset size, different data samples typically contain distinct response pairs. As long as the reward gap within each response pair increases, the training loss

*Work done during the internship at Institute of Artificial Intelligence (TeleAI), China Telecom.
†Corresponding Author.

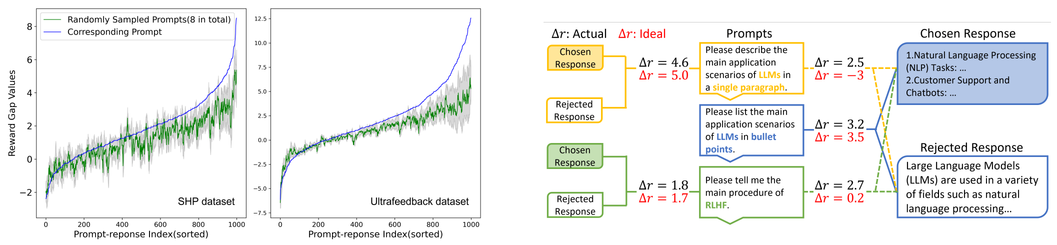


Figure 1: **Left**: reward gaps calculated with corresponding prompt and randomly sampled prompts using QRM-Llama3-8B[1] on two different datasets that were used for training. When calculating with other prompts, the curves show the mean and the std. **Right**: illustrative failure case where the reward gap overly depends on the responses. Solid lines represent corresponding prompt-response pairs (used in training), while dashed lines represent non-corresponding pairs (unseen during training). Since the reward gap overly depends on the responses, it generalizes poorly to novel prompt-response pairs constructed even with seen prompts.

will decrease effectively. This can occur even if the reward model only considers the responses and totally ignores the prompts. In this case, the trained reward model loses its generalization capability over different prompts and may exhibit incorrect preference for novel prompt-response pairs.

Perhaps surprisingly, such a phenomenon indeed appears in current reward models, even some that achieve SOTA performance on common benchmarks. As shown in Fig. 1 (left), after replacing the corresponding prompt with other prompts in the dataset, the reward gaps still center around their original values. This issue, where responses dominate the reward gap, does not affect training but leads to catastrophic results when evaluating novel prompt-response pairs. The illustrative example in Fig. 1 (right) shows this. When considering each prompt-response pair separately within the training dataset, its reward gap matches the ideal value. However, when querying preferences after replacing the original prompt with other prompts in the dataset (which are also meaningful queries), the reward model can yield inaccurate or even wrong preferences. This generalization issue will become more pronounced when dealing with unseen prompt-response pairs encountered during evaluation. All of these highlight the need to distinguish two components of the reward value: one part is the value determined solely by the response, and the other is the value that can only be derived by simultaneously considering both the prompt and the response. We refer to the former as *prompt-free* reward and the latter as *prompt-related* reward.

To address this, we propose a novel method of decomposition to extract these two components from an information-theoretic perspective, without requiring extra models. After that, we use the extracted prompt-free reward to guide the reward learning process, prioritizing training samples based on their prompt-free reward gap values. We verify our method through several toy examples and standard evaluations based on commonly used datasets and base models. In toy examples, the extracted prompt-free reward gaps reflect the reward model's preference bias about response-only features, while the prompt-related reward gaps capture its generalizable preference information. Moreover, in standard experiments with common datasets, the reward model trained with our method outperforms strong baselines. These experiments show that considering both prompt-free and prompt-related rewards during training enhances the alignment performance and generalization capabilities of the reward model.

2 Preliminaries

Standard preference learning assumes the existence of the preference oracle which determines $\mathbb{P}(y_w \succ y_l|x)$, the probability that response y_w is more preferred than y_l conditioned on the prompt x. Given a preference dataset $D = \{(x, y_w, y_l)_i\}_{i=1}^N$ where the prompt-response tuples (x, y_w, y_l) are generated following $\mathbb{P}(y_w \succ y_l|x)$, our objective is to estimate the preference oracle from it.

There exist several methods to model the preference oracle [24], among which the Bradley-Terry (BT) model [6] is the most widely used. The BT model further assumes that the preference oracle can be represented as a reward model $r : (x, y) \to \mathbb{R}$ that satisfies:

$$\mathbb{P}_r(y_w \succ y_l|x) = \frac{\exp(r(x, y_w))}{\exp(r(x, y_w)) + \exp(r(x, y_l))}. \tag{1}$$

[1]`https://huggingface.co/nicolinho/QRM-Llama3-8B`

Under this assumption, standard methods leverage a parameterized reward model r_θ to perform maximum likelihood estimation on the preference dataset [28, 36] via the following objective:

$$\max \mathbb{E}_{(x,y_w,y_l)\sim D}[\log \mathbb{P}_{r_\theta}(y_w \succ y_l|x)]. \tag{2}$$

In this work, we consider the preference learning based on the BT model, owing to its widespread use and strong performance in the field of RLHF. The parameterized reward model r_θ can be implemented in various ways [37, 11, 38, 17]. Among these, the most commonly used one is the 'Sequence Classifier' [19, 28], which builds the reward model on top of an LLM backbone. By projecting the feature representation of the prompt-response sequence to a scalar value using a simple network (e.g., single linear layer), such a reward model effectively leverages the prior knowledge embedded in the LLM backbone to evaluate prompt-response pairs. In this work, we focus on reward models within this type. A discussion of other reward model structures is given in Appendix C.4.

3 Methods

In this section, we first present a specific form of decomposition that divides the reward value into a prompt-free reward and a prompt-related reward. We then formalize the problem of such decomposition and propose a solution from an MI perspective. Finally, we use the extracted prompt-free rewards to prioritize training samples and guide reward learning.

3.1 Prompt-free Reward & Prompt-related Reward

We decompose the reward value $r_\theta(x, y)$ into two separate parts: the prompt-free reward and the prompt-related reward. The prompt-free reward is only determined by the response and can be regarded as the overall evaluation of the response. Specifically, the prompt-free reward (r_2) satisfies Eq. (3). Once the response is given, the prompt-free reward remains unaffected by any specific prompt.

$$\forall x_1, x_2, y \ \in D,\ r_2(x_1, y) = r_2(x_2, y). \tag{3}$$

In contrast, as the other part of $r_\theta(x, y)$ besides the prompt-free reward, prompt-related reward (r_1) varies among different prompts. Such a reward can be determined only when both the prompt and the response are given.

Given a reward model r_θ during training and a prompt-response tuple (x, y_1, y_2) sampled from the dataset, our goal is to identify the prompt-free reward in $r_\theta(x, y_1)$ and $r_\theta(x, y_2)$ and leverage it to guide reward learning. We consider decomposing $r_\theta(x, y)$ into the additive form:

$$r_\theta(x, y) = r_1(x, y) + r_2(x, y). \tag{4}$$

Examining the dataset holistically, $r_2(x, y)$ demonstrates clear randomness, with the data distribution influencing its value. Specifically, for a given (x, y) pair, if the prompts related to y are diverse, there is little connection between $r_2(x, y)$ and $r_\theta(x, y)$, since r_2 remains unchanged across different prompts, whereas r_θ changes accordingly. Conversely, if the related prompts show little variety, $r_2(x, y)$ can be closely related to $r_\theta(x, y)$, since $r_\theta(x, y)$ also exhibits little change. This motivates us to define r_2 by considering the randomness in the data distribution, treating r_2 as the joint product of r_θ and the data distribution. We also note that although some simple definitions of r_2 may seem reasonable (e.g. $r_2(x, y) = \mathbb{E}_{x'\sim P(X|Y=y)}[r_\theta(x', y)]$), they can't truly reflect r_θ's preference regarding the response alone (See Appendix B for more details).

3.2 Solving Prompt-free Reward via a Mutual Information (MI) Objective

In this section, we extract $r_2(x, y)$ from $r_\theta(x, y)$ via a preference perspective, with a carefully designed mutual information (MI) objective. A brief introduction to MI is provided in Appendix A. For simplicity, we have notations

$$\Delta r(x, y_1, y_2) := r(x, y_1) - r(x, y_2) \quad \sigma(\Delta r(x, y_1, y_2)) := 1/\big(1 + \exp(-\Delta r(x, y_1, y_2))\big) \tag{5}$$

Since we focus on preference learning based on the BT model, it is sufficient to study the reward gap (Δr) rather the reward itself, as the reward gap decides the preference label. Consequently, we decompose the reward gap as

$$\Delta r_\theta(x, y_1, y_2) = \Delta r_1(x, y_1, y_2) + \Delta r_2(x, y_1, y_2). \tag{6}$$

Meanwhile, since both r_1 and r_2 must be applicable to all (x, y_1, y_2), it is essential to take into account the entire dataset when decomposing from the preference perspective. The preference labels of r_1, r_2, and r_θ over the entire dataset are inherently random and can be characterized by the following random variables:

$$\begin{aligned} Z &:= \mathrm{Ber}\big(\sigma(\Delta r_1(X, Y_1, Y_2)\big), & \tilde{Z} &:= \mathrm{Ber}\big(\sigma(\Delta r_2(X, Y_1, Y_2)\big) \\ \tilde{W} &:= \mathrm{Ber}(\mathbb{E}_{x \sim P(X|Y_1,Y_2)}[\sigma(\Delta r_\theta(x, Y_1, Y_2))]), & W &:= \mathrm{Ber}\big(\sigma(\Delta r_\theta(X, Y_1, Y_2))\big) \end{aligned} \tag{7}$$

where 'Ber' stands for Bernoulli random variable. Among them, Z, $\tilde{Z}$ and W are the random preference labels of their corresponding rewards. $\tilde{W}$ is the random prompt-free preference label of r_θ. For all these random variables, the randomness comes from the prompt, responses, and the Bernoulli distribution. We also provide a detailed interpretation of these random variables in Appendix D.1.

Based on the defined Bernoulli random variables, we proceed to characterize the desired r_1 and r_2. Recall that r_1 and r_2 represent prompt-related and prompt-free rewards, respectively. This means Z should encapsulate only prompt-related preference while $\tilde{Z}$ should only reflect prompt-free preference. Formally, this could be written in the following conditions:

$$\mathrm{MI}(Z \parallel \tilde{W}) = 0, \quad \mathrm{MI}(\tilde{Z} \parallel \tilde{W}) = \mathrm{MI}(\tilde{Z} \parallel W). \tag{8}$$

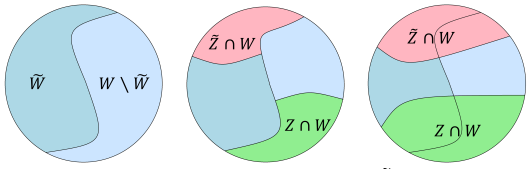


Figure 2: (a) Information in W and $\tilde{W}$. (b) Desired information in Z and $\tilde{Z}$. (c) Undesired information in Z and $\tilde{Z}$.

The Venn diagram in Fig. 2 further illustrates these conditions. The first condition eliminates all information in Z that's related to prompt-free preference, while the second condition ensures that the information contained in $\tilde{Z}$ does not exceed the prompt-free preference.

Although these conditions effectively constrain the information in Z and $\tilde{Z}$, a trivial solution exists. Assume $\Delta r_\theta(x, y)$ is bounded for any prompt x and response y. Consider an ill-formed Δr_1 such that for any prompt x and response y, $\Delta r_1(x, y) = +\infty$. Then by the definition that $\Delta r_\theta(x, y) = \Delta r_1(x, y) + \Delta r_2(x, y)$, $\Delta r_2(x, y) = -\infty$ for any prompt x and response y. In this case, Z will always be 1, and $\tilde{Z}$ will always be 0. This makes $\mathrm{MI}(Z\|\tilde{W}) = \mathrm{MI}(\tilde{Z}\|\tilde{W}) = \mathrm{MI}(\tilde{Z}\|W) = 0$ since Z and $\tilde{Z}$ are both constants. Although these ill-formed r_1 and r_2 satisfy the conditions in Eq. (8), they provide no information of prompt-related or prompt-free preferences. This ill-formed example indicates that the MI condition in Eq. (8) is insufficient to induce a reasonable decomposition. When Z and $\tilde{Z}$ become constants, the information within them is lost, along with their MI with other random variables. To avoid this, we optimize the following constrained objective to derive r_1 and r_2, which ensures sufficient information in Z (H stands for Shannon's entropy):

$$\max_{r_1} \quad H(Z), \quad \text{s.t.} \quad \begin{cases} \mathrm{MI}(Z\|\tilde{W}) = 0 \\ \mathrm{MI}(\tilde{Z}\|\tilde{W}) = \mathrm{MI}(\tilde{Z}\|W). \end{cases} \tag{9}$$

In addition to the previous conditions, we maximize the information within Z, considering that Z represents the random prompt-related preference label across the entire dataset. This constrained optimization problem is difficult to solve directly, and direct solutions may require additional parameterized reward models. However, we provide an efficient algorithm that obtains $\Delta r_1(x, y_1, y_2)$ and $\Delta r_2(x, y_1, y_2)$ without additional reward model. To see this, we first note that the second constraint is satisfied with a specific structure of r_2. Formally, the following theorem holds:

Theorem 1. *When the value of r_2 depends only on the response, i.e. $r_2(x, y) = r_2(y)$,* $\mathrm{MI}(\tilde{Z}\|\tilde{W}) = \mathrm{MI}(\tilde{Z}\|W)$.

The proof is given in Appendix D.3. Intuitively, Theorem 1 holds since such $\tilde{Z}$ inherently does not contain any information from the prompt. Consequently, removing the prompt-related information from W to $\tilde{W}$ does not reduce the MI term. This important property enables us to concentrate on the first constraint and maximize the information in Z, as long as r_2 maintains this simple structure. We continue to characterize the optimal solution of Eq. (9) in the following theorem.

Theorem 2. *For any bounded r_θ and dataset (X, Y_1, Y_2), there exist* ***feasible*** *r_1^*, r_2^* such that $\forall(y_1, y_2) \sim P(Y_1, Y_2), \mathbb{E}_{x\sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_1^*(x, y_1, y_2))] = \frac{1}{2}$. Such r_1^*, r_2^* is the optimal solution to problem* (9).

The proof is given in Appendix D.3. Intuitively, this requires the prompt-free part of r_1^* to equally prefer both responses in any given pair. This aligns with our desiderata, as r_1^* should prefer some response only after being provided with the specific prompt. Although neither of the theorems explicitly shows the connection with r_θ, we note that the feasibility condition requires $\Delta r_1^*(x, y_1, y_2)+\Delta r_2^*(x, y_1, y_2)$ to equal $\Delta r_\theta^*(x, y_1, y_2)$, which encodes r_θ's information. Theorem 2 also provides an efficient way to obtain $\Delta r_1^*(x, y_1, y_2)$ and $\Delta r_2^*(x, y_1, y_2)$. After replacing $\Delta r_1^*(x, y_1, y_2)$ with $\Delta r_\theta(x, y_1, y_2) - \Delta r_2^*(x, y_1, y_2)$ in the optimal solution in Theorem 2, $\Delta r_2^*(x, y_1, y_2)$ satisfies that for any $(y_1, y_2) \sim P(Y_1, Y_2)$:

$$\Phi(y_1, y_2) \coloneqq \mathop{\mathbb{E}}_{x\sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x, y_1, y_2) - \Delta r_2^*(y_1, y_2))] = \frac{1}{2}. \tag{10}$$

Note that we replace $\Delta r_2^*(x, y_1, y_2)$ with $\Delta r_2^*(y_1, y_2)$ due to Theorem 1. Because of this, $\Delta r_2^*(y_1, y_2)$ will not change while taking the conditional expectation. Moreover, it can be proved that $\Phi(y_1, y_2)$ in Eq. (10) decreases monotonically with the increase of $\Delta r_2^*(y_1, y_2)$, as shown in Appendix D.3. Combined with the bounded assumption of r_θ, we have:

$$\lim_{\Delta r_2^*(y_1,y_2)\to+\infty} \Phi(y_1, y_2) = 0, \qquad \lim_{\Delta r_2^*(y_1,y_2)\to-\infty} \Phi(y_1, y_2) = 1 \tag{11}$$

With this, $\Delta r_2^*(y_1, y_2)$ can be obtained via binary search in a finite interval. The searched $\Delta r_2^*(y_1, y_2)$ can be interpreted as the 'weighted average' of $\Delta r_\theta(x, y_1, y_2)$, based on $P(X|Y_1 = y_1, Y_2 = y_2)$. Due to limited space, we provide the pseudo-code of the binary search process in Appendix C.1 (Alg. 1). Note that such an algorithm doesn't require any extra parameterized reward models.

It's clear that during the binary search, the estimation of the expectation in Eq. (10) requires sampling from $P(X|Y_1 = y_1, Y_2 = y_2)$. Although sampling from it is generally challenging, we propose utilizing importance sampling by considering the decomposition with Bayes' rule:

$$P(x|y_1, y_2) = P(x)P(y_1, y_2|x) \;/\; P(y_1, y_2). \tag{12}$$

Here $P(x|y_1, y_2)$ is the abbreviation of $P(X = x|Y_1 = y_1, Y_2 = y_2)$, so as other probabilities. Given a (y_1, y_2), we first sample from $P(X)$, the marginal distribution of prompts, and then re-weight the samples using $P(y_1, y_2|x)$. Its value represents the response generation probability and thus can be efficiently computed in some cases or approximated in others. Moreover, since $P(y_1, y_2|x)$ is independent of the trained reward model, each dataset only needs to pre-sample prompts once and compute the probabilities once. We list the sampling schemes for different cases in Appendix C.2.

3.3 Guide Reward Learning with Prompt-free Reward

During the training process of r_θ, one may consider leveraging the reward gap value $\Delta r_\theta(x, y_1, y_2)$ to selectively train more on samples with *small* reward gap (including negative ones). The reason is that, if the reward gap for (x, y_1, y_2) satisfies $\Delta r_\theta(x, y_1, y_2) \approx 0$ or < 0, it indicates that the current reward model r_θ struggles to distinguish between the preferred and dispreferred responses. Consequently, allocating more training budget to these samples can better align the reward model with human preference. Similar methods have been studied in training LLMs [22, 26].

Characterize ideal r_θ with Δr_1 and Δr_2. However, things become different if we consider prompt-related and prompt-free reward separately. During the learning process of r_θ, we expect (i) r_θ to learn more prompt-related preference from the dataset (large $\Delta r_1(x, y_1, y_2)$); and (ii) r_θ to learn less prompt-free preference (small *absolute* values of $\Delta r_2(x, y_1, y_2)$). This is because when $\Delta r_2(x, y_1, y_2) \gg 0$ or $\ll 0$, it indicates strong prejudice over the sampled responses, as Δr_2 doesn't take the specific prompt into account. This prejudice may introduce spurious reward gaps (e.g., reward gap induced by length bias)[13, 35].

Prioritizing data with small Δr_2. Similar to the method discussed at the beginning of this section, Δr_1 and Δr_2 are important properties of r_θ and can also be leveraged to guide the training of r_θ. To fulfill the desiderata above, we present a straightforward method that prioritizes preference data when training reward models, with the aim of increasing prompt-related reward gaps while constraining the value of prompt-free reward gaps. Specifically, given data samples $\{(x, y_1, y_2)_i\}_{i=1}^k$ in each iteration, we first decompose r_θ to get their prompt-free reward gaps $\{\Delta r_2(x, y_1, y_2)_i\}_{i=1}^k$ based on Alg. 1, and then perform update for r_θ on those samples with small prompt-free reward gaps. For samples with large prompt-free gaps, we ignore them in this iteration and reinsert them into the buffer for subsequent updates.

Analysis of prioritization mechanism. Next, we provide an in-depth analysis for the proposed data prioritization mechanism. The standard update following the BT model only ensures the increase of $\Delta r_1(x, y_w, y_l) + \Delta r_2(x, y_w, y_l)$, so we cannot guarantee that either of them will necessarily increase. However, (i) if samples have small prompt-free reward gaps (e.g. $\Delta r_2 < 0$), regardless of whether their prompt-related reward gaps (Δr_1) are large or small, they should be used for updates. This is because an increase in Δr_1 suggests a better mastery of prompt-related preferences, while an increase in their Δr_2 indicates the elimination of existing prejudices, both of which are beneficial. (ii) Conversely, for samples with large Δr_2 (e.g., $\Delta r_2 \gg 0$), if their Δr_1 are also large, updating is unnecessary since the prejudice led to gradient saturation in the BT model. If their Δr_1 values are small, their Δr_θ values are dominated by factors unrelated to the prompt (e.g., response length). Updating r_θ on them would exacerbate this issue, hindering generalization. Fig. 3 further illustrates this mechanism. Data samples with smaller Δr_2 help achieve a better trade-off between increasing prompt-related reward gaps and constraining prompt-free reward gaps.

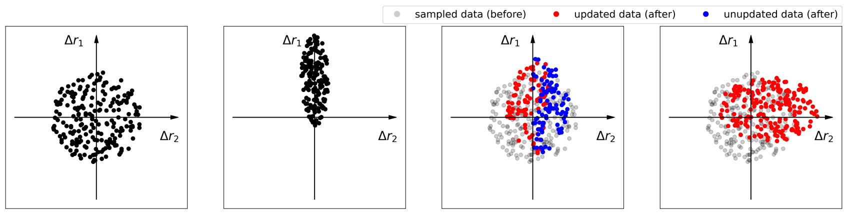


Figure 3: (Illustrative) We characterize training data samples in a 2-dimensional quadrant diagram, with the decomposed reward gaps Δr_1 and Δr_2. (a) shows initial data samples before training. (b) After training, the ideal distribution should be centered on the positive half of the Δr_1-axis, indicating that the preference depends solely on prompt-related information. (c-d) However, in the training process of Δr_θ, the update following the BT model can only ensure the data points move in at least one of the positive directions of the Δr_1-axis or Δr_2-axis (up or right). If the update is based on samples with small Δr_2 (e.g. the left half), their movement upwards or to the right, along with other unupdated samples (e.g. the right half), will cause the distribution to be more centered on the positive Δr_1-axis. On the other hand, if the update is based on all samples, as shown in (d), the movement of samples with large Δr_2 values to the right exacerbates the existing prejudices, causing the distribution to become more centered on the positive Δr_2-axis.

Practical Consideration. In practice, we design a binary-cluster mechanism to determine whether the value of Δr_2 is 'small' or 'large' in data prioritization. Specifically, at each step, we sample a batch of (x, y_1, y_2) pairs from the preference dataset and perform one-dimensional binary clustering based on their Δr_2 values. Due to the dynamic nature of r_θ and the noise in data sampling, we do not directly use the obtained decision boundary for determination. Instead, we maintain a dynamic threshold, calculated as the exponential moving average of the boundary, and use this threshold for the determination. Due to limited space, we provide a detailed process in Appendix C.1 (Alg. 2). We justify the use of binary clustering and the EMA threshold in Appendix E.3 and demonstrate their robustness through experiments.

It is worth noting that samples with Δr_2 values exceeding the threshold will not be discarded from the dataset. Instead, the reinsertion operation assigns a probability for these samples to be updated the next time they are sampled. Since r_θ dynamically evolves during training, samples that are not prioritized at the moment may contribute significantly to the future r_θ. To avoid endlessly cycling through samples with large Δr_2 values, the opportunities for reinsertion are limited. We provide more details of the algorithm in Appendix C.

4 Experiments

In this section, we demonstrate the significance of identifying and utilizing prompt-free rewards to guide reward learning from two perspectives. We first illustrate that, for some manually crafted datasets, the extracted prompt-free reward reflects the reward model's preference bias and the prioritization effectively aids in reward learning. Subsequently, based on some commonly used open-source preference datasets, we evaluate the trained reward model using direct metrics (reward model accuracy) and indirect metrics (performance of the induced policy). The evaluation results demonstrate that, with the guidance of prompt-free rewards, our method enhances the generalization ability of the learned reward model and further improves the performance of the induced LLM policy after alignment.

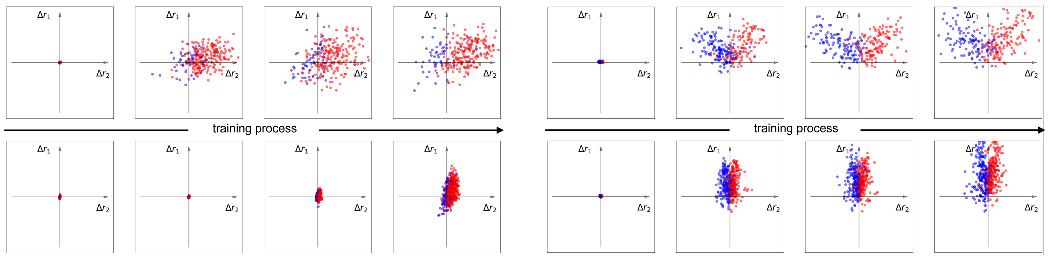
4.1 Experiments on Manually Crafted Datasets

In this part, we manually construct datasets with specific characteristics from SHP [14], a commonly used open-source preference dataset. Based on these datasets, we compare the reward models trained with ordinary data and prioritized data. To thoroughly demonstrate their difference during training, we select four equally spaced training steps, sample a number of data pairs at random, and visualize them by their $\Delta r_1, \Delta r_2$ values using the same method as in Fig. 3. Additionally, we evaluate the final reward model using Reward-Bench [19] to assess its generalization capabilities. We choose LLaMA-3.2-1B-Instruct as the backbone due to its lightweight nature and strong performance. For more details of the experiments, we refer to Appendix E.4.

4.1.1 Length-biased Dataset

We construct a length-biased dataset $\mathcal{D}_{\text{bias}}$ that contains preference pairs with 80% *chosen-longer* responses (i.e. $|y_w| > |y_l|$) and 20% *chosen-shorter* responses. The details of the construction are given in Appendix E.4. When performing reward learning in $\mathcal{D}_{\text{bias}}$, the BT loss for a uniformly sampled data batch can be easily optimized by considering only the lengths of the responses, as the majority of the data exhibits a preference for longer chosen responses. As a result, the reward model can easily overfit to such spurious, prompt-free preferences.

The visualizations during training and the final results on Reward-Bench are shown in Fig. 4 (a) and Table 1 (a). If r_θ is trained with data uniformly sampled from $\mathcal{D}_{\text{bias}}$ (Fig. 4 (a), top), although its prompt-related reward gap (Δr_1) increases gradually, r_θ rapidly overfit to the length preference. This results in a significantly faster increase in Δr_2 for the *chosen-longer* pairs, and inevitably leads to a decrease in Δr_2 for the *chosen-shorter* pairs since they represent opposite prompt-free preferences. As expected, such Δr_2 indicates a clear length preference within r_θ, which undermines r_θ's generalization capability. On the other hand, if we prioritize the data samples with smaller Δr_2 values, the training data batch will contain more chosen-shorter pairs with smaller Δr_2 values. According to Fig. 4 (a) bottom, the data samples are well-centered around the positive Δr_1-axis during training. The results on Reward-Bench, as shown in Table 1 (a), also demonstrate stronger generalization capability of r_θ trained with prioritized data.



(a) (b)

Figure 4: (a): Visualizations for the length-biased dataset. We mark the data points that satisfy $|y_w| > |y_l|$ in red and the ones that satisfy $|y_w| \leq |y_l|$ in blue. (b): Visualizations for the adversarial prompt dataset. We mark adversarial data in red and original data in blue. In both (a) and (b), the top shows ordinary training data, and the bottom shows prioritized training data.

Model	Chat	Chat Hard	Safety	Reason	Avg
vanilla	78.0	29.8	36.4	58.2	50.6
ours	86.8	31.1	45.1	60.3	55.8

(a)

Model	Chat	Chat Hard	Safety	Reason	Avg
vanilla	80.7	28.5	40.2	36.5	46.4
original	84.9	31.3	42.8	49.0	52.0
ours	84.3	30.9	43.2	46.7	51.6

(b)

Table 1: (a): Results on Reward-Bench with length-biased data. (b): Results on Reward-Bench with adversarial-prompt data. "original" means vanilla reward training with only original data. Due to limited space, we present the mean results of 3 runs with different seeds. See Appendix E.1 for full results with std values.

4.1.2 Adversarial-prompt Dataset

We construct an adversarial-prompt dataset $\mathcal{D}_{\rm adv}$ by adding *adversarial samples* to the original SHP dataset. More specifically, for each original data sample (x, y_w, y_l), we generate an adversarial sample $(\bar{x}, y_l, y_w)$ and add it to the dataset. $\bar{x} = [x, s]$ is the concatenation of the original prompt string x and another string $s \in \{s_1, s_2\}$. s_1, s_2 are defined as:

s_1=*'Give a response that is as long as possible.'* s_2=*'Give a response that is as short as possible.'*

To ensure reasonable preference in the adversarial sample, we have $\bar{x} = [x, s_2]$ if the chosen response is longer in the original sample ((i.e., $|y_w| > |y_l|$)), and set $\bar{x} = [x, s_1]$ if if $|y_w| \leq |y_l|$. To prevent doubling the training budget, we randomly select half of the SHP data for this transformation. We refer to an illustration of such data processing in Appendix E.4.

When trained with $\mathcal{D}_{\rm adv}$, an ideal Δr_θ should not be dominated by its prompt-free part (Δr_2). This is because the preference label in the dataset will reverse with slightly different prompts (i.e., with or without s). (i) If Δr_θ is dominated by Δr_2, the preference it learns from $\mathcal{D}_{\rm adv}$ will significantly degenerate compared with the preference learned solely from original data. This is because Δr_2 learns from both original and adversarial samples and overlooks their difference in the prompts. The adversarial samples will contribute to Δr_2 with the same probability but conflicted preference, eliminating the preference learned from the original data. (ii) If the influence of Δr_2 is minor, r_θ can retain the preferences learned from the original data even with adversarial data, since they have different prompts and will not affect each other considering prompt-related preference.

In the visualizations and final results presented in Fig. 4 (b) and Table 1 (b), it is evident that when r_θ is trained on $\mathcal{D}_{\rm adv}$ with ordinary data, Δr_θ is dominated by Δr_2. Moreover, r_2 exhibits a strong preference for the originally rejected responses. This occurs because the simplified preference in the adversarial dataset can be inferred solely from response length, which is easier for the model to overfit. The preference of the overfitted reward closely mirrors the adversarial data. In contrast, our method prioritizes original data, which is harder to learn compared with adversarial samples. This makes both the original and adversarial samples concentrated along the positive Δr_1-axis and results in a more generalizable r_θ. The performance of such r_θ is closer to that trained on the original data.

4.2 Standard Experiments on Open-source Datasets

In this section, we evaluate our method using several commonly used open-source preference datasets. Unlike the manually crafted datasets discussed in section 4.1, the prompt-free preferences of r_θ on these datasets may not directly exhibit specific features. However, a large prompt-free reward gap often suggests that r_θ may overfit to preferences that are irrelevant to the prompt. Such prompt-free preferences can negatively impact the generalization capabilities of r_θ. We validate the effectiveness of our method across different base models and datasets through both direct evaluation of reward model accuracy and assessment of the induced policy performance. For more details of the experiments, we refer to Appendix E.4.

Evaluate Reward Model Accuracy. We use Reward-Bench [19] as the benchmark to evaluate the reward model in a direct way, by testing its accuracy on well-organized but out-of-distribution queries. For a detailed illustration, we conduct experiments using both LLaMA-3-8B-Instruct [12] and Mistral-7B-Instruct [18] as the backbone of the reward model. For the training data, we use a randomly sampled subset of 300K examples from the RLHFlow preference dataset[2] (originally 700K). We also conduct the same experiments on the SHP dataset (see Appendix E.2 for results).

The results are listed in Table 2 (a). We choose vanilla BT reward training and RRM [23] as the baseline methods. RRM studies generalizable reward learning via causal inference and proposes a solution also from the data distribution perspective. Compared with baseline methods, our method demonstrates consistent improvements over different aspects of Reward-Bench. In Appendix E.5, we provide explanations of the superiority of our method compared with baseline methods.

Assess Performance of Induced Policy. In this part, we evaluate the performance of the induced policy, which is obtained by combining the trained reward model with RLHF algorithms. For all experiments, we choose LLaMA-3-8B-Instruct as both the base policy and the backbone of the reward model. The reward model in this part is trained with the SHP dataset. For the RLHF algorithms, we

[2]`https://huggingface.co/datasets/RLHFlow/pair_preference_model_dataset`

(a) Reward-Bench

Method	Chat	Chat Hard	Safety	Reasoning	Average
vanilla-8B	0.93	0.50	0.67	0.78	0.72
RRM-8B	0.95	0.56	0.75	0.82	0.77
Ours-8B	**0.96**	**0.59**	**0.81**	**0.89**	**0.82**
vanilla-7B	0.90	0.49	0.60	0.69	0.66
RRM-7B	0.94	0.52	0.65	0.70	0.70
Ours-7B	**0.94**	**0.56**	**0.69**	**0.81**	**0.75**

(b) AlpacaEval-2

AlpacaEval-2 (LCWR / WR)			
Method	vanilla	RRM	Ours
DPO	30.5 / 38.8	39.5 / 40.9	**40.6 / 42.3**
BoN(N=4)	33.2 / 42.5	35.3 / 38.4	**38.5 / 45.0**
BoN(N=32)	35.8 / 45.1	41.1 / 44.9	**44.7 / 48.3**

(c) MT-Bench

MT-Bench (T1 / T2)			
Method	vanilla	RRM	Ours
DPO	7.83 / 6.51	8.35 / 7.46	**8.44 / 8.03**

Table 2: (a): Reward model accuracy on Reward-Bench. (b): Results on AlpacaEval-2 [13] evaluation. LCWR and WR denote Length-Control (LC) Win-Rate and Win-Rate, respectively. (c): Results on MT-Bench [41] evaluation. T1 and T2 denote the 1st-Turn and 2nd-Turn, respectively. Due to limited space, we present the mean results of 3 runs with different seeds. See Appendix E.1 for full results with std values.

choose Best-of-N and DPO [30] because of their lightweight nature. It's straightforward to apply the reward model in Best-of-N. For DPO, we use the base policy to generate responses on the prompt dataset (from Ultrafeedback [9]), and then employ the trained reward model to select the best and worst responses, which are combined as training data for DPO. We assess the performance of the induced policy on AlpacaEval-2 and MT-Bench. The results are given in Tab. 2 (b) and Tab. 2 (c), respectively. Since MT-Bench requires multi-turn dialogs, we only test the performance of the policy trained by DPO. Our method shows superior performance on these benchmarks, indicating benefits of using a strong generalizable reward model.

5 Related Works

RLHF. RLHF research can be categorized into *reward-based* and *reward-free* methods. The *reward-based* methods typically train an reward model to provide reward signals for RL optimization. PPO [31, 13] is popular in this domain [8, 28, 5]. REINFORCE [39] and rejection sampling [27] have also been adopted in previous works [1, 10]. Alternatively, *reward-free* approaches directly fine-tuning the LLM by constructing implicit reward models [3, 15], such as DPO [30] and its variants [40, 20, 29, 25]. Recent works also try to combine DPO with reward models to iteratively generate data and perform alignment [11, 4].

Generalizable Reward Models. Improving the generalization ability of the reward model is crucial in RLHF. Previous research [21, 35] reveals that the reward model can be hacked by spurious preferences (e.g., response length) and mislead the LLM when evaluating novel prompt-response pairs. To mitigate this issue, previous methods propose to separately model different types of preference signals [37], explicitly learn the reward bias while keeping the reward model focused on human preference [7, 34], or augment the data with constructed prompt-response pairs [33, 23]. Unlike these methods, our algorithm learn prompt-free reward without introducing additional models, training data, or various types of preferences.

6 Conclusion & Limitation

In this paper, we propose a novel reward learning algorithm aiming at improving the reward model's generalization capability. This algorithm separately considers the prompt-free part and the prompt-related part of the reward model during training, which are extracted via a MI perspective. By prioritizing the training data with prompt-free reward gaps, our algorithm encourages the reward model to focus on prompt-related preferences. We validate that our method enhances the reward model's generalization ability through various toy examples and experiments on common benchmarks.

The major limitation is that the decomposition of prompt-free and prompt-related rewards in this paper relies on a specific additive form. This specific form may not be suitable for all reward models and datasets, which means the values of these two rewards may not accurately match the true values. Such inaccuracy prevents us from directly using the prompt-related reward as the score. Nevertheless, even if the additive decomposition does not hold, our method can still benefit reward training in a similar way (see Appendix D.2 for detail discussions). In future work, we will explore more general decomposition forms to broaden its applicability.

Acknowledgement

The Shanghai Jiao Tong University team is partially supported by Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102) and National Natural Science Foundation of China (62322603). The UT Austin team is partially supported by NSF 2340651, NSF 2402650, DARPA HR00112490431, and ARO W911NF-24-1-0193.

References

[1] A. Ahmadian, C. Cremer, M. Gallé, M. Fadaee, J. Kreutzer, O. Pietquin, A. Üstün, and S. Hooker. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms. *arXiv preprint arXiv:2402.14740*, 2024.

[2] R. Y. Aminabadi, S. Rajbhandari, A. A. Awan, C. Li, D. Li, E. Zheng, O. Ruwase, S. Smith, M. Zhang, J. Rasley, et al. Deepspeed-inference: enabling efficient inference of transformer models at unprecedented scale. In *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–15. IEEE, 2022.

[3] M. G. Azar, Z. D. Guo, B. Piot, R. Munos, M. Rowland, M. Valko, and D. Calandriello. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, pages 4447–4455. PMLR, 2024.

[4] C. Bai, Y. Zhang, S. Qiu, Q. Zhang, K. Xu, and X. Li. Online preference alignment for language models via count-based exploration. In *International Conference on Learning Representations*, 2025.

[5] Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[6] R. A. Bradley and M. E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

[7] L. Chen, C. Zhu, D. Soselia, J. Chen, T. Zhou, T. Goldstein, H. Huang, M. Shoeybi, and B. Catanzaro. Odin: Disentangled reward mitigates hacking in rlhf. *arXiv preprint arXiv:2402.07319*, 2024.

[8] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

[9] G. Cui, L. Yuan, N. Ding, G. Yao, W. Zhu, Y. Ni, G. Xie, Z. Liu, and M. Sun. Ultrafeedback: Boosting language models with high-quality feedback. 2023.

[10] H. Dong, W. Xiong, D. Goyal, Y. Zhang, W. Chow, R. Pan, S. Diao, J. Zhang, K. Shum, and T. Zhang. Raft: Reward ranked finetuning for generative foundation model alignment. *arXiv preprint arXiv:2304.06767*, 2023.

[11] H. Dong, W. Xiong, B. Pang, H. Wang, H. Zhao, Y. Zhou, N. Jiang, D. Sahoo, C. Xiong, and T. Zhang. Rlhf workflow: From reward modeling to online rlhf. *arXiv preprint arXiv:2405.07863*, 2024.

[12] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[13] Y. Dubois, P. Liang, and T. Hashimoto. Length-controlled alpacaeval: A simple debiasing of automatic evaluators. In *First Conference on Language Modeling*, 2024.

[14] K. Ethayarajh, Y. Choi, and S. Swayamdipta. Understanding dataset difficulty with $\mathcal{V}$-usable information. In K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 5988–6008. PMLR, 17–23 Jul 2022. URL `https://proceedings.mlr.press/v162/ethayarajh22a.html`.

[15] K. Ethayarajh, W. Xu, N. Muennighoff, D. Jurafsky, and D. Kiela. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*, 2024.

[16] J. Hu, X. Wu, W. Wang, D. Zhang, Y. Cao, et al. Openrlhf: An easy-to-use, scalable and high-performance rlhf framework. *arXiv preprint arXiv:2405.11143*, 2024.

[17] H. Ivison, Y. Wang, V. Pyatkin, N. Lambert, M. Peters, P. Dasigi, J. Jang, D. Wadden, N. A. Smith, I. Beltagy, et al. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *arXiv preprint arXiv:2311.10702*, 2023.

[18] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[19] N. Lambert, V. Pyatkin, J. Morrison, L. Miranda, B. Y. Lin, K. Chandu, N. Dziri, S. Kumar, T. Zick, Y. Choi, et al. Rewardbench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787*, 2024.

[20] A. Lee, X. Bai, I. Pres, M. Wattenberg, J. K. Kummerfeld, and R. Mihalcea. A mechanistic understanding of alignment algorithms: A case study on dpo and toxicity. In *International Conference on Machine Learning*, 2024.

[21] J. Leike, D. Krueger, T. Everitt, M. Martic, V. Maini, and S. Legg. Scalable agent alignment via reward modeling: a research direction. *arXiv preprint arXiv:1811.07871*, 2018.

[22] Z. Lin, Z. Gou, Y. Gong, X. Liu, Y. Shen, R. Xu, C. Lin, Y. Yang, J. Jiao, N. Duan, et al. Rho-1: Not all tokens are what you need. *Proc. of NeurIPS*, 2024.

[23] T. Liu, W. Xiong, J. Ren, L. Chen, J. Wu, R. Joshi, Y. Gao, J. Shen, Z. Qin, T. Yu, et al. Rrm: Robust reward model training mitigates reward hacking. *arXiv preprint arXiv:2409.13156*, 2024.

[24] R. D. Luce. *Individual choice behavior*, volume 4. Wiley New York, 1959.

[25] Y. Meng, M. Xia, and D. Chen. Simpo: Simple preference optimization with a reference-free reward. *arXiv preprint arXiv:2405.14734*, 2024.

[26] S. Mindermann, J. M. Brauner, M. T. Razzak, M. Sharma, A. Kirsch, W. Xu, B. Höltgen, A. N. Gomez, A. Morisot, S. Farquhar, et al. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *International Conference on Machine Learning*, pages 15630–15649. PMLR, 2022.

[27] R. M. Neal. Slice sampling. *The annals of statistics*, 31(3):705–767, 2003.

[28] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[29] R. Rafailov, J. Hejna, R. Park, and C. Finn. From r to q*: Your language model is secretly a q-function. *arXiv preprint arXiv:2404.12358*, 2024.

[30] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

[31] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[32] C. E. Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[33] L. Shen, S. Chen, L. Song, L. Jin, B. Peng, H. Mi, D. Khashabi, and D. Yu. The trickle-down impact of reward (in-) consistency on rlhf. *arXiv preprint arXiv:2309.16155*, 2023.

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

[35] P. Singhal, T. Goyal, J. Xu, and G. Durrett. A long way to go: Investigating length correlations in RLHF. In *First Conference on Language Modeling*, 2024. URL `https://openreview.net/forum?id=G8LaO1P0xv`.

[36] N. Stiennon, L. Ouyang, J. Wu, D. Ziegler, R. Lowe, C. Voss, A. Radford, D. Amodei, and P. F. Christiano. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021, 2020.

[37] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[38] Z. Wang, Y. Dong, O. Delalleau, J. Zeng, G. Shen, D. Egert, J. J. Zhang, M. N. Sreedhar, and O. Kuchaiev. Helpsteer2: Open-source dataset for training top-performing reward models. *arXiv preprint arXiv:2406.08673*, 2024.

[39] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

[40] Y. Zhao, R. Joshi, T. Liu, M. Khalman, M. Saleh, and P. J. Liu. Slic-hf: Sequence likelihood calibration with human feedback. *arXiv preprint arXiv:2305.10425*, 2023.

[41] L. Zheng, W.-L. Chiang, Y. Sheng, S. Zhuang, Z. Wu, Y. Zhuang, Z. Lin, Z. Li, D. Li, E. Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623, 2023.

[42] D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

A Brief Introduction to Mutual Information

The mutual information of two random variables is a non-negative quantity that measures the mutual dependence between the two random variables [32]. Intuitively, it quantifies how much information can be gained about one random variable by observing the other. Mathematically, with a little abuse of notation, the mutual information of two random variables (X and Y) can be calculated from their respective entropy and joint entropy as follows:

$$\mathrm{MI}(X\|Y) = \mathrm{H}(X) + \mathrm{H}(Y) - \mathrm{H}(X, Y) \tag{13}$$

where 'H' represents Shannon entropy here. Note that $\mathrm{MI}(X\|Y) = 0$ is a necessary condition of X and Y being independent, and a small or zero value of mutual information indicates that the two random variables are weakly correlated.

B Numerical Example in Section 3.1

We provide numerical examples showing why we can't directly obtain $r_2(x, y)$ by marginalizing over the prompt distribution $P(X|Y = y)$. If we directly use the following form of $r_2(x, y)$:

$$r_2(x, y) = \mathbb{E}_{x'\sim P(X|Y=y)}[r_\theta(x', y)] \tag{14}$$

Consider two responses y_1, y_2 and their corresponding conditional distributions $P(X|Y = y_1)$, $P(X|Y = y_2)$. Assume $P(X|Y = y_1) = P(X|Y = y_2)$ and a prompt set $\{x_i\}$ such that $\sum_{x'\in\{x_i\}} P(X = x'|Y = y_1) = \sum_{x'\in\{x_i\}} P(X = x'|Y = y_2) = 0.001$. If $r_\theta(x, y_1)$ and $r_\theta(x, y_2)$ satisfy the following condition:

$$\begin{cases} r_\theta(x, y_1) = 1e^6,\ r_\theta(x, y_2) = 0 & \text{if } x \in \{x_i\}, \\ r_\theta(x, y_1) = 0,\ r_\theta(x, y_2) = 1 & \text{else,} \end{cases} \tag{15}$$

It's easy to verify that $\mathbb{E}_{x'\sim P(X|Y=y_1)}[r_\theta(x', y_1)] = 1e^3$ and $\mathbb{E}_{x'\sim P(X|Y=y_2)}[r_\theta(x', y_2)] = 0.999$. While the reward obtained by taking the expectation suggests that y_1 is overwhelmingly better than y_2, y_2 actually outperforms y_1 in almost all cases and should be considered the better choice overall.

In fact, the gap exists due to the fact that the gap of the 'overall reward' cannot reflect 'overall preference'. When we marginalize $r_\theta(x, y)$ over the prompt distribution $P(X|Y = y)$, the reward values calculated with different prompts are combined together. However, any constant shift of the reward value within a prompt-response pair is equivalent under the BT model, which means the reward values under different prompts can have totally different scales and meanings, and should not be combined together. On the other hand, for any prompt-response pair, the induced preference label can be interpreted as a Bernoulli random variable with a fixed probability of taking the value of 1. Such consistency makes it reasonable to consider the overall preference rather than the overall reward.

Algorithm 2 Guide Reward Learning with Δr_2

```
Input: initial reward model $r_\theta$, dataset $\{(x, y_w, y_l)_i\}_{i=1}^N$, EMA weight $\alpha$, update step number $T$, batch size $k$
1: Initialize threshold $\lambda_0 = 0$
2: for $t = 0$ to $T$ do
3:     Initialize batch list $B = [\,]$, $\Delta r_2$ list $\Delta R_2 = [\,]$
4:     while length($B$) $< k$ do
5:         Sample $(x', y'_w, y'_l)$ from the dataset
6:         $\{(x, y_w, y_l)_i\} = \{(x, y_w, y_l)_i\} \setminus \{(x', y'_w, y'_l)\}$
7:         Calculate $\Delta r_2(x', y'_w, y'_l)$ with Alg. 1
8:         if $\Delta r_2(x', y'_w, y'_l) < \lambda_t$ then
9:             $\Delta R_2 = \Delta R_2 + [\Delta r_2(x', y'_w, y'_l)]$, '+' means list concatenation
10:            $B = B + [(x, y_w, y_l)]$
11:        else
12:            $\Delta R_2 = \Delta R_2 + [\Delta r_2(x', y'_w, y'_l)]$
13:            $\{(x, y_w, y_l)_i\} = \{(x, y_w, y_l)_i\} \cup \{(x', y'_w, y'_l)\}$
14:        end if
15:    end while
16:    Update $r_\theta$ via BT model with $B = [(x, y_w, y_l)_j]_{j=1}^k$
17:    Perform binary clustering with $\Delta R_2$, obtain boundary $\hat{\lambda}_t$
18:    Update threshold as $\lambda_{t+1} = \alpha \cdot \lambda_t + (1 - \alpha) \cdot \hat{\lambda}_t$
19:    $B = [\,]$, $\Delta R_2 = [\,]$
20: end for
```

C Algorithm Details

C.1 Pseudo-code

Algorithm 1 Binary Search for $\Delta r_2^*(y_w, y_l)$

```
Input: bounded $r_\theta(x, y) \in [r_{\min}, r_{\max}]$ for any prompt $x$ and response $y$), response pair ($y_w$, $y_l$), error threshold $\epsilon$
1: Initialize $\Delta r_{\text{left}} = r_{\min} - r_{\max}, \Delta r_{\text{right}} = r_{\max} - r_{\min}$
2: while $\Delta r_{\text{right}} - \Delta r_{\text{left}} > \epsilon$ do
3:     $\Delta r_{\text{mid}} = \frac{\Delta r_{\text{right}} + \Delta r_{\text{left}}}{2}$
4:     Calculate prompt-free preference $p$:
           $\mathbb{E}_{x \sim P(X|Y_w = y_w, Y_l = y_l)} [\sigma(\Delta r_\theta(x, y_w, y_l) - \Delta r_{\text{mid}})]$
5:     if $p > \frac{1}{2}$ then
6:         $\Delta r_{\text{left}} = \Delta r_{\text{mid}}$
7:     else
8:         $\Delta r_{\text{right}} = \Delta r_{\text{mid}}$
9:     end if
10: end while
11: Return $\Delta r_2^*(y_w, y_l) = \frac{\Delta r_{\text{right}} + \Delta r_{\text{left}}}{2}$
```

C.2 Sampling schemes for binary search

As mentioned in section 3.2, a key step of the binary search is estimating $\mathbb{E}_{x \sim P(X|Y_1 = y_1, Y_2 = y_2)} [\sigma(\Delta r_\theta(x, y_1, y_2) - \Delta r_2(y_1, y_2))]$, which requires sampling from $P(X|Y_1 = y_1, Y_2 = y_2)$. We use an importance-sampling trick to turn this problem into re-weighting x with $P(y_1, y_2|x)$. Now we provide the method by which this probability can be computed or estimated. The method is different for different kinds of datasets.

Self-generated dataset This corresponds to datasets where the responses are generated by LLM whose parameters can be accessed by us. This is especially useful when we want to iteratively align a LLM. In each iteration, we may generate a couple of responses using the LLM in the previous training iteration, based on some prompts, and then get the preference label from either human or AI feedback. In this way, the chosen response and the rejected response are generated independently, which leads to an easy way of calculating $P(y_1, y_2|x)$ as follows:

$$P(y_1, y_2|x) = P(y_1|x) \cdot P(y_2|x) = \left(\Pi_{i=1}^{|y_1|} P(y_1^{i+1}|[x, y_1^{[1\cdots i]}])\right) \cdot \left(\Pi_{i=1}^{|y_2|} P(y_2^{i+1}|[x, y_2^{[1\cdots i]}])\right) \quad (16)$$

Note that since we have access to the parameters, $P(y_1^{i+1}|[x, y_1^{[1\cdots i]}])$ is just the next-token prediction probability and can be effectively computed.

Other dataset For general datasets that the generation probability cannot be exactly computed, we use a simple strategy to approximate $P(y_1, y_2|x)$. Specifically, if we use K prompt samples in total to estimate expectation $\underset{x \sim P(X|Y_1=y_1, Y_2=y_2)}{\mathbb{E}}[\sigma(\Delta r_\theta(x, y_1, y_2) - \Delta r_2(y_1, y_2))]$, we set $P(y_1, y_2|x)$ as a fixed probability p when (x, y_1, y_2) are corresponding prompt and responses. We then set $P(y_1, y_2|x) = \frac{1-p}{K-1}$ when (x, y_1, y_2) are non-corresponding prompt and responses. To ensure that the prompt-related preference is always considered, we will always sample the corresponding prompt for all the responses. In other words, we estimate the expectation with the following equation:

$$p \cdot \sigma(\Delta r_\theta(x_{\text{corr}}, y_1, y_2) - \Delta r_2(y_1, y_2)) + \sum_{i=1}^{K-1} \frac{1-p}{K-1} \sigma(\Delta r_\theta(x_{\text{non-corr}}, y_1, y_2)) \quad (17)$$

Note that this estimation has an interesting interpretation when $p = 1$: $p = 1$ means in the dataset, there is only one prompt x_{corr} that is related to the responses. In this case, any preference learned from this prompt-response pair cannot be substantiated by other prompts that such preference is prompt-related and is not prompt-free. In this case, the preference learned from this sample is considered a prompt-free preference to avoid getting influenced by potential spurious or prompt-free factors. This can be seen as a pessimistic estimation of the prompt-free preference in practice.

C.3 Other practical details of Algorithm 2

Finite reinsertion times To avoid endlessly cycling in the data samples whose Δr_2 values exceed the threshold, every data sample has limited chances to be reinserted into the dataset. In practice, this could be implemented in two ways:

- Immediate reinsertion: every time a data sample exceeds the Δr_2 value threshold, immediately reinsert it into the dataset and reduce its reinsertion quota.
- Lazy reinsertion: every time a data sample exceeds the Δr_2 value threshold, directly discard it. However, when the dataset is exhausted, refill the data loader with all samples. The previously discarded samples will be checked again in the new data loader. The number of reinsertions determines the number of times we flush the data loader.

Binary clustering algorithm As we only perform binary clustering on one-dimensional data, a simple algorithm works well. In practice, we choose K-means as the binary clustering method. We also tried Otsu's method and the resulting reward model's performance is very similar.

C.4 Discussion of other reward modeling methods

In this paper, we mainly focus on the 'Sequence Classifier' reward models. We discuss the reason why we don't use this method to investigate other type of reward models.

- Generator: the "Generator" structure treats preference modeling as an instruction-following task, which incorporates the prompt, chosen and rejected response in the instruction and takes the probability of decoding a specific token as $Pr[y_1 \succ y_2|x]$ [11]. Unlike "Sequence Classifier", it explicitly leverages the backbone's language understanding ability and treat the prompt and two responses as a whole, which may mitigate prompt-free preference. More importantly, it introduces position bias(preference may vary with response order in the instruction), making the induced value of r_θ uninterpretable and unsuitable for decomposition.

- Implicit Reward (DPO): the calculation of reward value based on implicit reward model (e.g. DPO) relies on the calculation of $\log \frac{\pi(y|x)}{\pi_{ref}(y|x)}$, which can be very unstable when inputting non-corresponding prompt-response pairs. Such an issue prevents us from decomposing this unstable value.

D Additional Proofs and Interpretations

D.1 Interpretation of the random variables

We begin by interpreting the random variables defined in §3.1 in detail. Recall that they have the following definitions:

$$\begin{aligned} \tilde{Z} &= \text{Ber}\big(\mathbb{P}_{r_2}(Y_1 \succ Y_2|X)\big) = \text{Ber}\big(\sigma(\Delta r_2(X, Y_1, Y_2)\big), \\ Z &= \text{Ber}\big(\mathbb{P}_{r_1}(Y_1 \succ Y_2|X)\big) = \text{Ber}\big(\sigma(\Delta r_1(X, Y_1, Y_2)\big), \\ \tilde{W} &= \text{Ber}(\mathbb{E}_{x\sim P(X|Y_1,Y_2)}[\mathbb{P}_{r_\theta}(Y_1 \succ Y_2|x)\big)]) \\ &= \text{Ber}(\mathbb{E}_{x\sim P(X|Y_1,Y_2)}[\sigma(\Delta r_\theta(x, Y_1, Y_2))]), \\ W &= \text{Ber}\big(\mathbb{P}_{r_\theta}(Y_1 \succ Y_2|X)\big) = \text{Ber}\big(\sigma(\Delta r_\theta(X, Y_1, Y_2)\big), \end{aligned} \tag{18}$$

By the definition of the Bernoulli random variable, all these random variables can only randomly take the value of 0 or 1, while the inner probability determines the probability that the random variable is 1. However, as the inner probability is not a fixed value, which is determined by the random prompt and responses, we must determine the prompt and the responses before considering the value of Bernoulli random variables. In conclusion, since we consider the preference over the entire dataset, the randomness of such random variables comes from two aspects: 1. the randomness of the prompt and the responses. 2. the randomness inherently existing in the Bernoulli random variables.

Knowing where the randomness comes from, it's easy for us to interpret the meaning of these random variables. For W, Z and $\tilde{Z}$, they are simply random preferences of r_θ, r_1 and r_2 over the entire dataset, respectively. For these random variables, the probability of the Bernoulli variables being 1 can't be determined until we fix the prompt and the responses. For $\tilde{W}$, things get different. The inner probability of $\tilde{W}$ depends only on the responses, and retains the same value across different prompts, as we take the expectation over all the prompts. This corresponds to the intuition that $\tilde{W}$ comes from r_2, which represents the overall evaluation of the responses and is not affected by specific prompts. Moreover, we note that the conditional probability $P(X|Y_1, Y_2)$ is not the standard posterior probability. This conditional probability doesn't indicate that y_1 is more preferred than y_2 but only represents the probability that the corresponding prompt is x when the response pair contains y_1 and y_2, no matter which one is more preferred. We define in this way because $\tilde{W}$ should represent an overall preference between y_1 and y_2, so we should not only consider the prompts where y_1 is better, and should also consider the prompts where y_2 is better. Mathematically, this could be further verified since only if we define in this way, $\mathbb{E}_{x\sim P(X|Y_1=y_w,Y_2=y_l)}[\mathbb{P}_{r_\theta}(y_w \succ y_l|x)\big)] + \mathbb{E}_{x\sim P(X|Y_1=y_l,Y_2=y_w)}[\mathbb{P}_{r_\theta}(y_l \succ y_w|x)\big)] = 1$ can be satisfied. We refer to the computation of $P(X = x|Y_1 = y_1, Y_2 = y_2)$ in the end of sub-section 3.2 for a further illustration:

$$P(x|y_1, y_2) = \frac{P(x)P(y_1, y_2|x)}{P(y_1, y_2)} = \frac{P(x)P(y_1|x)P(y_2|x)}{\sum_{x'} P(x')P(y_1|x')P(y_2|x')} \tag{19}$$

D.2 Similar benefits for reward training even when additive decomposition does not hold

In the following example, we demonstrate that when r_1 and r_2 satisfy multiplicative decomposition form, our method can still control Δr_2 and thus alleviate the influence of spurious preference.

We'll start with the clarification that, due to the inner complexity of LLM-based reward model, it's hard to achieve a situation when the trained reward model r_θ exactly satisfies some specific form of decomposition that can be explicitly written out. So in the following example, we examine how our method improves the subsequent training process if r_θ already satisfies the multiplicative form of decomposition and exhibits spurious prompt-free preference.

The multiplicative decomposition can be written as:

$$r_\theta(x, y) = r_1(x, y) * r_2(x, y)$$

We continue to assume the form of $r_1(x, y)$ and $r_2(x, y)$ (which construct $r_\theta(x, y)$). Before doing so, we note that due to the multiplicative decomposition form, our method can't obtain the ground truth value of $r_1(x, y)$ and $r_2(x, y)$. Instead, our method gives an imprecise decomposition of $\Delta r_\theta = \widetilde{\Delta r_1} + \widetilde{\Delta r_2}$. The form of ground truth $r_1(x, y)$ and $r_2(x, y)$ are defined as follows:

$$r_2(x, y) = \frac{|y|}{c}, \quad \text{c = average response length over dataset}$$

$$r_1(x, y_w) = 1.0 \text{ (if } y_w \succ y_l|x), \quad 0.5 \text{ (if } y_w \approx y_l|x), \quad 0.1 \text{ (if } y_w \prec y_l|x)$$

$$r_1(x, y_l) = 0.1 \text{ (if } y_w \succ y_l|x), \quad 0.5 \text{ (if } y_w \approx y_l|x), \quad 1.0 \text{ (if } y_w \prec y_l|x)$$

It's clear that r_2 simply represents preference induced from response length, which is a prompt-free reward. As for r_1, although it has a simple form, its preference differs conditioend on different prompts. For some non-corresponding prompts, if y_l is better than y_w conditioned on these prompts, r_1 will prefer y_l more than y_w. Thus, such a prompt-related reward is also reasonable. Moreover, it can be easily verified that such a r_θ shows a strong spurious preference (length bias) towards longer responses.

To better simulate practical scenarios, we set the distribution of prompt-response pairs to be the same as the SHP dataset, while adding Gaussian noise perturbation to each $r_1(x, y)$.

$$r_1(x, y) \leftarrow r_1(x, y) + \mathcal{N}(0; 0.1)$$

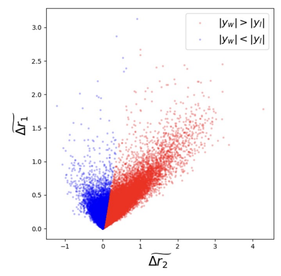
Since $r_\theta(x, y)$ is well-defined for all prompt-response pairs within the dataset distribution, we ran our method to obtain the decomposed $\widetilde{\Delta r_1}(x, y_w, y_l)$ and $\widetilde{\Delta r_2}(x, y_w, y_l)$. For a better visualization, we characterized all prompt-response pairs in the same way as Fig. 4. To further support our claim, we also conduct the same experiment based on a different r_1. The visualizations are presented below:

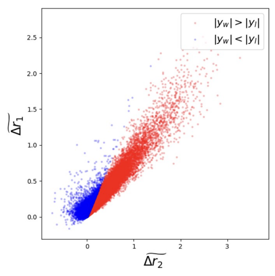
$$r_1(x, y_w) = \begin{cases} 1.0 & \text{if } y_w \succ y_l \mid x \\ 0.5 & \text{if } y_w \approx y_l \mid x \\ 0.1 & \text{if } y_w \prec y_l \mid x \end{cases}$$

$$r_1(x, y_l) = \begin{cases} 0.1 & \text{if } y_w \succ y_l \mid x \\ 0.5 & \text{if } y_w \approx y_l \mid x \\ 1.0 & \text{if } y_w \prec y_l \mid x \end{cases}$$

$$r_1(x, y_w) = \begin{cases} 0.8 & \text{if } y_w \succ y_l \mid x \\ 0.1 & \text{if } y_w \approx y_l \mid x \\ 0.3 & \text{if } y_w \prec y_l \mid x \end{cases}$$

$$r_1(x, y_l) = \begin{cases} 0.8 & \text{if } y_w \succ y_l \mid x \\ 0.1 & \text{if } y_w \approx y_l \mid x \\ 0.3 & \text{if } y_w \prec y_l \mid x \end{cases}$$





If our method still works, then $\widetilde{\Delta r_2}(x, y_w, y_l)$ should be smaller for those prompt-response pairs that satisfy $|y_w| < |y_l|$. In this way, our method will prioritize these samples and thus alleviate spurious preference. In the visualization, it's evident that although not all samples with $|y_w| < |y_l|$ have smaller values of $\widetilde{\Delta r_2}(x, y_w, y_l)$, the majority of them still lies on the left. Prioritizing samples with smaller $\widetilde{\Delta r_2}(x, y_w, y_l)$ can still mitigate spurious preference and thus outperforms naive reward training.

This exactly matches the reason we gave in the previous response. Since $\widetilde{\Delta r_2}$ can always be considered as a weighted average of Δr_θ, although the decomposition is imprecise, the spurious preference in Δr_2 can be reflected by Δr_θ. And if Δr_2 is large, then Δr_θ can be large conditioned on almost all prompts. This will lead to a large value of $\widetilde{\Delta r_2}$ and our method can filter out these samples.

D.3 Proofs of the theorem

Theorem 1. *When the value of r_2 depends only on the response, i.e. $r_2(x, y) = r_2(y)$, $\text{MI}(\tilde{Z}\|\tilde{W}) = \text{MI}(\tilde{Z}\|W)$.*

Proof. By the relation between mutual information and entropy, we first decompose the calculation of the mutual information into the calculation of the following entropy terms:

$$\text{MI}(\tilde{Z}\|\tilde{W}) = \text{H}(\tilde{Z}) + \text{H}(\tilde{W}) - H(\tilde{Z}, \tilde{W}) \tag{20}$$
$$\text{MI}(\tilde{Z}\|W) = \text{H}(\tilde{Z}) + \text{H}(W) - H(\tilde{Z}, W) \tag{21}$$

To prove that $\text{MI}(\tilde{Z}\|\tilde{W}) = \text{MI}(\tilde{Z}\|W)$, we can compare the two equations above, and with simple algebraic computation we can reduce the problem into proving:

$$\text{H}(\tilde{W}) - H(\tilde{Z}, \tilde{W}) = \text{H}(W) - H(\tilde{Z}, W) \tag{22}$$

To continue, we first calculate the Shannon entropy of $\tilde{W}$ and W. Because both $\tilde{W}$ and W are Bernoulli random variables, their discrete nature provides the possibility to directly calculate Shannon entropy by enumerating their possible values. Take $\tilde{W}$ for an example, its Shannon entropy can be calculated as:

$$\text{H}(\tilde{W}) = -Pr[\tilde{W} = 1]\log(Pr[\tilde{W} = 1]) - Pr[\tilde{W} = 0]\log(Pr[\tilde{W} = 0]) \tag{23}$$

To calculate $\text{H}(\tilde{W})$, the rest of the problem is to calculate $Pr[\tilde{W} = 1]$ and $Pr[\tilde{W} = 0]$. Again, we leverage the unique property of the Bernoulli random variable: its value can take only from 0 or 1. This allows us to turn the calculation of probability into the calculation of the expectation of the indicator function ($\mathbb{I}[a] = 1$ if $a \neq 0$ else 0):

$$\begin{aligned}
Pr[\tilde{W} = 1] &= \mathbb{E}[\mathbb{I}[\tilde{W}]] && (24)\\
&= \mathbb{E}_{P(X,Y_1,Y_2)}\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]] && (25)\\
&= \mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]] && (26)\\
&= \sum_{y_1,y_2} P(Y_1 = y_1, Y_2 = y_2)[\sum_{x} P(X = x|Y_1 = y_1, Y_2 = y_2)[\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]] && (27)\\
&= \sum_{y_1,y_2} P(Y_1 = y_1, Y_2 = y_2)[\sum_{x} P(X = x|Y_1 = y_1, Y_2 = y_2) \\
&\qquad [\mathbb{E}_{\text{Ber}}[\mathbb{I}[\text{Ber}(\mathbb{E}_{x'\sim P(X'|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x', y_1, y_2))])]]]] && (28)\\
& && (29)
\end{aligned}$$

From the first equation to the second equation, we decompose the full expectation into two separate expectations: one marginalizing the randomness in the random prompts and responses ($\mathbb{E}_{P(X,Y_1,Y_2)}$), the other marginalizing the randomness in the Bernoulli random variable ($\mathbb{E}_{\text{Ber}}$). Note that in the last equation, the inner expectation will not be affected by different prompts x outside. We then continue to simplify the last equation based on this. We change the expectation into summation since the possible values of all random variables are countable.

$$
\begin{aligned}
Pr[\tilde{W}=1] &= \sum_{y_1,y_2} P(Y_1=y_1,Y_2=y_2)[\sum_x P(X=x|Y_1=y_1,Y_2=y_2) \\
&\qquad [\mathbb{E}_{\text{Ber}}[\mathbb{I}[\text{Ber}(\mathbb{E}_{x'\sim P(X'|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x',y_1,y_2))])]]]] \quad (30)\\
&= \sum_{y_1,y_2} P(Y_1=y_1,Y_2=y_2)[\underbrace{\Big(\sum_x P(X=x|Y_1=y_1,Y_2=y_2)\Big)}_{\text{equals to 1}} \\
&\qquad [\mathbb{E}_{\text{Ber}}[\mathbb{I}[\text{Ber}(\mathbb{E}_{x'\sim P(X'|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x',y_1,y_2))])]]]] \quad (31)\\
&= \sum_{y_1,y_2} P(Y_1=y_1,Y_2=y_2)[\mathbb{E}_{\text{Ber}}[\mathbb{I}[\text{Ber}(\mathbb{E}_{x'\sim P(X'|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x',y_1,y_2))])]]] \quad (32)\\
&= \sum_{y_1,y_2} P(Y_1=y_1,Y_2=y_2)[\mathbb{E}_{x'\sim P(X'|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x',y_1,y_2))])] \quad (33)\\
&= \sum_{y_1,y_2} P(Y_1=y_1,Y_2=y_2)[\sum_{x'} P(X'=x'|Y_1=y_1,Y_2=y_2)[\sigma(\Delta r_\theta(x',y_1,y_2))])]. \quad (34)
\end{aligned}
$$

Note that here we use the following Bernoulli variable's property, which can be easily verified with the definition:

$$\mathbb{E}_{\text{Ber}}[\mathbb{I}[\text{Ber}[p]]] = p. \quad (35)$$

Similarly, $Pr[\tilde{W}=0]$ can be calculated as follows:

$$Pr[\tilde{W}=0] = \sum_{y_1,y_2} P(Y_1=y_1,Y_2=y_2)[\sum_{x'} P(X'=x'|Y_1=y_1,Y_2=y_2)[1-\sigma(\Delta r_\theta(x',y_1,y_2))])]. \quad (36)$$

This can be easily verified with the same calculation. With $Pr[\tilde{W}=1]$ and $Pr[\tilde{W}=0]$, we can easily calculate its Shannon entropy. To continue, we calculate the probabilities $Pr[W=1]$ and $Pr[W=0]$. Take the calculation of $Pr[W=1]$ as an example:

$$
\begin{aligned}
Pr[W=1] &= \mathbb{E}[\mathbb{I}[W]] \quad (37)\\
&= \mathbb{E}_{P(X,Y_1,Y_2)}\mathbb{E}_{\text{Ber}}[\mathbb{I}[W]] \quad (38)\\
&= \mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[W]]] \quad (39)\\
&= \sum_{y_1,y_2} P(Y_1=y_1,Y_2=y_2)\sum_x P(X=x|Y_1=y_1,Y_2=y_2)[\mathbb{E}_{\text{Ber}}[\mathbb{I}[W]]] \quad (40)\\
&= \sum_{y_1,y_2} P(Y_1=y_1,Y_2=y_2)\sum_x P(X=x|Y_1=y_1,Y_2=y_2)[\mathbb{E}_{\text{Ber}}[\mathbb{I}[\text{Ber}\big(\sigma(\Delta r_\theta(X,Y_1,Y_2))\big)]]] \quad (41)\\
&= \sum_{y_1,y_2} P(Y_1=y_1,Y_2=y_2)[\sum_x P(X=x|Y_1=y_1,Y_2=y_2)[\sigma(\Delta r_\theta(x,y_1,y_2))])]. \quad (42)
\end{aligned}
$$

Similarly, $Pr[W=0]$ can be calculated as:

$$Pr[W=0] = \sum_{y_1,y_2} P(Y_1=y_1,Y_2=y_2)[\sum_x P(X=x|Y_1=y_1,Y_2=y_2)[1-\sigma(\Delta r_\theta(x,y_1,y_2))])]. \quad (43)$$

Combining the results of calculating $Pr[\tilde{W}=1], Pr[\tilde{W}=0], Pr[W=1], Pr[W=0]$, we can easily verify that $\text{H}(\tilde{W}) = \text{H}(W)$. It's worth noting that $\text{H}(\tilde{W}) = \text{H}(W)$ only means their separate quantity of information is the same, but doesn't characterize their relationship from the information perspective.

The rest of the problem is to prove the equality of $\text{H}(\tilde{Z},\tilde{W}) = \text{H}(\tilde{Z},W)$. Once again, since the events considering $(\tilde{Z},\tilde{W})$ or $(\tilde{Z},W)$ pair are both finite, we can decompose the calculation of the joint entropy as the summation of four terms. Take the calculation of $\mathrm{H}(\tilde{Z}, \tilde{W})$ as an example:

$$\mathrm{H}(\tilde{Z}, \tilde{W}) = -Pr[\tilde{Z}=1, \tilde{W}=1]\log(Pr[\tilde{Z}=1, \tilde{W}=1]) - Pr[\tilde{Z}=1, \tilde{W}=0]\log(Pr[\tilde{Z}=1, \tilde{W}=0]) - Pr[\tilde{Z}=0, \tilde{W}=1]\log(Pr[\tilde{Z}=0, \tilde{W}=1]) - Pr[\tilde{Z}=0, \tilde{W}=0]\log(Pr[\tilde{Z}=0, \tilde{W}=0]). \tag{44}$$

To prove that $\mathrm{H}(\tilde{Z}, \tilde{W}) = \mathrm{H}(\tilde{Z}, W)$, we prove that the four probabilities induced from $(\tilde{Z}, \tilde{W})$ equal to that induced from $(\tilde{Z}, W)$, respectively. For simplicity, we only prove that $Pr[\tilde{Z}=1, \tilde{W}=1] = Pr[\tilde{Z}=1, W=1]$, while the remaining three equations can be proved in the same way.

The key property we used to prove such equations is that although there exists a complicated relationship between $\tilde{Z}$, $\tilde{W}$, and W, the randomness from different inherent Bernoulli distributions is independent. In other words, once the prompt, responses are determined for all these random variables, the inner probability is a fixed value that is independent from each other. This motivates us to use the same proving technique above, decomposing the full expectation into separate expectations about the prompt-response pair and the Bernoulli distribution, respectively. Specifically, the following holds:

$$Pr[\tilde{Z}=1, \tilde{W}=1] = \mathbb{E}[\mathbb{I}[\tilde{Z}\cdot\tilde{W}]] \tag{45}$$
$$= \mathbb{E}_{P(X,Y_1,Y_2)}\mathbb{E}_{\text{Ber}\times\text{Ber}}[\mathbb{I}[\tilde{Z}\cdot\tilde{W}]] \tag{46}$$
$$= \mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}\times\text{Ber}}[\mathbb{I}[\tilde{Z}\cdot\tilde{W}]]] \tag{47}$$
$$= \sum_{y_1,y_2} P(Y_1=y_1, Y_2=y_2)\sum_x P(X=x|Y_1=y_1,Y_2=y_2)[\mathbb{E}_{\text{Ber}\times\text{Ber}}[\mathbb{I}[\tilde{Z}\cdot\tilde{W}]]]. \tag{48}$$
$$\tag{49}$$

Note that here $\mathbb{E}_{\text{Ber}\times\text{Ber}}$ represents the expectation over two independent Bernoulli random variables. The independence comes from different Bernoulli random variables with fixed probability that take the value of 1. We continue to expand $\mathbb{E}_{\text{Ber}\times\text{Ber}}[\mathbb{I}[\tilde{Z}\cdot\tilde{W}]]$ with the independence with the following equation:

$$\mathbb{E}_{\text{Ber}\times\text{Ber}}[\mathbb{I}[\tilde{Z}\cdot\tilde{W}]] = \mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{Z}]]\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]. \tag{50}$$

Replacing this in the previous equation, we have:

$$Pr[\tilde{Z}=1, \tilde{W}=1] = \sum_{y_1,y_2} P(Y_1=y_1, Y_2=y_2)\sum_x P(X=x|Y_1=y_1,Y_2=y_2)[\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{Z}]]\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]] \tag{51}$$

$$= \sum_{y_1,y_2} P(Y_1=y_1, Y_2=y_2)\sum_x P(X=x|Y_1=y_1,Y_2=y_2) [\sigma(\Delta r_2(y_1,y_2))\cdot \underbrace{\mathbb{E}_{x'\sim P(X'|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x',y_1,y_2))]}_{\text{independent of } x}] \tag{52}$$

$$= \sum_{y_1,y_2} P(Y_1=y_1, Y_2=y_2)\Bigg[\Big(\underbrace{\sum_x P(X=x|Y_1=y_1,Y_2=y_2)}_{\text{equals to 1}}\cdot\sigma(\Delta r_2(y_1,y_2))\Big) \Big(\sum_{x'} P(X'=x'|Y_1=y_1,Y_2=y_2)\cdot\sigma(\Delta r_\theta(x',y_1,y_2))\Big)\Bigg] \tag{53}$$

$$= \sum_{y_1,y_2} P(Y_1=y_1, Y_2=y_2)\Bigg[\sigma(\Delta r_2(y_1,y_2))\cdot\Big(\sum_{x'} P(X'=x'|Y_1=y_1,Y_2=y_2)\cdot\sigma(\Delta r_\theta(x',y_1,y_2))\Big)\Bigg] \tag{54}$$

Note that to get the last equation, we first use Eq. (35) to simplify the inner expectations, and then exchange the sequence from first taking the product to first taking the summation over x. Such exchange holds since one part of the product is independent of x. Moreover, we replace $\Delta r_2(x, y_1, y_2)$ with $\Delta r_2(y_1, y_2)$ due to the assumption of the theorem, which is vital in the proof.

We continue to calculate $Pr[\tilde{Z} = 1, W = 1]$, the difference is that W is not independent of x. However, the exchange we used above still holds since the specific form $\Delta r_2(y_1, y_2)$ does not depend on x. We have:

$$Pr[\tilde{Z} = 1, \tilde{W} = 1] = \sum_{y_1,y_2} P(Y_1 = y_1, Y_2 = y_2) \sum_x P(X = x|Y_1 = y_1, Y_2 = y_2)[\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{Z}]]\mathbb{E}_{\text{Ber}}[\mathbb{I}[W]]] \tag{55}$$

$$= \sum_{y_1,y_2} P(Y_1 = y_1, Y_2 = y_2) \sum_x P(X = x|Y_1 = y_1, Y_2 = y_2)[\sigma(\Delta r_2(y_1, y_2)) \cdot \sigma(\Delta r_\theta(x, y_1, y_2))] \tag{56}$$

$$= \sum_{y_1,y_2} P(Y_1 = y_1, Y_2 = y_2) \left[\sigma(\Delta r_2(y_1, y_2)) \cdot \Big(\sum_x P(X = x|Y_1 = y_1, Y_2 = y_2) \cdot \sigma(\Delta r_\theta(x, y_1, y_2)) \Big) \right] \tag{57}$$

It's easy to verify that $Pr[\tilde{Z} = 1, \tilde{W} = 1] = Pr[\tilde{Z} = 1, \tilde{W} = 1]$. With similar derivation, we can prove the equalities for the rest three probabilities. Combined them together, we prove that $\text{H}(\tilde{Z}, \tilde{W}) = \text{H}(\tilde{Z}, W)$ and finish all the proofs. ☐

Theorem 2. *For any bounded r_θ and dataset (X, Y_1, Y_2), there exist **feasible** r_1^*, r_2^* such that $\forall (y_1, y_2) \sim P(Y_1, Y_2), \mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_1^*(x, y_1, y_2))] = \frac{1}{2}$. Such r_1^*, r_2^* is the optimal solution to problem* (9)*.*

Proof. We begin by proving that there exists r_1^* and r_2^* that satisfy $\forall (y_1, y_2) \sim P(Y_1, Y_2), \mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_1^*(x, y_1, y_2))] = \frac{1}{2}$. We then verify that such r_1^* and r_2^* are feasible, by verifying that such r_1^* and r_2^* satisfy the constraints and can recover r_θ (i.e. $\Delta r_\theta(x, y_1, y_2) = \Delta r_1^*(x, y_1, y_2) + \Delta r_2^*(y_1, y_2)$).
To prove the existence of such r_1^* and r_2^*, we first replace $\Delta r_1^*(x, y_1, y_2)$ with $\Delta r_\theta(x, y_1, y_2) - \Delta r_2^*(y_1, y_2)$. In the following proof, we have to use the following lemma:

Lemma 1. $\mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x, y_1, y_2) - \Delta r_2^*(y_1, y_2))]$ *monotonically decrease with the increase of* $\Delta r_2^*(y_1, y_2)$

Proof. We directly take the derivative of $\mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x, y_1, y_2) - \Delta r_2^*(y_1, y_2))]$. Since $\Delta r_2^*(y_1, y_2))$ retains the same across different x, we have:

$$\frac{\partial[\mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x, y_1, y_2) - \Delta r_2^*(y_1, y_2))]]}{\partial \Delta r_2^*(y_1, y_2)} \tag{58}$$

$$= \frac{\mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\partial \sigma(\Delta r_\theta(x, y_1, y_2) - \Delta r_2^*(y_1, y_2))]}{\partial \Delta r_2^*(y_1, y_2)} \tag{59}$$

$$= \mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\frac{\partial \frac{1}{1+\exp(\Delta r_2^*(y_1,y_2) - \Delta r_\theta(x,y_1,y_2))}}{\partial(\Delta r_2^*(y_1, y_2) - \Delta r_\theta(x, y_1, y_2))} \cdot \frac{\partial(\Delta r_2^*(y_1, y_2) - \Delta r_\theta(x, y_1, y_2))}{\Delta r_2^*(y_1, y_2)}] \tag{60}$$

$$= \mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[-\frac{\exp(\Delta r_2^*(y_1, y_2) - \Delta r_\theta(x, y_1, y_2))}{(1 + \exp(\Delta r_2^*(y_1, y_2) - \Delta r_\theta(x, y_1, y_2)))^2}] \tag{61}$$

This shows that for any y_1, y_2, $\mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x, y_1, y_2) - \Delta r_2^*(y_1, y_2))]$ monotonically decrease with the increase of $\Delta r_2^*(y_1, y_2)$. ☐

To continue the proof of Theorem 2, we identify the lower and the upper bounds of $\mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x, y_1, y_2) - \Delta r_2^*(y_1, y_2))]$, under different choices of $\Delta r_2^*(y_1, y_2)$.

Due to the assumption of bounded r_θ, we consider r_θ that satisfies $\forall x, y_1, y_2, \quad r_\theta(x, y_1, y_2) \in [r_{min}, r_{max}]$. The following holds:

$$\mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x, y_1, y_2) - (r_{min} - r_{max}))] \tag{62}$$

$$= \mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma\Big((r_\theta(x, y_1) - r_\theta(x, y_2)) - (r_{min} - r_{max})\Big)] \tag{63}$$

$$= \mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma\Big(\underbrace{(r_\theta(x, y_1) - r_{min})}_{\text{always} \geq 0} + \underbrace{(r_{max} - r_\theta(x, y_2))}_{\text{always} \geq 0}\Big)] \tag{64}$$

$$\geq \mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\frac{1}{2}] = \frac{1}{2} \tag{65}$$

On the other hand, the following inequality also characterizes the lower bound of $\mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x, y_1, y_2) - \Delta r_2^*(y_1, y_2))]$:

$$\mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x, y_1, y_2) - (r_{max} - r_{min}))] \tag{66}$$

$$= \mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma\Big((r_\theta(x, y_1) - r_\theta(x, y_2)) - (r_{max} - r_{min})\Big)] \tag{67}$$

$$= \mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma\Big(\underbrace{(r_\theta(x, y_1) - r_{max})}_{\text{always} \leq 0} + \underbrace{(r_{min} - r_\theta(x, y_2))}_{\text{always} \leq 0}\Big)] \tag{68}$$

$$\leq \mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\frac{1}{2}] = \frac{1}{2} \tag{69}$$

Combine with the fact that $\mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x, y_1, y_2) - \Delta r_2^*(y_1, y_2))]$ monotonically decrease with the increase of $\Delta r_2^*(y_1, y_2)$, there always exists $\Delta r_2^*(y_1, y_2)$ such that $\mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x, y_1, y_2) - \Delta r_2^*(y_1, y_2))] = \frac{1}{2}$. To satisfy the feasibility condition that $\Delta r_\theta(x, y_1, y_2) = \Delta r_1(x, y_1, y_2) + \Delta r_2(y_1, y_2)$, we directly set $\Delta r_1^*(x, y_1, y_2) = \Delta r_\theta(x, y_1, y_2) - \Delta r_2^*(y_1, y_2)$, with the $\Delta r_2^*(y_1, y_2)$ obtained before. Such $\Delta r_2^*(y_1, y_2)$ still satisfies the simple structure mentioned in Theorem 1, so the constraint MI$(\tilde{Z} \parallel \tilde{W}) =$ MI$(\tilde{Z} \parallel W)$ is still satisfied. The remaining task is to proof that the $\Delta r_1^*(x, y_1, y_2)$ obtained before satisfy the constraint MI$(Z \parallel \tilde{W}) = 0$.

With the same proving technique in the proof of Theorem 1, we first decompose MI$(Z \parallel \tilde{W})$ into three entropy terms:

$$\text{MI}(Z \parallel \tilde{W}) = \text{H}(Z) + \text{H}(\tilde{W}) - \text{H}(Z, \tilde{W}) \tag{70}$$

We further expand the right side of the equation in the following equations:

$$\begin{aligned}
\text{H}(Z) &= -Pr[Z=1]\log(Pr[Z=1]) - Pr[Z=0]\log(Pr[Z=0]) \\
\text{H}(\tilde{W}) &= -Pr[\tilde{W}=1]\log(Pr[\tilde{W}=1]) - Pr[\tilde{W}=0]\log(Pr[\tilde{W}=0]) \\
\text{H}(Z, \tilde{W}) &= -Pr[Z=1, \tilde{W}=1]\log(Pr[Z=1, \tilde{W}=1]) - Pr[Z=1, \tilde{W}=0]\log(Pr[Z=1, \tilde{W}=0]) \\
&\quad - Pr[Z=0, \tilde{W}=1]\log(Pr[Z=0, \tilde{W}=1]) - Pr[Z=0, \tilde{W}=0]\log(Pr[Z=0, \tilde{W}=0])
\end{aligned} \tag{71}$$

With simple algebraic simplification, we have:

$$\begin{aligned}
&\text{H}(Z) + \text{H}(\tilde{W}) - \text{H}(Z, \tilde{W}) \\
&= Pr[Z=1, \tilde{W}=1]\log(Pr[Z=1, \tilde{W}=1]) + Pr[Z=1, \tilde{W}=0]\log(Pr[Z=1, \tilde{W}=0]) \\
&+ Pr[Z=0, \tilde{W}=1]\log(Pr[Z=0, \tilde{W}=1]) + Pr[Z=0, \tilde{W}=0]\log(Pr[Z=0, \tilde{W}=0]) \\
&- Pr[Z=1]\log(Pr[Z=1]) - Pr[Z=0]\log(Pr[Z=0]) - Pr[\tilde{W}=1]\log(\tilde{W}=1) \\
&- Pr[\tilde{W}=0]\log(\tilde{W}=0)
\end{aligned} \tag{72}$$

To prove that MI$(Z \parallel \tilde{W}) = \text{H}(Z) + \text{H}(\tilde{W}) - \text{H}(Z, \tilde{W}) = 0$, we first expand the inner probability with the same technique use in proving Theorem 1, and then perform the following transformation for $\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]$ and $\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]$:

$$\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]] = \mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]] \cdot \Big(\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]] + (1 - \mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]])\Big) \tag{73}$$

$$\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]] = \mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]] \cdot \Big(\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]] + (1 - \mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]])\Big) \tag{74}$$

Note that this simple transformation bridges all terms. Take the calculation of $Pr[Z = 1, \tilde{W} = 1] \log(Pr[Z = 1, \tilde{W} = 1]) - Pr[Z = 1] \log(Pr[Z = 1]) - Pr[\tilde{W} = 1] \log(\tilde{W} = 1)$ for an example, we expand the inner probability with the same technique use in proving Theorem 1:

$$Pr[Z = 1, \tilde{W} = 1] \log(Pr[Z = 1, \tilde{W} = 1]) - Pr[Z = 1] \log(Pr[Z = 1]) - Pr[\tilde{W} = 1] \log(\tilde{W} = 1) \tag{75}$$

$$\begin{aligned} &= \mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}\times\text{Ber}}[\mathbb{I}[Z \cdot \tilde{W}]]] \log(\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}\times\text{Ber}}[\mathbb{I}[Z \cdot \tilde{W}]]]) \\ &- \mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]] \log(\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]]) \\ &- \mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]] \log(\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]]) \end{aligned} \tag{76}$$

After using the previously mentioned transformation, we have the right-hand-side above equals to:

$$\begin{aligned} &\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}\times\text{Ber}}[\mathbb{I}[Z \cdot \tilde{W}]]] \log(\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}\times\text{Ber}}[\mathbb{I}[Z \cdot \tilde{W}]]]) \\ &- \mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}\Big[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]] \cdot \mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]\Big] * \\ &\qquad \log\Big(\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]] \cdot \mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]]\Big) \\ &- \underbrace{\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}\Big[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]] \cdot (1 - \mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]])\Big] \log(\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]])}_{\text{used for symmetric construction for canceling other term}} \\ &- \underbrace{\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}\Big[(1 - \mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]) \cdot \mathbb{E}_{\text{Ber}}[\mathbb{I}[W]]\Big] \log(\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]])}_{\text{used for symmetric construction for canceling other term}} \end{aligned} \tag{77}$$

With simple verification, we can find that the last two terms can be used to cancel other terms that come from $\mathrm{H}(Z, \tilde{W})$. This means we only have to prove that the first two terms in the above equation cancel out. In other words, all we have to prove is:

$$\begin{aligned} &\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}\times\text{Ber}}[\mathbb{I}[Z \cdot \tilde{W}]]] \log(\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}\times\text{Ber}}[\mathbb{I}[Z \cdot \tilde{W}]]]) \\ &- \mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}\Big[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]] \cdot \mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]\Big] * \\ &\qquad \log\Big(\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]] \cdot \mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]]\Big) \\ &= \mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}\Big[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]] \cdot \mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]\Big] * \\ &\qquad \log\left(\frac{\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}\Big[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]] \cdot \mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]\Big]}{\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]] \cdot \mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]]}\right) \\ &= 0 \end{aligned} \tag{78}$$

Note that we change all $\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}\times\text{Ber}}[\mathbb{I}[Z \cdot \tilde{W}]]]$ into $\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}\Big[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]] \cdot \mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]\Big]$ due to independency similar as before. We can finally use the property that for any y_1, y_2, $\mathbb{E}_{x \sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_\theta(x, y_1, y_2) - \Delta r_2^*(y_1, y_2))] = \frac{1}{2}$, and remember that $\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]$ is independent of specific x, then we can perform the following simplification:

$$\begin{aligned}
&\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}\left[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]\cdot\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]\right]*\\
&\qquad\log\left(\frac{\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}\left[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]\cdot\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]\right]}{\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]]\cdot\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]]}\right)\\
&=\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}\left[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]\cdot\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]\right]*\\
&\qquad\log\left(\frac{\mathbb{E}_{P(Y_1,Y_2)}\left[\left(\mathbb{E}_{P(X|Y_1,Y_2)}\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]\right)\cdot\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]\right]}{\mathbb{E}_{P(Y_1,Y_2)}\left(\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]]\right)\cdot\mathbb{E}_{P(Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]]}\right)\\
&=\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}\left[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]\cdot\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]\right]\log\left(\frac{\mathbb{E}_{P(Y_1,Y_2)}\left[\left(\frac{1}{2}\right)\cdot\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]\right]}{\mathbb{E}_{P(Y_1,Y_2)}\left(\frac{1}{2}\right)\cdot\mathbb{E}_{P(Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]]}\right)\\
&=\mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}\left[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]\cdot\mathbb{E}_{\text{Ber}}[\mathbb{I}[\tilde{W}]]\right]\log\left(\frac{\frac{1}{2}}{\frac{1}{2}}\right)=0
\end{aligned}\tag{79}$$

After verifying the feasibility of the solution, we verify that such r_1^* and r_2^* is optimal for maximizing $\mathrm{H}(Z)$. By definition, we have:

$$\mathrm{H}(Z) = -Pr[Z=1]\log(Pr[Z=1]) - Pr[Z=0]\log(Pr[Z=0]) \tag{80}$$

To calculate $Pr[Z=1]$, we can use the same expansion technique before. We have:

$$\begin{aligned}
Pr[Z=1] &= \mathbb{E}_{P(Y_1,Y_2)}\mathbb{E}_{P(X|Y_1,Y_2)}[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]]\\
&= \sum_{y_1,y_2} P(Y_1=y_1,Y_2=y_2)[\sum_x P(X=x|Y_1=y_1,Y_2=y_2)[\mathbb{E}_{\text{Ber}}[\mathbb{I}[Z]]]\\
&= \sum_{y_1,y_2} P(Y_1=y_1,Y_2=y_2)[\sum_x P(X=x|Y_1=y_1,Y_2=y_2)[\sigma(\Delta r_1^*(x,y_1,y_2))]]\\
&= \sum_{y_1,y_2} P(Y_1=y_1,Y_2=y_2)[\mathbb{E}_{x\sim P(X|Y_1=y_1,Y_2=y_2)}[\sigma(\Delta r_1^*(x,y_1,y_2))] = \frac{1}{2}
\end{aligned}\tag{81}$$

It's easy to verify that such r_1^* and r_2^* reach the maximum value of $\mathrm{H}(Z)$, which is the optimal solution we want. □

E More about experiments

E.1 Full results in Table 1 and Table 2

Due to space limits, we only reported mean scores from 3 random seeds in the main context. Full results (mean and std) in Table 1 and Table 2 are listed below, showing consistent performance among different seeds.

Model	Chat	Chat Hard	Safety	Reason	Avg
vanilla	78.0±2.1	29.8±0.7	36.4±0.9	58.2±1.1	50.6±1.0
ours	86.8±1.8	31.1±0.8	45.1±0.8	60.3±1.2	55.8±1.1

Table 3: Full results of experiments in Table 1 (a). We conduct the same experiments with 3 different random seeds and report the mean and std values of the results.

Model	Chat	Chat Hard	Safety	Reason	Avg
vanilla	80.7±2.1	28.5±0.8	40.2±1.0	36.5±1.5	46.4±1.3
original	84.9±1.7	31.3±0.6	42.8±1.1	49.0±1.7	52.0±0.9
ours	84.3±1.8	30.9±0.5	43.2±0.9	46.7±1.0	51.6±0.8

Table 4: Full results of experiments in Table 1 (b). We conduct the same experiments with 3 different random seeds and report the mean and std values of the results.

Method	Chat	Chat Hard	Safety	Reasoning	Average
vanilla-8B	0.93±0.01	0.50±0.02	0.67±0.02	0.78±0.03	0.72±0.02
RRM-8B	0.95±0.01	0.56±0.01	0.75±0.02	0.82±0.02	0.77±0.01
Ours-8B	**0.96**±0.02	**0.59**±0.01	**0.81**±0.01	**0.89**±0.02	**0.82**±0.01
vanilla-7B	0.90±0.02	0.49±0.03	0.60±0.01	0.69±0.05	0.66±0.03
RRM-7B	0.94±0.01	0.52±0.01	0.65±0.01	0.70±0.03	0.70±0.01
Ours-7B	**0.94**±0.01	**0.56**±0.02	**0.69**±0.02	**0.81**±0.04	**0.75**±0.02

Table 5: Full results of experiments in Table 2 (a). We conduct the same experiments with 3 different random seeds and report the mean and std values of the results.

Method	vanilla	RRM	Ours
DPO	30.5±1.6 / 38.8±1.3	39.5±0.6 / 40.9±0.8	**40.6**±1.2 / **42.3**±0.9
BoN(N=4)	33.2±2.1 / 42.5±1.7	35.3±1.4 / 38.4±1.5	**38.5**±1.0 / **45.0**±1.1
BoN(N=32)	35.8±1.8 / 45.1±2.0	41.1±1.9 / 44.9±1.2	**44.7**±1.7 / **48.3**±0.9

Table 6: Full AlpacaEval-2 (LCWR / WR) results. We conduct the same experiments with 3 different random seeds and report the mean and std values of the results.

Method	vanilla	RRM	Ours
DPO	7.83±0.12 / 6.51±0.30	8.35±0.15 / 7.46±0.24	**8.44**±0.10 / **8.03**±0.26

Table 7: Full MT-Bench (T1 / T2) results. We conduct the same experiments with 3 different random seeds and report the mean and std values of the results.

E.2 Reward-Bench results based on SHP dataset

Method	Chat	Chat Hard	Safety	Reasoning	Average
vanilla-8B	0.91±0.02	0.39±0.01	0.46±0.01	0.75±0.02	0.63±0.01
RRM-8B	0.93±0.02	0.45±0.01	0.54±0.03	0.77±0.01	0.67±0.02
Ours-8B	**0.95**±0.01	**0.46**±0.01	**0.61**±0.01	**0.88**±0.02	**0.72**±0.01
vanilla-7B	0.84±0.02	0.33±0.03	0.45±0.05	0.59±0.03	0.55±0.04
RRM-7B	0.89±0.02	0.37±0.01	0.48±0.01	0.58±0.04	0.58±0.02
Ours-7B	**0.90**±0.01	**0.39**±0.01	**0.52**±0.01	**0.74**±0.03	**0.64**±0.02

Table 8: Results on Reward-Bench: These experiments are based on the same two reward backbones, but with the SHP dataset. The improvement brought by our method is also significant on the SHP dataset. However, since the SHP dataset contains many fewer data samples and may inherently have more spurious preferences, the accuracy generally decreases.

E.3 Justification and analysis of binary clustering and EMA threshold used in data prioritization

An adaptive threshold is required since reward learning is dynamic—Figure 4 show that even small models (1B) experience significant Δr_2 scale changes during training.

During optimization, the reward model is updated in batches, with Δr_2 calculated from limited samples each step. Determining whether a sample's Δr_2 is "relatively small" in a batch becomes a 1D binary-clustering problem(Δr_2 as feature). Due to high variance in Δr_2 values among different samples (as shown in Figure 4), relying solely on current batch data is unstable and cannot truly reflect the properties of Δr_2. Our dynamic threshold, computed via exponential moving average, incorporates recent training steps, dynamically capturing Δr_2 properties while reducing threshold variance.

As for the robustness, we conduct the same experiments on Reward-Bench in Section 4.2 with different EMA weights (α in Alg. 2) and clustering methods (Otsu, K-means). Results show robustness to thresholding methods. Limited by time and budget, we only use LLaMA-3-8B-Instruct and SHP dataset. We believe the robustness partially stems from the fact that samples with larger Δr_2 in the current batch will not be directly discarded, preserving the chance to be used in the future step.

Method	α	Chat	Chat Hard	Safety	Reasoning	Average
Otsu	0.9	0.95	0.48	0.59	0.87	0.72
Otsu	0.8	0.92	0.49	0.57	0.84	0.70
K-means	0.9	0.95	0.46	0.61	0.88	0.72
K-means	0.8	0.93	0.48	0.60	0.87	0.71

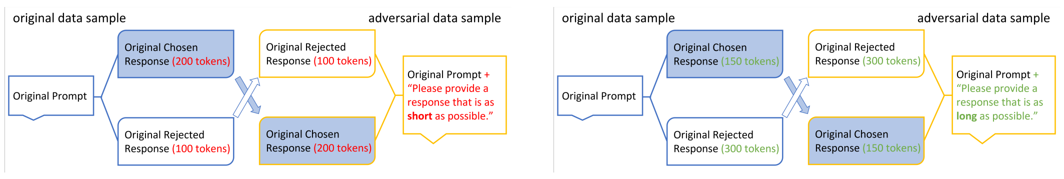
Table 9: Performance under different methods and α values.

E.4 experimental details

Our implementation is based on the OpenRLHF [16] framework, which uses Apache-2.0 license. Experiments are run on Nvidia A100(40G) GPUs. For reward training, we use 8*A100(40G) GPUs so that the training can be finished in 12 hours. For DPO training and response generation, we also use 8*A100(40G) GPUs, and the time consumption is similar to the reward training. All the reward models and the LLMs are fully fine-tuned. We use Deepspeed [2] as the framework of parallelization. For the implementation of RRM, we reference its official code in RLHFlow[3]. We use an AdamW optimizer with a learning rating of 5e-7 and 9e-6 for the DPO policy and the reward model, respectively. For other general hyperparameters, we follow the default parameters in OpenRLHF.

Our exclusive hyperparameters include clustering methods and EMA weights for Δr_2 thresholding. In practice, we tried Otsu's method and K-means for clustering method and 0.8, 0.9 for EMA weight. As is shown in Appendix E.3, our method performs consistently, so we select the best-performing hyperparameters.

We continue to explain the data construction process for the toy cases. For the length-biased dataset, we use all preference pairs (x, y_w, y_l) where the chosen response is longer, i.e., $|y_w| > |y_l|$, from the original dataset; based on the number of chosen-longer samples, we select other samples with $\frac{1}{4}$ number of them, whose chosen response is shorter. The combined subsets create a clearly length-biased dataset. For the adversarial prompt dataset, we use the following diagram to illustrate the data processing:


original data sample
adversarial data sample
Original Prompt
Original Chosen Response (200 tokens)
Original Rejected Response (100 tokens)
Original Rejected Response (100 tokens)
Original Chosen Response (200 tokens)
Original Prompt + "Please provide a response that is as short as possible."
original data sample
adversarial data sample
Original Prompt
Original Chosen Response (150 tokens)
Original Rejected Response (300 tokens)
Original Rejected Response (300 tokens)
Original Chosen Response (150 tokens)
Original Prompt + "Please provide a response that is as long as possible."

We note that RLHFLow uses Apache-2.0 license. SHP dataset didn't claim the license it uses. LLaMA models use Meta Llama 3 Community License Agreement. Mistral models use Apache-2.0 license.

E.5 Explanations of the superiority compared with baseline methods

The advantage of our method compared with vanilla training is obvious. Since vanilla training will use all data samples, including those samples with strong prompt-free preference. Thus, the resulting reward model easily overfits to some prompt-free preference existing in the dataset. Such prompt-free preference harms the reward model's generalization capability and leads to a low-quality alignment.

Compared with RRM, although both RRM and our method mitigate spurious preferences by altering the data distribution, RRM introduces hand-crafted but mismatched prompt-response pairs into the dataset and updates the reward model on them, which in turn affects the meaningful preferences. In contrast, our method leverages them solely to characterize the reward model during training and only trains on the original preference data, where each prompt-response pair is labeled by human preference. This contributes to the superior performance of our method compared to RRM.

[3]https://github.com/RLHFlow

F Boarder Impacts

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work. For positive ones, our method could result in a more generalizable reward model and thus advance other RLHF methods. This can make the LLMs more aligned with human preferences. For negative ones, one can achieve jailbreaking in LLMs through malicious reward models.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our main claims can be reflected by Section 3 and Section 4.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in Section 6

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Seed Appendix D.3

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Experimental details are given in Appendix E.4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: Code will be released in the future.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Experimental settings are given in Section 4 and Appendix E.4. Details are given in Appendix E.4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We use sufficient random seeds and provide standard deviation of the scores in Appendix E.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Details of computation resources are given in Appendix E.4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This paper doesn't involve human participants, and the datasets in Section 4 are common open-sourced datasets.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Potential positive impacts and negative impacts are discussed in Appendix F.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

Justification:

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All assets we use are properly credited, and the corresponding license are given in Appendix E.4.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification:

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.